Security and Privacy in Heterogeneous Wireless and Mobile Networks: Challenges and Solutions

by

Rui Zhang

A Dissertation Presented in Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy

Approved June 2013 by the
Graduate Supervisory Committee:

Yanchao Zhang, Chair
Tolga Mete Duman
Guoliang Xue
Junshan Zhang

ARIZONA STATE UNIVERSITY

August 2013

ABSTRACT

The rapid advances in wireless communications and networking have given rise to a number of emerging heterogeneous wireless and mobile networks along with novel networking paradigms, including wireless sensor networks, mobile crowdsourcing, and mobile social networking. While offering promising solutions to a wide range of new applications, their widespread adoption and large-scale deployment are often hindered by people's concerns about the security, user privacy, or both. In this dissertation, we aim to address a number of challenging security and privacy issues in heterogeneous wireless and mobile networks in an attempt to foster their widespread adoption.

Our contributions are mainly fivefold. First, we introduce a novel secure and loss-resilient code dissemination scheme for wireless sensor networks deployed in hostile and harsh environments. Second, we devise a novel scheme to enable mobile users to detect any inauthentic or unsound location-based top-k query result returned by an untrusted location-based service providers. Third, we develop a novel verifiable privacy-preserving aggregation scheme for people-centric mobile sensing systems. Fourth, we present a suite of privacy-preserving profile matching protocols for proximity-based mobile social networking, which can support a wide range of matching metrics with different privacy levels. Last, we present a secure combination scheme for crowdsourcing-based cooperative spectrum sensing systems that can enable robust primary user detection even when malicious cognitive radio users constitute the majority.

Dedicated to my family

ACKNOWLEDGEMENTS

The last five and half years have been an exciting and unforgettable journey of learning and exploring. I am very fortunate to have Prof. Yanchao Zhang as my advisor, to whom I would like to express my sincerest gratitude for his invaluable guidance, encouragement and support over the course of my PhD study. Besides research, he has also provided thoughtful advice to my career development as well as warm-hearted help to my personal life. I am deeply indebted to Prof. Zhang and wish I could be a good advisor like him for my own students in the future.

I would also like to thank other supervisory committee members, Prof. Tolga Duman, Prof. Guoliang Xue, and Prof. Junshan Zhang, not only for serving on my supervisory committee, but also for their help and guidance at various stages of my PhD study and career.

I would also like to thank Prof. Nirwan Ansari and Prof. Mengchu Zhou for their guidance and help when I was a PhD student at New Jersey Institute of Technology.

I would also like to thank my former and current labmates, Dr. Jing shi, Yunzhong Liu, Jingchao Sun, Jinxue Zhang, and Xiaocong Jin. It has been my great pleasure to work with them on various exciting projects and learn from each other.

I would also like to thank my colleagues and friends, Dejun Yang, Dr. Xi Fang, Dr. Jin Zhang, Zhiming Zhao, Dr. Bo Niu, Dr. Jingjing Zhang, for enjoyable and inspiring discussions. Their friendship is a great gift to me.

Last but not least, I would like to thank my family. I am sincerely grateful to my wife, Dr. Rong Luo. Without her endless love, encouragement and support, whatever I have achieved could never be possible. I would also like to thank my daughter, Olivia, who has brought a lot of fun to the family since she was born. I am also extremely thankful to my parents, Ronghua Ding and Jiazheng Zhang, and my parents-in-law, Zhiyu Li and Shenghua Luo, who are always good listeners and offer their support without any hesitation.

TABLE OF CONTENTS

CHAPTER Page

CHAPTER Page

CHAPTER Page

CHAPTER Page

CHAPTER Page

LIST OF TABLES

LIST OF FIGURES

Figure Page

Chapter 1

INTRODUCTION

The rapid advances in wireless communications and networking have given rise to a number of emerging heterogeneous wireless and mobile networks along with novel and promising networking paradigms. Wireless sensor networks, formed by a large number of low-cost, low-power, and multi-functional sensor nodes with communication capability, have been widely considered as ideal candidates for a wide range of applications such as health monitoring, environment monitoring, and military operations. Participatory mobile sensing and crowdsourcing systems, powered by the explosive growth of smartphones and tablets with ever-growing capabilities in sensing, computation, storage, and communications, makes it possible for people-centric urban-scale distributed data collection, analysis, and sharing. Mobile social networking, driven by the marriage between traditional web-based social networks and mobile devices, makes it easier than ever for people to stay connected and interact with each other at anywhere and anytime. These heterogeneous wireless and mobile networks are expected to drastically change people's daily lives.

While offering promising solutions to a wide range of applications, the widespread adoption and large-scale deployment of these emerging heterogeneous wireless and mobile network are often hindered by the concerns about the system security, user privacy, or both. On the one hand, these heterogeneous wireless and mobile networks not only inherit all the security vulnerabilities of traditional wired networks, but also face new security challenges raised by the openness of wireless medium, the lack of trust of individual participating nodes, the resource constraints of mobile devices, and so on. On the other hand, people have growing concerns about disclosing their personal information, data access pattern, etc, which may be used against their interests. In other words, potential users may not want to adopt emerging networking paradigms if their privacy cannot be guaranteed.

In this dissertation, we introduce novel solutions to a number of challenging security and privacy issues in heterogeneous mobile and wireless networks to pave the way for their wide adoption and deployment. These challenging issues are either newly identified or not well addressed in the literature. The rest of this dissertation is structured as follows.

Chapter 2 tackles secure and loss-resilient code dissemination in wireless sensor networks. Code dissemination refers to the process of disseminating a new code image via wireless links to all sensor nodes after they are deployed. It is desirable and often necessary due to the need for, e.g., removing program bugs and adding new functionalities in a multi-task sensor network. A sound code dissemination scheme need be both loss-resilient and attack-resilient, which are crucial for sensor networks deployed in lossy and hostile environments. We propose LR-Seluge, the first loss-resilient and secure code dissemination scheme based on a novel integration of fixed-rate erasure code and efficient cryptographic primitive. The efficacy and efficiency of LR-Seluge are confirmed by both theoretical analysis and extensive simulation results. In particular, LR-Seluge can reduce up to 40% communication overhead in lossy environments with the same level of attack resilience in contrast to existing schemes.

Chapter 3 considers verifiable location-based top-k query processing against untrusted location-based service provider. In view of the significant drawbacks of existing location-based query services, we propose a novel architecture for collaborative location-based information collection and sharing, which relies on some trusted data collectors acting as the central hubs using various incentives to collect point-of-interest (POI) reviews from consumers and then sell the aggregated POI data set to individual location-based service providers (LBSPs). We then develop effective and efficient solutions to enable location-based service users to verify the authenticity and correctness of the query results from the untrusted LBSP.

Chapter 4 investigates verifiable privacy-preserving aggregation in people-centric mobile sensing systems (PC-MSSs). People-centric mobile sensing systems refer to

using human-carried mobile devices such as smartphones and tablets with ever-growing capabilities in sensing, computation, storage, and communications for urbanscale distributed data collection, analysis, and sharing. A main obstacle for the widespread adoption of PC-MSSs is the privacy concern of participating individuals as well as the concern about data integrity. To tackle this open challenge, we propose VPA, a novel solution to verifiable privacy-preserving data aggregation in PC-MSSs. VPA can support a wide range of statistical additive and non-additive aggregation functions such as Sum, Average, Variance, Count, Max/Min, Median, Histogram, and Percentile with accurate aggregation results.

Chapter 5 addresses privacy-preserving profile matching in proximity-based social networking (PMSN). PMSN refers to adjacent mobile users interacting through the Bluetooth/WiFi interfaces on their mobile devices. The first step toward effective PMSN is for two users to compare their profiles, known as profile matching, which is nevertheless hindered by the growing privacy concerns about disclosing sensitive personal profiles to strangers. To tackle this open challenge, we designed a suite of novel protocols to enable two users to perform profile matching without disclosing any information about their profiles beyond the comparison result. In contrast to existing coarse-grained private matching schemes for PMSN, our protocols allow finer differentiation between PMSN users and can support a wide range of matching metrics at different privacy levels.

Chapter 6 studies secure combination of spectrum-sensing results in crowdsourcing-based cooperative (spectrum) sensing systems. Cooperative sensing is a key function for dynamic spectrum access and is essential for avoiding interference with licensed primary users and identifying spectrum holes. A promising approach for effective cooperative sensing over a large geographic region is to rely on special spectrum-sensing providers (SSPs), which outsource spectrum-sensing tasks to distributed mobile users. Crowdsourcing-based cooperative spectrum sensing is, however, vulnerable to malicious sensing data injection attack, in which a malicious cognitive radio users

submit false sensing reports containing power measurements much larger (or smaller) than the true value to inflate (or deflate) the final average, in which case the SSP may falsely determine that the channel is busy (or vacant). To tackle this challenge, we propose a novel scheme to enable secure crowdsourcingbased cooperative spectrum sensing by jointly considering the instantaneous trustworthiness of mobile detectors in combination with their reputation scores during data fusion. Our scheme can enable robust cooperative sensing even if the malicious CR users are the majority.

We summarize our work along with a discussion on future directions in Chapter 7.

Chapter 2

LOSS-RESILIENT AND SECURE CODE DISSEMINATION IN WIRELESS SENSOR NETWORKS

2.1 Introduction

Code dissemination [47] or *over-the-air reprogramming* [43] in wireless sensor networks refers to the process of disseminating a new code image via wireless links to all sensor nodes after they are deployed. It is desirable and often necessary due to the need for, e.g., removing program bugs and adding new functionalities in a multi-task sensor network [104].

A sound code dissemination scheme faces two critical challenges. First, wireless channels are lossy in nature, especially for sensor networks deployed in remote and harsh environments. Packets may get lost during transmission due to many reasons such as RF interference and environmental factors [115]. This calls for loss-resilient code dissemination schemes like [43,46,92] that can withstand high packet losses. Second, sensor networks in hostile environments such as the battlefield may be attacked. In particular, the adversary may exploit the code dissemination mechanism to launch various attacks. For example, the adversary may inject bogus code images to take over the control of the whole sensor network or launch the Denial-of-Service (DoS) attack by transmitting many bogus image packets to deplete the limited energy and/or buffer of sensor nodes. This situation necessitates secure code dissemination schemes such as Seluge [49] which provides code-image integrity and DoS attack resilience through immediate and efficient packet authentication.

To the best of our knowledge, no existing code dissemination scheme satisfies loss resilience and attack resilience at the same time. On the one hand, all existing secure code dissemination schemes such as [25, 49, 60, 105, 106, 110] are secure versions of Deluge [47], the de facto non-secure code dissemination scheme. Deluge [47] relies on Automatic Repeat Request (ARQ) protocols for reliable broadcast transmissions, in which each local broadcast receiver uses NACKs to request retransmission

of lost packets. ARQ protocols, however, are generally not suitable for broadcasting in highly lossy networks due to too many retransmissions and the accompanying high latency [115]. On the other hand, most loss-resilient code dissemination schemes employ rateless erasure codes such as Fountain codes [92] and random linear codes [43] at the sender side to cope with packet losses. The common idea is to encode the code image into an unlimited number of packets such that each local receiver can recover the code image after receiving sufficiently many packets. It is unfortunately infeasible to use the similar methods as in Seluge [49] to realize immediate and efficient authentication of potentially unlimited erasure-coded packets. There is a clear gap between these two lines of research.

In this chapter, we fill the this gap by LR-Seluge, a novel loss-resilient and secure code dissemination scheme for sensor networks deployed in hostile and lossy environments. As the first of its kind, LR-Seluge is aimed to strike a good balance between loss resilience and immediate packet authentication by using a limited number of predetermined redundant packets. More specifically, we encode the code image using a fixed-rate erasure code and carefully create chaining relationships between original and encoded packets using lightweight cryptographic hash functions. This design allows sensor nodes not only to recover the original code image from a subset of the encoded packets but also to efficiently authenticate any encoded packet upon its arrival, thus simultaneously achieving sufficiently high loss resilience and attack resilience.

Our main contributions in this chapter are summarized as follows.

- We notice the lack of a sound code dissemination scheme for sensor networks in lossy and hostile environments with satisfactory loss resilience and attack resilience and fill this void by proposing LR-Seluge.
- We theoretically analyze and compare the performance of LR-Seluge with and Seluge [49], a representative secure code dissemination scheme that ensures code-image integrity and provides very strong resilience to DoS attacks. We fur-

ther confirm the efficacy and efficiency of LR-Seluge by extensive simulation results. Our results reveal that LR-Seluge can save up to 40% in communication overhead and 40% in code-dissemination latency with the same level of attack resilience in comparison with Seluge.

The rest of the chapter is structured as follows. We introduce our network and adversary models and the design goal in Section 2.3, followed by a brief discussion on the background Section 2.2. We then present the design of LR-Seluge in Section 2.4. Subsequently, we analyze the performance of LR-Seluge in Section 2.5 and evaluate LR-Seluge using extensive simulation results in Section 2.6. We discuss the related work in Section 2.7 and finally concludes this chapter in Section 2.8.

2.2 Background

This section introduces the background knowledge necessary for understanding the design of LR-Seluge.

2.2.1 Deluge

Deluge [47] is the de facto code dissemination paradigm for sensor network, which is also one of the standard components of the TinyOS distribution [2].

Deluge employs a page-by-page dissemination strategy, in which a large code image is divided into smaller fixed-size pages, and each page is further divided into same-size packets. A sensor node requests for a new page only after completely receiving all the packets of the previous page. During the code dissemination process, each node works in one of the following three states: *maintenance*, in which node periodically advertises the version of its code image and the number of pages it has for that version, *receiving*, in which the node actively requests the remaining packets to complete the current page using Selective NACK (SNACK for short) requests, and *transmitting*, in which the node broadcasts all the requested packets of the current page and continuously serves any subsequent request.

Deluge employs various suppression techniques to maximize the effect of overhearing and reduce packet collisions. For example, each node suppresses its own advertisement after overhearing a threshold number of advertisements with the same information. Moreover, a node suppresses its own request (or data) packet if overhearing request (or data) packets for a page with the same or smaller indices. We refer readers to [47] for the details about these suppression techniques.

2.2.2 Seluge

Seluge [49] is an exemplary secure code dissemination scheme, which ensures code-image integrity and DoS attack resilience by enabling efficient and immediate packet authentication.

Seluge integrates three different techniques to realize efficient and immediate packet authentication. First, Seluge chains the packets of two adjacent pages with a cryptographic hash function. For example, the hash image of the jth packet of page i is embedded into the jth packet of page $i-1$. Since Seluge adopts the same page-by-page dissemination strategy from Deluge, any packet of page i can be authenticated upon arrival because all the packets of page $i-1$ must have been received. Second, to authenticate the first page, Seluge introduces a special *hash page* formed by concatenating all the hash images of packets of the first page. A Merkle hash tree is built on top of the hash page, and the base station digitally signs the root of the Merkle hash tree to ensure its integrity. In this way, each packet of the hash page can be authenticated by computing the hash images along the path to the tree root. Finally, to prevent the adversary from transmitting a large number of signature packets to force sensor nodes to perform computationally expensive signature verifications, Seluge let the base station attach a weak authenticator to the signature packet, which is a *message specific puzzle* [81]. Only if the weak authenticator is valid do sensor nodes verify the signature. Seluge does not work well in lossy environments due to its dependence on Deluge, as to be shown later.

2.2.3 Erasure Code

A k-n-k' erasure code transforms k packets into $n \geq k$ encoded packets of the same length such that the original k packets can be recovered from any k' encoded packets. The fraction k/n is called the *code rate*, and the fraction k'/k is called *reception efficiency*, where k' denotes the number of packets required for recovery. When $k' = k$, the erasure code is *optimal*. Typical erasure codes include Reed-Solomon codes, Tornado codes, Raptor codes, Online codes, and LT codes, for which a good survey can be found in [75].

2.3 Network/Adversary Models and Design Goals

This section outlines the network and adversary models underlying LR-Seluge as well as our design goals.

2.3.1 Network Model

As in Seluge [49], we consider a WSN consisting of a base station and many sensor nodes densely deployed in hostile environments such as the battlefield. The base station has a new code image $\mathcal{M}$ to be disseminated to all the sensor nodes via wireless links. We assume that the base station has abundant resources in computation and communication and is safeguarded from attacks. We also assume that the base station has a public/private key pair. In contrast, sensor nodes are more constrained in storage, energy, computation, and communication capabilities. We, however, adopt the same assumption in [49] that a sensor node can verify a few digital signatures, e.g., one signature verification per code image. It is worth noting that technical advances have rendered it quite feasible to execute once-daunting public-key operations on sensor nodes. For example, it takes 1.12s for a Tmote Sky mote to verify an ECDSA signature [113]. Such public-key operations will be minimized in LR-Seluge, as to be shown later. Moreover, we assume lossy and unreliable wireless channels such that packets may get lost due to many reasons such as environmental factors and accidental or malicious RF interference [115].

2.3.2 Adversary Model

We also adopt the adversary model from [49]. In particular, we consider a computationally bounded adversary consisting of both *external* and *internal* attackers. External attackers do not belong to the target WSN, but they are capable of overhearing packet transmissions, injecting bogus packets, and replaying intercepted packets. Internal attackers are compromised yet undetected sensor nodes which are fully controlled by the adversary. We, however, follow the conventional assumption that non-compromised sensor nodes are always the majority.

Despite the many attacks the adversary can launch, this chapter has the same target as Seluge [49] and focuses on defeating the following two attacks on code dissemination.

- The adversary may inject forged code images to take control of the sensor network.
- The adversary may send many fake packets or replay intercepted packets to force sensor nodes into wasteful packet processing so as to quickly deplete their limited energy and/or memory buffer.

2.3.3 Design Objectives

In view of the two attacks outlined above, LR-Seluge is designed with the following goals in mind.

- *Code-image integrity*: Every sensor node is guaranteed to receive an authentic code image unless the node is isolated from the rest of the network.
- *DoS attack resilience*: Any forged packets should be immediately detected upon their arrivals, and packet authentication should be efficient.
- *Loss resilience*: LR-Seluge should enable more efficient code dissemination in the presence of severe packet losses than prior work such as Seluge [49].

2.4 LR-Seluge Design

In this section, we first give an overview of LR-Seluge and then present its design in detail.

2.4.1 Overview of LR-Seluge

LR-Seluge is largely inspired by two observations. First, loss-resilient broadcasting can be achieved by using erasure codes at the sender side, i.e., introducing redundant packets, as demonstrated in [43, 46, 92, 115]. Second, immediate and efficient packet authentication can be realized if all the packets to be transmitted are predetermined. Consider Seluge [49] as an example, in which all the image packets are preprocessed to create some chaining relationships with cryptographic hash functions to enable immediate and efficient packet authentication. Such preprocessing can be done only if all the packets can be determined prior to transmission.

The key idea of LR-Seluge is to introduce a limited number of predetermined redundant packets to increase loss resilience and also achieve immediate packet authentication by carefully creating chaining relationships between encoded packets and original packets. More specifically, LR-Seluge differs from existing loss-resilient code dissemination schemes [43, 46, 92, 115] in that it uses a fixed-rate erasure code instead of rateless ones to encode the image packets. To enable immediate packet authentication, LR-Seluge creates some chaining relationships between the original packets of one page and the encoded packets of the next page. In this way, once a node receives sufficient encoded packets to recover one page, it also recovers all the hash images of the next page at the same time. This design differs from Seluge [49] in that there is no one-to-one correspondence between the packets of adjacent pages. Furthermore, LR-Seluge employs a simple but effective scheduling algorithm that allows each sender to transmit much fewer packets to ensure that every one-hop neighbor receives sufficient encoded packets to recover a code page.

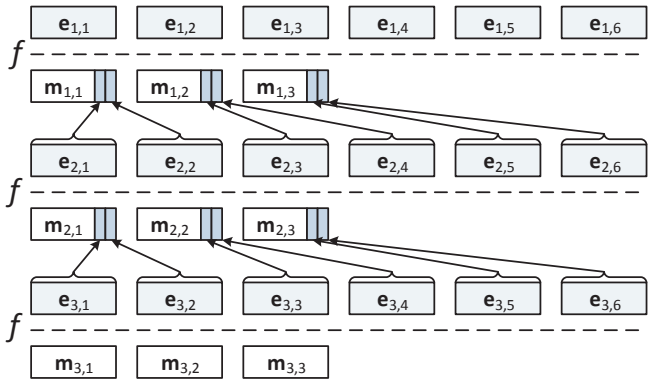


Figure 2.1: An example of code-image preprocessing for pages $\mathcal{M}_1$ to $\mathcal{M}_g$, where $g = 3, k = 3, n = 6$.

In what follows, we detail the operations of LR-Seluge, including *initialization*, *code-image preprocessing*, *efficient loss-resilient code dissemination*, and *authenticated packet transmission*.

2.4.2 Initialization

Before the network deployment, the network owner preloads each sensor node with the following information.

- The same instance of a k-n-k' erasure code $f(\cdot)$;
- The same instance of a k_0-n_0-k'_0 erasure code $f_0(\cdot)$;
- The public key of the base station;
- A public cryptographic hash function $H(\cdot)$.

With $f(\cdot)$ or $f_0(\cdot)$, every node can generate the same n or n_0 encoded packets from the same k or k_0 input packets.

2.4.3 Code-Image Preprocessing

Assume that the base station has a code image $\mathcal{M}$ for dissemination to all sensor nodes. As in [47, 105], the base station partitions the original image $\mathcal{M}$ into g pages of fixed size, denoted by $\{\mathcal{M}_i\}_{i=1}^{g}$. Each page $\mathcal{M}_i$ is then further divided into k original

blocks of equal length, i.e., $\mathcal{M}_i = \{\mathbf{m}_{i,j}\}_{j=1}^{k}$, $\forall i \in [1, g]$. We will use the terms "Page i" and "Page $\mathcal{M}_i$" interchangeably hereafter. Starting from page $\mathcal{M}_g$, the base station constructs the packets for each page in the reverse order as the pages are transmitted. An example is given in Fig. 2.1, where $g = 3, k = 3, n = 6$.

2.4.3.1 Page g

For page $\mathcal{M}_g$, the base station applies f on $\{\mathbf{m}_{g,j}\}_{j=1}^{k}$ to generate n encoded blocks as

$$f(\mathbf{m}_{g,1}, \cdots, \mathbf{m}_{g,k}) = (\mathbf{e}_{g,1}, \cdots, \mathbf{e}_{g,n}) \ . \tag{2.1}$$

The n packets of page $\mathcal{M}_g$ are then constructed as $P_{g,j} := \langle \nu, g, j, \mathbf{e}_{g,j} \rangle, \forall j \in [1, n]$, where ν and g denote the code version and the page number, respectively.

2.4.3.2 Pages $g-1$ to 1

After constructing $\{P_{g,j}\}_{j=1}^{n}$, the base station proceeds to construct $\{P_{g-1,j}\}_{j=1}^{n}$ for page $\mathcal{M}_{g-1}$. The basic idea is to append the hash images of $\{P_{g,j}\}_{j=1}^{n}$ to the original k blocks of page $\mathcal{M}_{g-1}$ and then apply f to generate the n packets of $\mathcal{M}_{g-1}$. In particular, the base station computes $h_{g,j} = H(P_{g,j})$, $\forall j \in [1, n]$, and then splits $h_{g,1}|| \cdots ||h_{g,n}$ into k slices of equal length, denoted by $\{\mathbf{h}_{g,j}\}_{j=1}^{k}$. The n encoded blocks of page $\mathcal{M}_{g-1}$ are then generated as

$$f(\mathbf{m}'_{g-1,j}, \cdots, \mathbf{m}'_{g-1,k}) = (\mathbf{e}_{g-1,1}, \cdots, \mathbf{e}_{g-1,n}) \ , \tag{2.2}$$

where $\mathbf{m}'_{g-1,j} := \mathbf{m}_{g-1,j}||\mathbf{h}_{l,j}$. The n packets of page $\mathcal{M}_{g-1}$ to be transmitted are finally generated as $P_{g-1,j} := \langle \nu, g-1, j, \mathbf{e}_{g-1,j} \rangle$, $\forall j \in [1, n]$.

By repeating the above process, the base station can also iteratively construct the packets for pages $g-2$ to 1, i.e., $\{P_{i,j}\}_{j=1}^{n}, \forall i \in [1, g-2]$.

2.4.3.3 Page 0 and Signature Packet

We use a similar approach as in Seluge [49] to authenticate page $\mathcal{M}_1$ by purposefully introducing a hash page $\mathcal{M}_0$ as the concatenation of the n hash images of page $\mathcal{M}_1$, i.e., $\mathcal{M}_0 := h_{1,1}|| \cdots ||h_{1,n}$. In general, $\mathcal{M}_0$ is much shorter than other pages but still cannot fit into one packet. Therefore, the base station first splits $\mathcal{M}_0$ into k_0 blocks of

equal length, denoted by $\{\mathbf{m}_{0,i}\}_{i=1}^{k_0}$, and then applies the erasure code f_0 to generate n_0 encoded blocks as follows,

$$f_0(\mathbf{m}_{0,1}, \cdots, \mathbf{m}_{0,k_0}) = (\mathbf{e}_{0,1}, \cdots, \mathbf{e}_{0,n_0}) \ , \tag{2.3}$$

where $n_0 = 2^d$ for some integer d.

The base station then builds a Merkle hash tree [71] of depth d on top of $\{\mathbf{e}_{0,j}\}_{j=1}^{n_0}$, as illustrated in Fig. 2.2. In particular, the base station computes $\mathbf{v}_j = H(\mathbf{e}_{0,j})$, $j \in [1, n_0]$, and builds a binary tree by computing each internal node as the hash of its adjacent children nodes. For example, $\mathbf{v}_{3-4} = H(\mathbf{v}_3||\mathbf{v}_4)$ and $\mathbf{v}_{1-4} = H(\mathbf{v}_{1-2}||\mathbf{v}_{3-4})$ in Fig. 2.2.

Given the Merkle hash tree, the base station constructs one packet for each $\mathbf{e}_{0,j}$, which consists of $\mathbf{e}_{0,j}$ itself and its authentication information, i.e., all the siblings of the nodes in the path from $\mathbf{v}_{0,j}$ to the root of the Merkle hash tree. For example, $P_{0,2} := \langle \nu, 0, 2, \mathbf{e}_{0,2}, \mathbf{v}_1, \mathbf{v}_{3-4}, \mathbf{v}_{5-8} \rangle$ in Fig. 2.2.

The whole code image is authenticated by the base station signing the root of the Merkle hash tree using its private key. To mitigate the possible DoS attack in which the adversary injects many signature packets to deceive sensor nodes into continuous relatively expensive signature verifications, a weak authenticator like a message specific puzzle in Seluge [49] can be attached to the signature packet as a defense.

2.4.4 Efficient Loss-Resilient Code Dissemination

This subsection details the code dissemination process of LR-Seluge which efficiently copes with packet losses. To ease the illustration, we defer the illustration of LR-Seluge's packet authentication mechanisms to Section 2.4.5. As in Deluge and Seluge, every node in LR-Seluge works in one of three states at any time: MAINTAIN, RX and TX. LR-Seluge differs from Deluge and Seluge mainly in the TX state due to the use of erasure codes. For self-containment, below we briefly discuss the operations in the MAINTAIN and RX states and then detail the operations in the TX state.

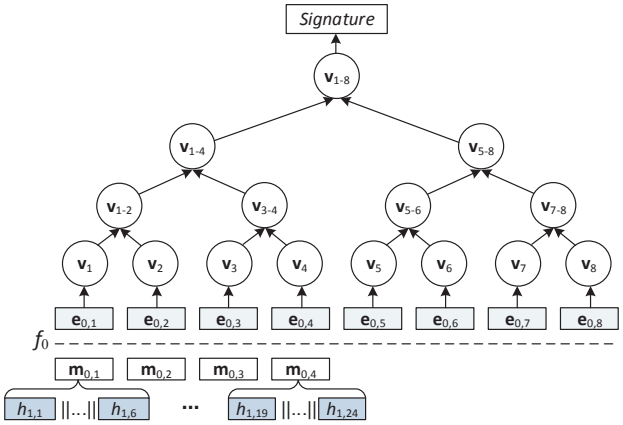


Figure 2.2: An example of construction Merkle hash tree over page $\mathcal{M}_0$, where $n = 24, k_0 = 4, n_0 = 8$.

2.4.4.1 MAINTAIN

Every node in the MAINTAIN state monitors all its one-hop neighbors to ensure that they all possess the same number of pages of the latest code image. For this purpose, every node periodically broadcasts advertisements, each consisting of the sender ID, the code version number, and the largest page number in possession (which implies all previous pages are also in possession). Here a page is said to be possessed if the sender has received at least k' or k'_0 out of n of n_0 encoded packets of the page and successfully decoded it.

If a node detects that any neighboring node has either a newer code image or more pages of the same code image, it requests the missing pages from that neighbor with an SNACK request. For example, assume that node v overhears an advertisement from node u indicating that node u has more pages, node v switches to the RX state and begins requesting the missing pages from node u which will switch to the TX state upon an SNACK request from node v. In addition, to enable fast code propagation while limiting the number of advertisement packets, every node adjusts the advertisement frequency using Trickle [63], a protocol for maintaining code updates in WSNs.

Node	$P_{i,1}$	$P_{i,2}$	$P_{i,3}$	$P_{i,4}$	$\mathbf{d}$
v_1	0	1	0	1	1
v_2	1	1	0	1	2
v_3	1	1	1	1	3
Pop.	2	3	1	3	

(a) Node u's tracking table at some time T

Node	$P_{i,1}$	$P_{i,2}$	$P_{i,3}$	$P_{i,4}$	$\mathbf{d}$
v_2	1	0	0	1	1
v_3	1	0	1	1	2
Pop.	2	0	1	2	

(b) After transmitting packet $P_{i,2}$

Node	$P_{i,1}$	$P_{i,2}$	$P_{i,3}$	$P_{i,4}$	$\mathbf{d}$
v_3	1	0	1	0	1
Pop.	1	0	1	0	

(c) After transmitting packet $P_{i,4}$

Table 2.1: An example of node operation in *TX* state, where $k = k_0 = 3, n = 4$.

2.4.4.2 RX

A node in the RX state keeps sending SNACK requests to corresponding neighboring nodes which possess a missing page until receiving enough packets to decode the missing page. A SNACK request includes a bit-vector of n bits with every bit indicating whether the corresponding packet is desired. For every received packet, the requesting node first authenticates it using the methods in Section 2.4.5 and stores only the packet passing the authentication. Once k' or k'_0 out of n or n_0 authenticated packets are received, the requesting node can erasure-decode the missing page and then returns to the MAINTAIN state.

2.4.4.3 TX

A node, say u, switches to the TX state after receiving an SNACK request for a page it has. The operations of LR-Seluge in the TX state differs from Deluge and Seluge mainly in the following two aspects.

First, to serve an SNACK request, a node need erasure-encode the requested page with the hash images of the next page's packets. Consider node u as an example. Assume that u has received at least k' authenticated packets of page $\mathcal{M}_i$ and erasure-decoded them to recover $\mathcal{M}_i$ and the appended hash images of page $\mathcal{M}_{i+1}$'s encoded packets, i.e., $h_{i+1,1}||\cdots||h_{i+1,n}$. Suppose that u receives an SNACK request from node v requesting $\mathcal{M}_i$. As the base station does in Section 2.4.3, node u applies the same erasure code f to $\mathcal{M}_i$ appended by $h_{i+1,1}||\cdots||h_{i+1,n}$ and obtains the n encoded packets. Node u can then broadcast the encoded packets corresponding to the bit vector in the SNACK request. Since node v has received page $\mathcal{M}_{i-1}$ and thus the hash images of $\mathcal{M}_i$'s n encoded packets, it can immediately authenticate the packets from u using the method in Section 2.4.5.

Second, a suitable scheduling algorithm is needed for nodes in the TX state to reduce the number of transmissions for reducing communication overhead. In particular, different packets of the same page may be needed by different neighbors of the node having that page due to random packet losses. It is thus desirable for the sender to transmit the smallest subset of the n encoded packets to simultaneously satisfy the requests from all its neighbors. This scheduling requirement is not found in existing code dissemination schemes. In particular, a node in Deluge and Seluge simply transmits packets corresponding to the union of bit vectors in SNACK packets, and a node in the schemes [43, 46, 92] based on rateless erasure codes always transmits a fresh encoded packet for an SNACK request.

We propose an effective *greedy round-robin* scheduling algorithm for nodes in the TX state. The basic idea is for a node to transmit the packet desired by the highest number of neighbors in a round-robin fashion. More specifically, every node in the TX state, say node u, maintains a so-called *tracking table* with every table entry corresponding to one neighbor from which an SNACK was received. The tracking entry for a node, say v, consists of the following fields.

- The node ID v;

- A bit vector of length n indicating which packets have been received by v from u's current point of view;
- The *distance* of node v, denoted by $\mathbf{d}_v$ and referring to the number of additional packets v needs to recover the requested page.

Initially, the tracking table is empty. Upon receiving an SNACK request from a node, say v, node u first checks if there is an entry for node v. If not, node u creates an entry for node v, copies the bit vector from the SNACK request, and sets the distance $\mathbf{d}_v$ as the additional number of packets needed by v. For example, if all bits in the bit vector of the SNACK request are ones, then $\mathbf{d}_v = k'$. In general, since node v needs at most k' packets for decoding the requested page, we have $\mathbf{d}_v = q + k' - n$, where q is the number of ones in the bit vector. If there is an entry for node v, node u updates the entry according to the SNACK request.

To illustrate the scheduling algorithm, assume that node u is handling the S-NACK requests from its neighbors for page $\mathcal{M}_i$ which is erasure-encoded into packets $\{P_{i,j}\}_{j=1}^{n}$. We also define the *popularity* of a packet as the number of nodes requesting it. Also assume that there are z entries in node u's tracking table. The z bit vectors form a $z \times n$ bitmap, in which the total number of ones in the jth column indicates the popularity of packet $P_{i,j}$. The first packet, say $P_{i,x}$, sent by u is the packet with the highest popularity and also the lowest packet index in case that there are multiple packets of the highest popularity. After sending $P_{i,x}$, node u updates the tracking table by setting all the bits in the xth column to zero and decreasing the distances of the nodes needing $P_{i,x}$ by one. Note that if $P_{i,x}$ failed to reach some nodes, these nodes may request it again in a later SNACK packet. If some nodes' distances reach zero, their entries are deleted. The next packet is selected as the one with the highest popularity and the index equal to $\min\{x+1, \cdots, n, n+1, \cdots, n+x-1\} \mod n$, i.e., the first packet to the right of $P_{i,x}$ with the highest popularity. This process continues until u's tracking table is empty.

Table 2.1 gives an example where $k = k' = 3, n = 4$. Assume that at some point, node u's tracking table has three entries for nodes v_1, v_2 and v_3, as shown in Table 2.1a. The popularity of packets $P_{i,1}$ to $P_{i,4}$ are $2, 3, 1,$ and 3, respectively. Node u will first transmit $P_{i,2}$ and then updates the tracking table to Table 2.1b. Note that node v_1 has been removed from the tracking table because its distance reaches zero. Subsequently, node u chooses $P_{i,4}$ to transmit because it is the first packet of the highest popularity on $P_{i,2}$'s right side. After $P_{i,4}$ is transmitted, the tracking table is updated again to Table 2.1c. Finally, packet $P_{i,1}$ is transmitted, after which the tracking table becomes empty.

2.4.5 Authenticated Code Dissemination

This subsection details how LR-Seluge realizes authenticated code dissemination. LR-Seluge adopts the page-by-page dissemination approach from Deluge and Seluge in which a node can only request a new page if all previous pages have been completely received and recovered. This page-by-page strategy together with LR-Seluge's packet construction enables immediate packet authentication to ensure code-image integrity and also prevents the DoS attack that targets exhausting the receivers' energy or buffers.

In particular, the base station initiates the dissemination process by broadcasting the signature packet. On receiving the signature packet, every sensor node, say u, verifies the signature to authenticate the root of the Merkle hash tree, e.g., $\mathbf{v}_{1-8}$ in Fig. 2.2. If the verification succeeds, node u begins to send SNACK packets requesting the packets of page $\mathcal{M}_0$. Every packet in page $\mathcal{M}_0$ can be immediately authenticated upon its arrival. For example, for packet $P_{0,1}$ in Fig. 2.2, node u can verify its authenticity by checking if the following equation holds,

$$\mathbf{v}_{1-8} = H(H(H(H(\mathbf{e}_{0,1})||\mathbf{v}_2)||\mathbf{v}_{3-4})||\mathbf{v}_{5-8}) \ .$$

If so, it stores the packet and otherwise drops it.

Once node u has collected k_0' authenticated packets of page $\mathcal{M}_0$, say $\{P_{0,j_x}\}_{x=1}^{k_0'}$, it can erasure-decode $\mathcal{M}_0$ as

$$f_0^{-1}(\mathbf{e}_{0,j_1}, \cdots, \mathbf{e}_{0,j_{k_0'}}) = (\mathbf{m}_{0,1}, \cdots, \mathbf{m}_{0,k_0}) \ . \tag{2.4}$$

Recall that $\mathcal{M}_0$ contains all the hash images of the packets of page $\mathcal{M}_1$. Therefore, node u can subsequently authenticate all the packets of page $\mathcal{M}_1$ upon their arrivals by a simple hash verification. Similarly, once at least k' authenticated packets of $\mathcal{M}_1$ have been collected, node u can decode $\mathcal{M}_1$ to get all the hash images of the packets of page $\mathcal{M}_2$ whereby to immediately authenticate all the packets of page $\mathcal{M}_3$. In short, the page-by-page strategy guarantees that whenever node u requests a new page from a neighboring node, all the information needed for authenticating the new page is available at that time. Therefore, any data packet can be immediately authenticated upon their arrivals.

In addition, LR-Seluge adopts the same mechanisms in Seluge, i.e., cluster key and message specific puzzle, to authenticate advertisement and SNACK packets and to effectively filter out forged signatures of the root of the Merkle hash tree, respectively. Therefore, LR-Seluge inherits the same level of resilience to DoS attacks that exploit Deluge's epidemic propagation and suppression mechanisms.

It is worth noticing that LR-Seluge and all existing secure code dissemination schemes [25, 49, 60, 105, 106, 110] are vulnerable to a special kind of *denial of receipt* attack in which a compromised sensor node denies it has received any data packets but keeps sending SNACK packets to a victim node in order to deplete its energy. In particular, a victim node need transmit k' data packets on receiving a SNACK packet with all bits set to one. It is fundamentally difficult to verify whether a particular packet has been received by certain nodes in lossy environment.

To mitigate the impact of this attack, a possible solution is to replace cluster key by a local authentication scheme like LEAP [140] to simultaneously authenticate and identify the source of any SNACK packet. In addition, each node in TX state maintains

a counter for the number of SNACK packets from each neighbor. For any page, if the number of data packets requested by a neighboring node, say v, exceeds some certain threshold, the node that serving the page, say u, can simply ignore future SNACK packets from v, under the assumption that either v is launching the denial of receipt attack or the channel between u and v is too bad so that v should request data packets from its other neighbors.

2.5 Performance Analysis

Section 2.4.5 discusses how LR-Seluge realizes code-image integrity and DoS resilience. In this section, we analyze the communication and computation overhead of LR-Seluge.

2.5.1 Communication Overhead

The communication costs of Seluge and LR-Seluge both comprise the transmissions of data, advertisement, and SNACK packets, among which data-packet transmissions account for the most. Both costs are very difficult to analyze in a general multi-hop sensor network with arbitrary topologies and random packet losses. To enable theoretical tractability, we here analyze the number of data-packet transmissions under Seluge and LR-Seluge, respectively, under a one-hop scenario. This is a meaningful simplification, as the performance of Seluge and LR-Seluge largely depends on hop-by-hop local broadcasting. The impact of advertisement and SNACK packets and also the communication costs of Seluge and LR-Seluge in multi-hop scenarios will be demonstrated using simulations in Section 2.6.

We consider a one-hop scenario consisting of N receivers and a local sender at the center. Our main goal is to demonstrate the impact of employing erasure codes in lossy environments. So we adopt a similar model as in [80] in which every packet to node i gets lost with probability p_i and packet losses at different nodes are uncorrelated.

We then have the following theorems regarding the number of data packet transmissions in Seluge and LR-Seluge, respectively, whose proofs are available in our technical report [135]. It is worth mentioning that the result in Theorem 2.5.2 is obtained by

analyzing a variation of LR-Seluge. In this variation, instead of using SNACK packets, each receiver returns an ACK packet only after receiving k' packets of the current page. Since the local sender has no information about which packets are missing at each node, it has to repeatedly transmit the n erasure-coded packets until receiving an ACK from every neighboring node. This variation is apparently less efficient than LR-Seluge, as the sender may unnecessarily transmit some packets which have reached all the receivers. Therefore, its communication cost can be viewed as the upper bound of LR-Seluge

Theorem 2.5.1. With Seluge, the expected number of data-packet transmissions needed to transmit a page of k packets to all N nodes is given by

$$k \sum_{t=1}^{\infty} t \cdot \big(\prod_{i=1}^{N} (1 - p_i^t) - \prod_{i=1}^{N} (1 - p_i^{t-1}) \big) \ . \tag{2.5}$$

The proof is given in Appendix A.

Theorem 2.5.2. With LR-Seluge employing a k-n-k' erasure code, the expected number of data-packet transmissions needed for all N nodes to receive at least k' packets to recover the original page of k packets is bounded by

$$n \sum_{r=1}^{\infty} r \cdot (Pr(\mathbf{R} \leq r) - Pr(\mathbf{R} \leq r - 1)) \ , \tag{2.6}$$

where $Pr(\mathbf{R} \leq r) = \prod_{i=1}^{N} \sum_{j=k'}^{n} \binom{n}{j} (1 - p_i^r)^j p_i^{r(n-j)}$.

The proof is given in Appendix B.

2.5.2 Computation Overhead

The computation overhead of LR-Seluge comes from packet authentication and page encoding/decoding.

The computation cost incurred by packet authentication is very similar to that of Seluge. First, authenticating a signature packet requires one hash function for the weak authenticator and one signature verification. Second, authenticating a packet of page $\mathcal{M}_0$ requires $d + 1$ hash computations, and total $k_0'(d + 1)$ hash computations

are needed for page $\mathcal{M}_0$. In contrast, authenticating a packet of page $\mathcal{M}_i, 1 \leq i \leq g$ requires one hash computation, so total k' hash computations are needed for each page $\mathcal{M}_i$.

The computation cost incurred by erasure decoding and encoding depends on the particular erasure code used by LR-Seluge. Theoretically speaking, LR-Seluge can be integrated with any erasure code as long as the parameters (k, n, k') are satisfied. In our evaluation, we adopt a fixed-rate LT code from SYNAPSE [92], in which each encoded packet is the XOR of some original packets. Assuming that $k = 32$, $n = 64$, and $k' = 35$, decoding a page of 32 original packets of 25 bytes requires 5142 XOR operations on average and takes about 462 ms on a Tmote Sky sensor node [92]. Same as Seluge [49], LR-Seluge uses data packets with an effective payload length of 96 bytes. Since the decoding cost of the LT code is linear to the packet length, decoding a page of 32 original packets of 96 bytes requires approximately 19745 XOR operations and takes about 1.8 seconds. Finally, nodes in the TX state need erasure-encode the original page to obtain the other $n - k'$ missing packets, which requires $s(n - k') * 96$ XOR operations on average, where s is the average number of original blocks XORed to generate an encoded packet. If $s = 12.06$ as in [92], the encoding cost for a page in LR-Seluge requires approximately 33576 XOR operations and takes about 3.1 seconds on a Tmote Sky sensor node. It is also worth noting that in LR-Seluge, only a few nodes in the TX state need perform encoding.

Although LR-Seluge introduces additional decoding and encoding delays at sensor nodes in comparison with Seluge, it greatly reduces the overall code-dissemination latency due to the dramatic reduction in the number of packet transmissions. This argument will be validated using simulations in the next section.

2.6 Performance Evaluation

In this section, we compare LR-Seluge with Seluge using extensive simulations in TOSSIM [62], a discrete event simulator distributed with TinyOS 2.1.1.

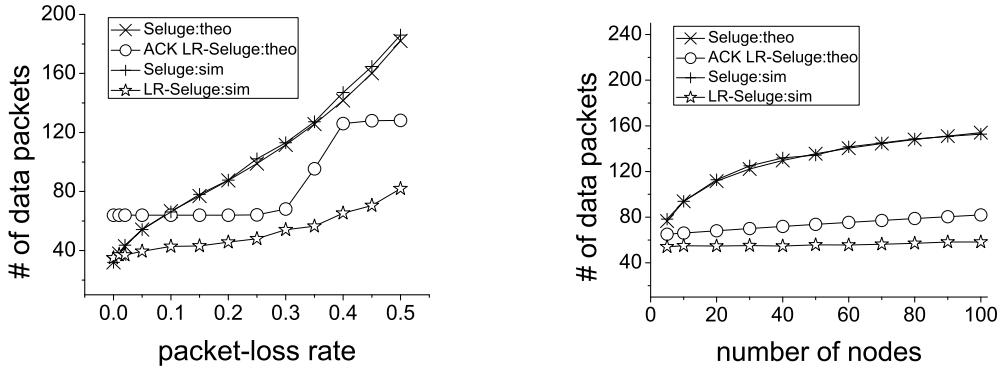
Unless stated otherwise, the following simulation configurations are used. For the k-n-k' erasure code $f(\cdot)$, we have $k = 32$, $n = 64$, and $k' = 35$; for the k_0-n_0-k'_0 erasure code $f_0(\cdot)$, we have $k_0 = 8$, $n_0 = 16$, and $k'_0 = 11$. As in Seluge [49], we set the packet-payload size to 102 bytes (the maximum payload size in the IEEE 802.15.4 standard) and use the 64-bit truncation of SHA-1 as the hash function $H(\cdot)$, and use a gap of 17 ms between two data-packet transmissions. The image version number, page number, and packet number in both Seluge and LR-Seluge packets totally consume 6 bytes, which leaves 96 bytes for the effective packet payload. In addition, except the packets of Page 0, each packet in Seluge and LR-Seluge contains one and n/k hash values, respectively. [1] Therefore, the packets of Seluge and LR-Seluge have $96 - 8 = 88$ bytes and $96 - 8 * n/k = 80$ bytes, respectively, for code-image slices. In our simulations, each page consists of 32 packets for both Seluge and LR-Seluge, and the code image $\mathcal{M}$ is of 20 KB. Under Seluge, $\mathcal{M}$ leads to totally $g = 8$ pages, among which the last page comprises only 9 packets; under LR-Seluge, $\mathcal{M}$ leads to totally $g = 8$ pages as well, all of which comprise 32 packets.

In addition, we set the minimum delays between two advertisement packets and between two SNACK packets to 1 second and 128 ms, respectively, for both Seluge and LR-Seluge. [2] Moreover, we use the delays of 2.5 and 3.5 seconds to emulate the decoding and encoding of a page, respectively, which is clearly in favor of Seluge because the decoding and encoding times are estimated as 1.8 and 3.1 seconds, respectively, in Section 2.5.2.

The performance metrics used in our comparisons include *the total number of data packets*, *the total number of SNACK packets*, *the total number of advertisement packets*, and *the overall dissemination latency* which is defined as the time required to finish disseminating a code image to all the nodes in the network. Since SNACK

[1] The packets of the last page in both Seluge and LR-Seluge do not contain any hash value, but we ignore such subtly and assume that all the packets in both Seluge and LR-Seluge have the same effective payload length.

[2] The two delays are set to 2 seconds and 1 second in [49], respectively. Our simulations indicate that our chosen parameters help significantly reduce the dissemination latency of Seluge.



(a) 20 receivers

(b) $p = 0.3$

Figure 2.3: One-hop one-page scenario.

packets in LR-Seluge are $n - k$ bits longer than those in Seluge, we will also show *the total communication cost* covering data, SNACK and advertisement packets in bytes for fairness. Each measurement in the following figures is the average over 20 simulation runs, each with a different seed.

2.6.1 Validation of Analytical Results

To validate the analytical results in Section 2.5.1, we first simulate the transmission of one page in Seluge and LR-Seluge in a fully-connected one-hop scenario with one local sender and a varying number of local receivers. To fully control and illustrate the impact of packet losses, we use a similar simulation strategy as in [92], where nodes are placed close enough to eliminate packet transmission errors caused by channel impairments, and packet losses are emulated by each node dropping received data, advertisement, or SNACK packets with the same probability p at the application layer.

Figs. 2.3a and 2.3b show the analytical results of Seluge and ACK-based LR-Seluge and the simulation results of Seluge and LR-Seluge. We can see that the simulation result of Seluge closely matches the analytical result, and the number of data packets transmitted in ACK-based LR-Seluge is always larger than that of LR-Seluge obtained from simulations, which confirms that the number of data-packet transmissions in LR-Seluge is upper bounded by that in ACK-based LR-Seluge. In addition, we can

see a significant increase in the number of data packet transmissions in ACK-based LR-Seluge when the packet loss rate increases from 0.3 to 0.4. The reason is that when $p \leq 0.3$ (or $p \geq 0.4$), ACK-based LR-Seluge can finish transmitting one page in one round (or two rounds) with high probability. The figures also confirm that LR-Seluge incurs much fewer data-packet transmissions than Seluge in lossy environments and is less sensitive to the number of receivers as well.

2.6.2 The One-Hop Case

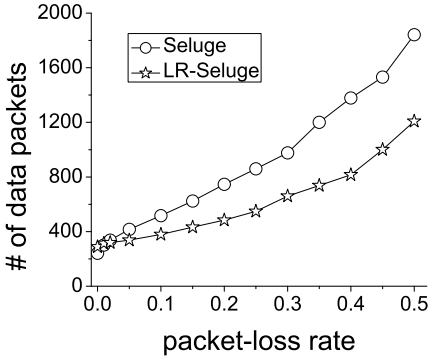
We focus on comparing LR-Seluge and Seluge with simulations which involve disseminating a code image $\mathcal{M}$ of 20 KB. We first show more simulation results for the aforementioned one-hop scenario, which is the basis of the more general multi-hop scenario.

2.6.2.1 Impact of the Packet-Loss Rate

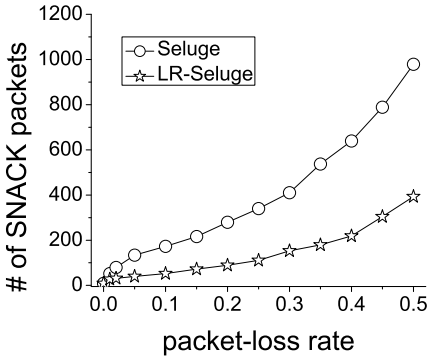
Figs. 2.4a$\sim$2.4e show the impact of the packet-loss rate p on LR-Seluge and Seluge, where there are $N = 20$ local receivers. It is not surprising to see that the total communication costs and dissemination latencies of LR-Seluge and Seluge both increase as p increases. In addition, when $p \leq 0.01$, LR-Seluge has a slightly larger communication cost than Seluge for both data and control packets. There are two reasons. First, LR-Seluge has more data packets than Seluge for the same code image due to the use of erasure codes. Second, under LR-Seluge, each node needs $k' > k$ packets to recover each page, so more data-packet transmissions are needed to disseminate one page if there are no or rare packet losses. In contrast, when $p > 0.01$, LR-Seluge outperforms Seluge in all the five performance metrics. For example, when $p = 0.4$, LR-Seluge reduces the total communication cost by 44% and the dissemination latency by 48% in comparison with Seluge. These results clearly demonstrate that LR-Seluge is much more resilient to packet losses than Seluge.

2.6.2.2 Impact of the Node Density

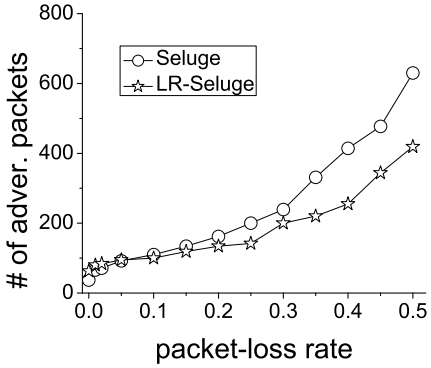
Figs. 2.5a$\sim$2.5e show the impact of the number N of local receivers on LR-Seluge and Seluge, where the packet-loss rate $p = 0.1$. We can see that the communication costs of LR-Seluge and Seluge all increase as N increases. This is understandable because



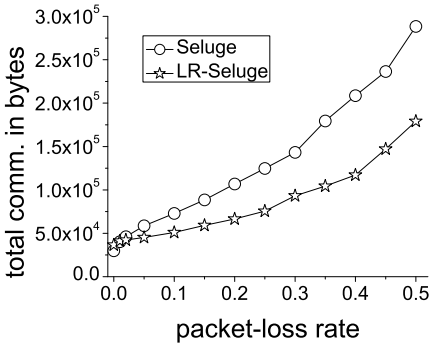
(a) data packet



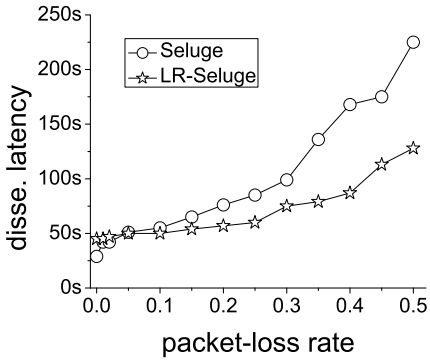
(b) SNACK packet



(c) advertisement packet

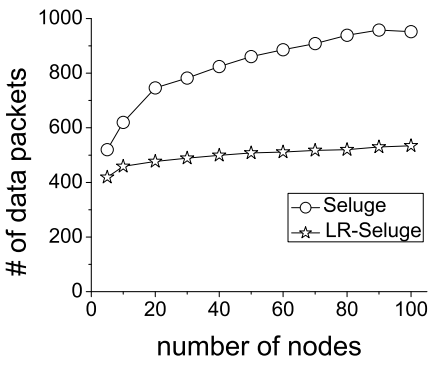


(d) total communication cost

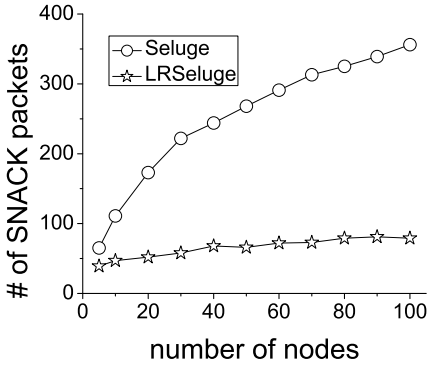


(e) dissemination latency

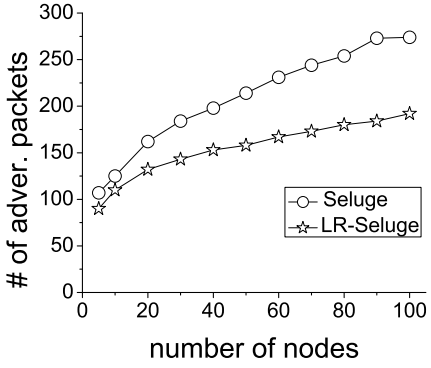
Figure 2.4: Impact of the packet-loss rate, where there are $N = 20$ local receivers.



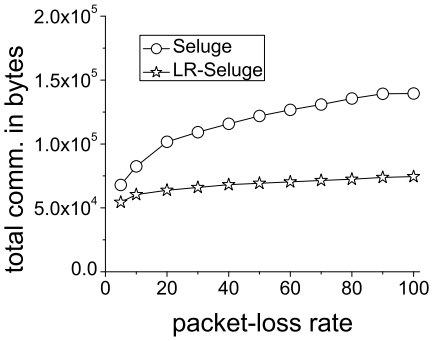
(a) data packet



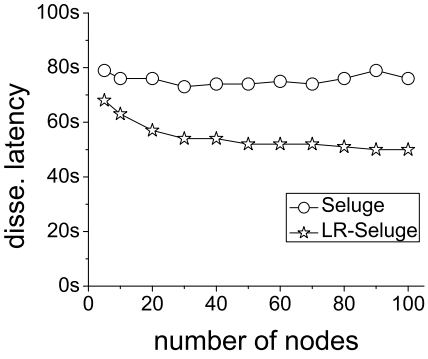
(b) SNACK packet



(c) advertisement packet



(d) total communication cost



(e) dissemination latency

Figure 2.5: Impact of the node density, where the packet-loss rate $p = 0.1$.

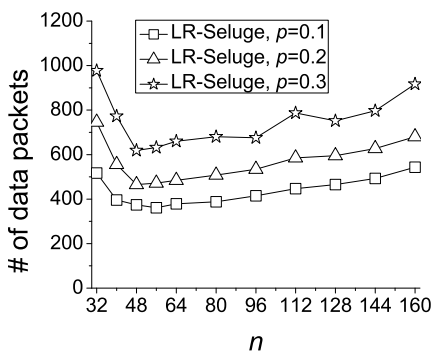
Table 2.2: Performance comparison under network with high density

	LR-Seluge	**Seluge**	Ratio
Total # of SNACK packets	1804	3629	49.71%
Total # of data packets	4040	5496	73.50%
Total # of adver. packets	1059	1678	63.11%
Total comm. cost in bytes	6.05×10^5	8.78×10^5	68.91%
Dissemination latency (s)	93	146	63.70%

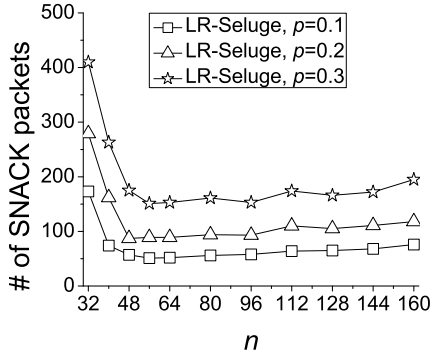
it always requires more data and control packet transmissions to disseminate the same code image under packet losses. However, LR-Seluge is much less sensitive to the increase of N, which can be clearly seen in Figs. 2.5a∼2.5d. In addition, the dissemination latency of Seluge increases slightly as N increases, while that in LR-Seluge slightly decreases. This could be explained as follows. In Seluge, as N increases, the numbers of SNACK and data packet transmissions increase significantly, which leads to higher the dissemination latency. In contrast, the numbers of SNACK and data packet transmissions increase much slower in LR-Seluge as N increases. In addition, the more nodes that demand the current page, the earlier the first node receives k' packets and thus recovers the current page, and the earlier the SNACK packet is transmitted to request the next page. This leads to the decrease in total dissemination latency.

2.6.2.3 Impact of the erasure-coding rate

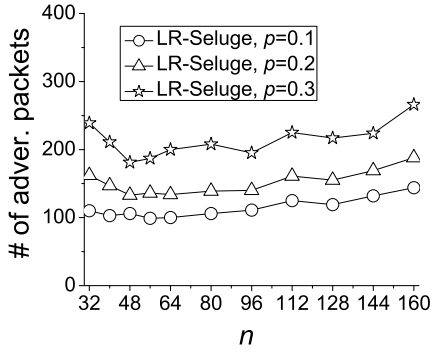
Figs. 2.6a∼2.6e show the impact of the erasure-coding rate n/k on LR-Seluge under different packet-loss rates, where k is fixed to 32. We can see that by introducing a limited number of redundant data packets, the communication cost of LR-Seluge decreases significantly. For example, when $p = 0.1$ and $n = 56$, the total number of SNACK and data packet transmissions decrease by 70.5% and 30%, respectively. As n/k further increases, the communication cost and dissemination latency increase slowly. The reason is that higher erasure-coding rates lead to shorter packet space for code-image slices and thus more packets for the same code image.



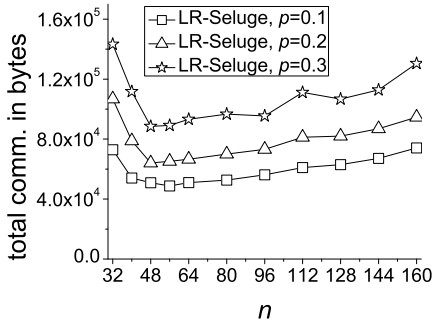
(a) data packet



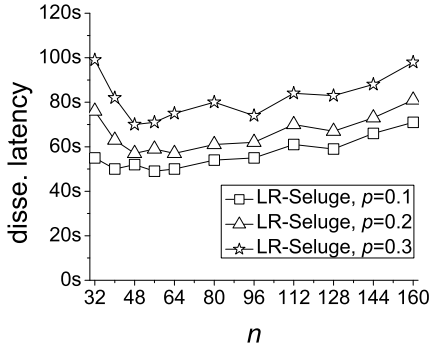
(b) SNACK packet



(c) advertisement packet



(d) total communication cost



(e) dissemination latency

Figure 2.6: Impact of the erasure-coding rate.

Table 2.3: Performance comparison under network with medium density

	LR-Seluge	**Seluge**	Ratio
Total # of SNACK packets	10927	20287	53.86%
Total # of data packets	38197	47373	80.63%
Total # of adver. packets	13088	18812	69.57%
Total comm. cost in bytes	5.55×10^6	7.27×10^6	76.34%
Dissemination latency (s)	1154	1534	75.23%

2.6.3 The Multi-Hop Case

We also simulate LR-Seluge and Seluge in multi-hop networks. In particular, we simulate them under two 15×15 grid sensor networks using the exemplary topologies specified in *15-15-tight-mica2-grid.txt* (high node density) and *15-15-medium-mica2-grid.txt* (low node density) and RF noise and interference from the sample noise trace file *meyer-heavy.txt* of the TinyOS distribution.

Tables 2.2 and 2.3 compare the performance of LR-Seluge and Seluge under these two topologies, respectively. We can see that LR-Seluge outperforms Seluge for all the performance metrics by significant margins, which coincides with the results under the one-hop scenario. We have also simulated other network topologies generated by the topology tool provided by the TinyOS distribution which is based on theoretical propagation models. In general, the results are very similar to those shown in Tables 2.2 and 2.3 and thus are omitted here.

2.6.4 Discussion

We summarize the simulation results as follows.

- LR-Seluge incurs slightly higher communication cost and dissemination latency when packet loss rate is small.
- LR-Seluge can tolerate packet loss while incur much less communication overhead than Seluge does.

- LR-Seluge does not require high erasure code rate, because high rate decreases the effective payload length.

2.7 Related Work

In addition to Deluge [47] and Seluge [49], the following work is most related to our LR-Seluge scheme.

Sluice [60] aims at authenticated code dissemination based on signature and cryptographic hash functions. It creates a chaining relationship among adjacent pages by embedding the hash image of each page into the previous page and signing only the first page. A scheme similar to Sluice is presented in [25], in which the hash image of each packet is included in the previous packet. Both schemes, however, are vulnerable to DoS attacks in which the adversary keeps sending bogus packets that cannot be immediately authenticated, as pointed out in [49]. A scheme with better DoS resilience is presented in [20] and uses Merkle hash trees to enable immediate authentication of packets upon their arrivals. In addition, Tan *at al.* propose a secure code dissemination scheme based on multiple hash chains [105] and also a code dissemination scheme which preserves the confidentiality of the code image [106]. Most recently, Ugus *et al.* [110] present a ROM-friendly secure code dissemination protocol which significantly reduces the memory requirement. All these previous schemes rely on Deluge and thus do not work well in lossy environments. In this chapter, we demonstrate the significant advantages of the proposed LR-Seluge over Seluge which has the best DoS resilience among similar schemes. Also note that the techniques in [110] can be easily incorporated into LR-Seluge to make LR-Seluge also ROM-friendly.

There are also some loss-resilient code dissemination schemes such as Adap-Code [46], Rateless Deluge and ACKless Deluge [43], and SYNAPSE [92], which do not take security into consideration.

2.8 Summary

This chapter presents the design and evaluation of LR-Seluge, the first loss-resilient and secure code dissemination scheme for sensor networks. LR-Seluge achieves loss-

resilience and attack-resilience by seamlessly integrating fixed-rate erasure code and efficient cryptographic primitives. The performance of LR-Seluge is confirmed by both theoretical analysis and thorough simulation results.

Chapter 3

SECURE TOP-k QUERY PROCESSING VIA UNTRUSTED LOCATION-BASED SERVICE PROVIDERS

3.1 Introduction

The explosive growth of Internet-capable and location-aware mobile devices and the surge in social network usage are fostering collaborative information generation and sharing on an unprecedented scale. In particular, IDC believes that total worldwide smartphone shipments will reach 659.8 million units in 2012 and will grow at a CAGR of 18.6% until 2016.[1] Almost all smartphones have cellular/WiFi Internet access and can always acquire their precise locations via pre-installed positioning software. Also owing to the growing popularity of social networks, it is more and more convenient and motivating for mobile users to share with others their experience with all kinds of *points of interest*s (POIs) such as bars, restaurants, grocery stores, coffee shops, and hotels. Meanwhile, it becomes commonplace for people to perform various spatial POI queries at online location-based service providers (LBSPs) such as Google and Yelp. As probably the most familiar type of spatial queries, a spatial (or location-based) top-k query asks for the POIs in a certain region and with the highest k ratings for a given POI attribute. For example, one may search for the best 10 Italian restaurants with the highest food ratings within five miles of his current location. This chapter focuses on spatial top-k queries, and the term "spatial" will be omitted hereafter for brevity.

We observe two essential drawbacks with current top-k query services. First, individual LBSPs often have very small data sets comprising POI reviews. This would largely affect the usefulness and eventually hinder the more prevalent use of spatial top-k query services. Continue with the restaurant example. The data sets at individual LBSPs may not cover all the Italian restaurants within a search radius. Additionally, the same restaurant may receive diverse ratings at different LBSPs, so users may get confused by very different query results from different LBSPs for the same query. A leading reason for limited data sets at individual LBSPs is that people tend to leave

[1] `http://www.idc.com/getdoc.jsp?containerId=233553`

reviews for the same POI at one or at most only a few LBSPs's websites which they often visit. Second, LBSPs may modify their data sets by deleting some reviews or adding fake reviews and return tailored query results in favor of the restaurants which would like to pay. Even if LBSPs are not malicious, they may return unfaithful query results under the influence of various attacks such as the Sybil attack [127,128] whereby the same attacker can submit many fake reviews for the same POI. In either case, top-k query users may be misled by the query results to make unwise decisions.

A promising solution to the above two issues is to introduce some trusted data collectors as the central hubs for collecting POI reviews. In particular, data collectors can offer various incentives such as free coffee coupons for stimulating review submissions and then profit by selling the review data to individual LBSPs. Instead of submitting POI reviews to individual LBSPs, people (called *data contributor*s) can now submit them to a few data collectors to earn rewards. The data sets maintained by data collectors can thus be considered the union of the small data sets currently at individual LBSPs. Such centralized data collection also makes it much easier and feasible for data collectors to employ sophisticated defenses such as [127, 128] to filter out fake reviews from malicious entities like Sybil attackers. Data collectors can be either new service providers or more preferably existing ones with a large user base, such as Google, Yahoo, Facebook, Twitter, and MSN. Many of these service providers have offered open APIs for exporting selected data from their systems. We postulate that they may act as location-based data collectors and sellers if sound techniques and business models are in place.

The above system model is also highly beneficial for LBSPs. In particular, they no longer need struggle to solicit faithful user reviews, which is often a daunting task especially for small/medium-scale LBSPs. Instead, they can focus their limited resources on developing appealing functionalities (such as driving directions and aerial photos) combined with the high-quality review data purchased from data collectors. The query results they can provide will be much more trustworthy, which would in turn help them

attract more and more users. This system model thus can greatly help lower the entrance bar for new LBSPs without sufficient funding and thus foster the prosperity of location-based services and applications.

A main challenge for realizing the appealing system above is how to deal with untrusted and possibly malicious LBSPs. Specifically, malicious LBSPs may still modify the data sets from data collectors and provide biased top-k query results in favor of POIs willing to pay. Even worse, they may falsely claim generating query results based on the review data from trusted data collectors which they actually did not purchase. Moreover, non-malicious LBSPs may be compromised to return fake top-k query results.

In this chapter, we propose three novel schemes to tackle the above challenge for fostering the practical deployment and wide use of the envisioned system. The key idea of our schemes is that the data collector precomputes and authenticates some auxiliary information (called *authenticated hint*s) about its data set, which will be sold along with its data set to LBSPs. To faithfully answer a top-k query, a LBSP need return the correct top-k POI data records as well as proper *authenticity* and *correctness* proofs constructed from authenticated hints. The authenticity proof allows the query user to confirm that the query result only consists of authentic data records from the trusted data collector's data set, and the correctness proof enables the user to verify that the returned top-k POIs are the true ones satisfying the query. The first two schemes both target snapshot top-k queries but differ in how authenticated hints are precomputed and how authenticity and correctness proofs are constructed and verified as well as the related communication and computation overhead. The third scheme, built upon the first scheme, realizes efficient and verifiable moving top-k queries. The efficacy and efficiency of our schemes are thoroughly analyzed and evaluated through detailed simulation studies.

The rest of this chapter is organized as follows. Section 3.2 discusses the related work, and Section 3.3 gives the problem formulation. Section 3.4 presents two schemes for secure snapshot top-k query processing, which are extended for secure

moving top-k query processing in Section 3.5. All the schemes are then thoroughly analyzed in Section 3.6 and evaluated via detailed simulations in Section 3.7. This chapter is finally concluded in Section 3.8.

3.2 Related Work

Our work is most related to data outsourcing [42], for which we can only review representative schemes due to space constraints. The framework of data outsourcing was first introduced in [42], in which a data owner outsources its data to a third-party service provider who is responsible for answering the data queries from either the data owner or other users. In general, there are two security concerns in data outsourcing: data privacy and query integrity [59].

Ensuring data privacy requires the data owner to outsource encrypted data to the service provider, and efficient techniques are needed to support querying encrypted data. A bucketization approach was proposed in [41, 45] to enable efficient range queries over encrypted data. Shi *et al.* presented novel methods for multi-dimensional range queries over encrypted data [102]. Some most recent proposals aim at secure ranked keyword search [9, 10] or fine-grained access control [129] over encrypted data. This line of work is orthogonal to our work, as we focus on publicly accessible location-based data without need for privacy protection.

Another line of research has been devoted to ensure query integrity, i.e., that a query result was indeed generated from the outsourced data (the authenticity requirement) and contains all the data satisfying the query (the correctness requirement). In these schemes, the data owner outsources both its data and also its signatures over the data to the service provider which returns both the query result and a *verification object* (VO) computed from the signatures for the querying user to verify query integrity. Many techniques were proposed for signature and VO generations, such as those [79, 83, 84] based on signature chaining and those [59, 119, 123, 124] based on the Merkle hash tree [72] or its variants. None of these schemes consider spatial top-k queries and thus are not directly applicable to our intended scenario.

Secure remote query processing in tiered sensor networks [14, 101, 103, 133, 134] is also loosely related to our work here. These schemes assume that some master nodes are in charge of storing data from regular sensor nodes and answering the queries from the remote network owner. Various techniques were proposed in [14, 101, 103, 134] to ensure data privacy against master nodes and also enable the network owner to verify range-query integrity. Moreover, Zhang *et al.* [133] proposed efficient techniques for the network owner to validate the integrity of top-k queries. These schemes cannot be adapted to address our problem in this chapter.

3.3 Problem Formulation

In this section, we first introduce our system model and then formulate the problem.

3.3.1 System Model

We assume a distributed system comprising a data collector, data contributors, LBSPs, and top-k query users. Data contributors are common people who submit POI reviews to the data collector's website. The data collector normally need offer some incentives such as FourSquare's badges to stimulate review submissions and also employ necessary countermeasures such as [127, 128] to filter out fake reviews from malicious data contributors. The data collector sells aggregated POI reviews in the form of a location-based data set to individual LBSPs. Every LBSP operates a website for users to perform top-k queries over the purchased data set and may add some appealing functionalities to the query result such as street maps and photos. In addition, although there might be multiple data collectors with each selling data to a number of LBSPs, we hereafter focus on one pair of data collector and LBSP for the purpose of this chapter.

The data set is classified according to POI categories such as restaurants, bars, and coffee shops, and it contains a unique record for every POI in every category. As a result, POIs falling into multiple categories (e.g., both a restaurant and bar) have one record for every affiliated category. This chapter focuses on top-k queries involving a single category, which are most commonly used in practice, and the extension of our schemes to involve multiple categories is part of our future work. In particular, our

discussion will focus on one POI category whose total data records form a set $\mathcal{D}$. For simplicity, we assume that the category has one numerical attribute taking values from a given range. For instance, if restaurant is the category under consideration, there may be $\lambda = 4$ attributes including *food*, *price*, *service*, and *hygiene*, with each rated on a scale of 1 to 10.

The geographic area covered by the data collector is partitioned into $M \geq 1$ equally-sized non-overlapping zones. For every zone i, let n_i denote the number of POIs, and POI$_{i,j}$ and $D_{i,j}$ denote the jth POI and its corresponding data record, respectively. It follows that $\mathcal{D} = \bigcup_{i=1}^{M} \mathcal{D}_i$, $\mathcal{D}_i = \bigcup_{j=1}^{n_i} D_{i,j}$, and $\mathcal{D}_i \bigcap \mathcal{D}_j = \phi$ for all $i \neq j$. Also note that $\mathcal{D}_i$ can be empty for some $i \in [1, M]$, meaning that there is no POI in zone i that has been reviewed.

To illustrate the content of a data record, assume that the data collector got reviews about POI$_{i,j}$ from $n_{i,j}$ data contributors. Every review includes a rating on every attribute and possibly text comments. We also let $A_{i,j,q}$ denote the rating for attribute q averaged over $n_{i,j}$ individual ratings. The data record $d_{i,j}$ for POI$_{i,j}$ includes its name, location $l_{i,j}$, $\{A_{i,j,q}\}_{q=1}^{\lambda}$, $n_{i,j}$ reviews, and possibly other information.

3.3.2 Problem Statement

We consider two types of top-k queries in this chapter. A *snapshot top-k query* includes the interested POI *category*, a *query region* $\mathbf{R}$, and an integer $k \geq 1$. As an example, the POI category and attribute can be *restaurant* and *food*, respectively. The query region can be in multiple formats. For instance, the user can specify a GPS location or street address along with a search radius, and he may also select multiple zones on a map provided by the LBSP. An authentic and correct query result should include the records for k POIs in the specified category of the data collector's true data set, all of which are in the query region $\mathbf{R}$, have the attribute-q rating among the highest k, and are ordered with respect to the attribute-q rating in the descending order. For brevity, we will refer to the POIs that are both authentic and correct as top-k POIs hereforth. In contrast, a *moving top-k query* can be viewed as the continuous version of snapshot

top-k queries, whereby the user is interested in the top-k POIs in a moving region $\mathbf{R}$ defined with respect to the user's current location.

We assume that the data collector is trusted, while the LBSP is untrusted. In particular, the LBSP may alter the query result in favor of the POIs willing to pay, to which similar misbehavior has been widely reported in web-search industry. For example, the LBSP may replace some true top-k POIs with others not among the top k or even not in the data collector's data set, and it may also modify some data records by adding more good reviews and deleting bad ones. In addition, a LBSP good in nature may be compromised by attackers to forge query results as well.

Our design objective is to enable the user to verify the authenticity and correctness of the query result returned by the LBSP. The query result is considered authentic if all its k POI records exist in the data collector's data set and have not been tampered with, and it is called correct if it contains the true top-k POI records in the query region.

3.4 Secure Snapshot Top-k Query Processing

In this section, we propose two novel schemes for secure snapshot top-k query processing via untrusted LBSPs.

3.4.1 Overview: Design Challenge and Basic Idea

The main design challenge is the lack of shared information between the data collector and top-k query users. On the one hand, the data collector cannot predict the content of any top-k query from arbitrary users. On the other hand, users do not know the data collector's data set and thus have difficulty in verifying the authenticity and correctness of query results. The only entity knowing both the query content and the data set is the untrusted LBSP. A seemingly workable solution is to let the data collector attach a digital signature to every POI record for the receiving user to verify. This solution, however, can only enable authenticity verification, and incorrect query results comprising only authentic POI records can still escape detection.

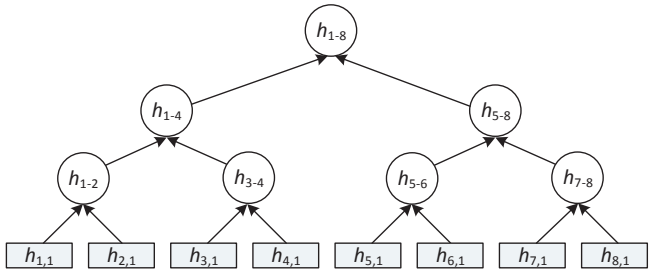


Figure 3.1: An example of constructing the Merkle hash tree over $\{h_{i,1}\}_{i=1}^{8}$.

The above challenge motivates the key idea underlying our two schemes. In particular, we let the data collector precomputes and authenticates some auxiliary information (called *authenticated hint*s) about its data set, which will be sold along with its data set to LBSPs. To faithfully answer a top-k query, the LBSP need return the true top-k POI data records as well as proper authenticity and correctness proofs constructed from authenticated hints. As the names suggest, authenticity and correctness proofs enable the user to verify the authenticity and correctness of the query result, respectively. Our two schemes differ in how authenticated hints are precomputed and how authenticity and correctness proofs are constructed and verified.

In the remainder of this section, we illustrate our two schemes which both comprises three phases and differ in operation details. In the *data-preprocessing* phase, the data collector uses cryptographic methods to create authenticated hints over its data set. In the subsequent *query-processing* phase, the LBSP answers a top-k query by returning the query result as well as the authenticity and correctness proofs to the query user. In the final *verification* phase, the user verifies authenticity and correctness proofs. For ease of presentation, we shall temporarily assume that no two POIs have the same rating for any attribute $q \in [1, \lambda]$, which implies that there is one and only one correct result for any top-k query. We will also temporarily assume that there are always at least k POIs in the query region so that the query result contains exactly k POI records for arbitrary k. These two assumptions are relaxed in Section 3.4.4.

3.4.2 Scheme 1

In Scheme 1, authenticated hints are created by chaining ordered POIs in every zone via cryptographic hash functions and then tieing the POIs in different zones via a Merkle hash tree [71]. The details about constructing and using authenticated hints are as follows.

3.4.2.1 Data Preprocessing

The data collector preprocesses its data set $\mathcal{D} = \bigcup_{i=1}^{M} \mathcal{D}_i$ before selling it to LBSP-s, where M denotes the total number of zones. Recall that $\mathcal{D}_i = \bigcup_{j=1}^{n_i} D_{i,j}$, where $D_{i,j}$ denotes the record of POI$_{i,j}$ and includes its name, location $l_{i,j}$, received ratings $\{A_{i,j,q}\}_{q=1}^{\lambda}$ for q attributes, individual reviews, and some other information. The data collector performs the following operations for every attribute $q \in [1, \lambda]$.

First, for each $i \in [1, M]$, the data collector sorts $\mathcal{D}_i$ according to the attribute-q rating to generate an orderer list $\mathcal{D}'_i = \langle D'_{i,1}, D'_{i,2}, \ldots, D'_{i,n_i} \rangle$ such that $A'_{i,1,q} > A'_{i,2,q} > \cdots > A'_{i,n_i,q}$. It then computes an *index* for every $D'_{i,j} \in \mathcal{D}'_i$ as

$$\phi_{i,j} = \langle l'_{i,j}, A'_{i,j,q}, H(D'_{i,j}) \rangle \ , \tag{3.1}$$

where $l'_{i,j}$ denotes the location of $D'_{i,j}$, and $H(\cdot)$ denotes a cryptographic hash function. Note that $\phi_{i,j}$ contains sufficient information for a user to determine whether $D'_{i,j}$ satisfies a top-k query, which will be further illustrated shortly.

Second, the data collector chains $\{\phi_{i,j}\}_{j=1}^{n_i}$ using cryptographic hash functions to enable authenticity verifications of query results. In particular, recall that every attribute rating is on a given range $[A_{\min}, A_{\max}]$, say [1, 10]. Let χ denote a publicly known number smaller than $A_{\min}$. The data collector recursively computes a sequence of hash values as follows,

$$h_{i,j} = \begin{cases} H(\chi) & j = n_i + 1, \\ H(h_{i,j+1} || \phi_{i,j}) & 1 \leq j \leq n_i, \end{cases} \tag{3.2}$$

where $||$ denotes concatenation and $n_i \geq 0$. Note that if $n_i = 0$, i.e., there is no POI in zone i, we let $\phi_{i,1} = h_{i,1} = H(\chi)$.

Finally, the data collector builds a Merkle hash tree over $\{h_{i,1}\}_{i=1}^{M}$ to enable efficient authentication of query results. More specifically, assuming that $M = 2^d$ for some integer d, the data collector builds a binary tree of depth d, in which every leaf node corresponds to one of $\{h_{i,1}\}_{i=1}^{M}$, and every non-leaf node is computed as the hash of the concatenation of its immediate children nodes. We also define an *auxiliary set* $\mathcal{T}_i$ as the set of non-leaf nodes required along with any leaf node $h_{i,1}$ to compute the Merkle root hash. An example for $M = 8$ is shown in Fig. 3.1, in which $h_{1-2} = H(h_{1,1}||h_{2,1})$, $h_{3-4} = H(h_{3,1}||h_{4,1})$, $h_{5-6} = H(h_{5,1}||h_{6,1})$, $h_{7-8} = H(h_{7,1}||h_{8,1})$, $h_{1-4} = H(h_{1-2}||h_{3-4})$, $h_{5-8} = H(h_{5-6}||h_{7-8})$, and $h_{1-8} = H(h_{1-4}||h_{5-8})$. If $h_{3,1}$ is the given leaf node, we then have $\mathcal{T}_3 = \{h_{4,1}, h_{1-2}, h_{5-8}\}$, as the root $h_{1-8} = H(H(h_{1-2}||H(h_{3,1}||h_{4,1}))||h_{5-8})$. Note that if M is not a power of two, some dummy leaf nodes need be introduced for constructing the Merkle hash tree.

Since there are totally λ attributes, every POI$_{i,j}$ has λ indexes, based on which the data collector builds a separate Merkle hash tree for every attribute and signs every root using its private key. In addition, the data collector need perform the above operations separately for the data set of every POI category.

3.4.2.2 Query Processing

The LBSP purchases the data sets of interested POI categories from the data collector. For every POI category selected by the LBSP, the data collector returns the original data set $\mathcal{D}$, the signatures on λ Merkle root hashes, and all the intermediate results for constructing the Merkle hash tree. Alternatively, the data collector can just return the first two pieces of information and let the LBSP itself perform a one-time process to derive the third piece in the same way as the date collector.

Now we illustrate the processing of a snapshot top-k query, including the desired POI category, the interested attribute $q \in [1, \lambda]$ for ranking POIs, the query region $\mathbf{R}$, and k. We denote the k POIs in $\mathbf{R}$ with the highest k attribute-q ratings by kPOI, a-

mong which the lowest attribute-q rating is denoted by γ. In addition, we call each zone either completely or partially covered by the query region a *candidate zone*. A correct and authentic query result needs to satisfy two conditions. The correctness condition requires the query result to contain at least the following information: (1) the complete data records for kPOI; (2) the data indexes (much shorter than data records) for all the POIs in each candidate zone but not in $\mathbf{R}$ whose attribute-q rating is larger than γ; and (3) some additional information to prove that the query result includes either the data record or index of every POI in every candidate zone with attribute-q rating not smaller than γ. In addition, the authenticity condition requires that the query result include the auxiliary set for every candidate zone for the calculation and verification of the qth Merkle root hash.

To satisfy the correctness condition, the LBSP first searches $\{\mathcal{D}'_i\}_{i=1}^{M}$ to locate kPOI and then determine the lowest attribute-q rating γ. Next, the LBSP determines the set of candidate zones, denoted by $\mathcal{I} \subseteq \{1, \ldots, M\}$. Let τ_i the number of POIs in zone i with attribute-q ratings higher than γ. Apparently, we have $n_i \geq \tau_i, \forall i \in \mathcal{I}$. It follows that $\sum_{i \in \mathcal{I}} \tau_i \geq k$, which holds because any candidate zone that partially overlaps with $\mathbf{R}$ may have some POIs outside $\mathbf{R}$ but with attribute-q ratings higher than γ. We further define

$$X_{i,j} = \begin{cases} D'_{i,j} & \text{if } l'_{i,j} \in \mathbf{R}, \\ \phi_{i,j} & \text{o.w.}, \end{cases} \tag{3.3}$$

for all $i \in \mathcal{I}, j \in [1, n_i]$. In other words, $X_{i,j}$ equals $D'_{i,j}$ if the POI is in $\mathbf{R}$ and a shorter index otherwise. The LBSP returns the following information $\mathcal{S}_i$ for each candidate zone $i \in \mathcal{I}$ in the query result to enable correctness verification.

- Case 1: if $n_i = 0$, $\mathcal{S}_i = \langle i \rangle$.
- Case 2: if $n_i = 1$, $\mathcal{S}_i = \langle i, X_{i,1} \rangle$.
- Case 3: if $n_i \geq 2$ and $\tau_i = 0$, $\mathcal{S}_i = \langle i, \phi_{i,1}, h_{i,2} \rangle$.

- Case 4: if $n_i \geq 2$ and $n_i > \tau_i \geq 1$,

$$\mathcal{S}_i = \langle i, X_{i,1}, \ldots, X_{i,\tau_i}, \phi_{i,\tau_i+1}, h_{i,\tau_i+2} \rangle \ .$$

- Case 5: if $n_i = \tau_i \geq 2$, $\mathcal{S}_i = \langle i, X_{i,1}, \ldots, X_{i,\tau_i} \rangle$.

Note that the last two fields in both Case 3 and Case 4 correspond to the POI in zone i with the largest attribute-q rating smaller than γ. Since the POIs in zone i have been ranked and chained together under cryptographic hash functions during data preprocessing, the inclusion of such fields is necessary for proving that every POI in every candidate zone whose attribute-q rating not smaller than γ has been covered in the query result in the form of either a data record or index. Such information has been implicitly covered in the other three cases as well. In addition, the LBSP returns $\mathcal{T} = \bigcup_{i \in \mathcal{I}} \mathcal{T}_i$ and the data collector's signature on the qth Merkle root hash to enable authenticity verification.

3.4.2.3 Query-Result Verification

Now we discuss how the user verifies the authenticity and correctness of the query result, which can be done via a small plug-in developed by the data collector and installed on his web browser. The security analysis of Scheme 1 is postponed to Section 3.6.

For authenticity verification, the user checks if every piece of information in the query result can lead to the same Merkle root hash matching the data collector's signature. Specifically, the user first determines which of the above five cases $\mathcal{S}_i$ ($\forall i \in \mathcal{I}$) belongs to based on its message format. He then derives the indexes for all related POIs in $\{\mathcal{S}_i\}_{i \in \mathcal{I}}$. Note that the indexes of the POIs outside $\mathbf{R}$ are explicitly included in $\{\mathcal{S}_i\}_{i \in \mathcal{I}}$, while those of the POIs in $\mathbf{R}$ can be computed from their corresponding data records in $\{\mathcal{S}_i\}_{i \in \mathcal{I}}$. Subsequently, the user computes $h_{i,1}$ for each $i \in \mathcal{I}$ according to Eq. (3.2). Since the auxiliary information $\mathcal{T}_i$ for $h_{i,1}$ is also in the query result, the user further uses $h_{i,1}$ and $\mathcal{T}_i$ to compute the Merkle root hash. If the query result is authentic, the user can derive the same root hash for each $i \in \mathcal{I}$, in which case he further verifies

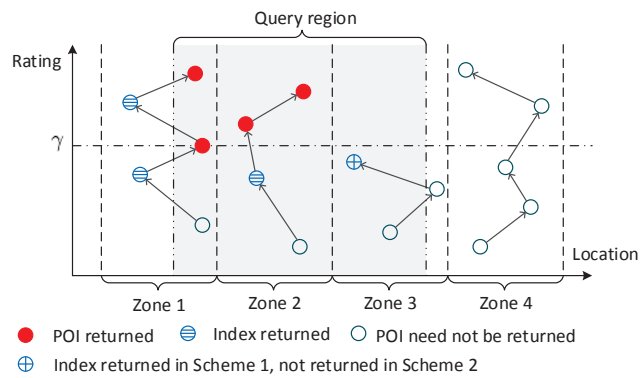


Figure 3.2: An example for Scheme 1, where $M = 4$, $k = 4$, and the dots in zone i correspond to POI records $D'_{i,1}$ to $D'_{i,4}$ from top to bottom.

whether the data collector's signature in the query result is a valid signature on the derived root hash. If so, he considers the query result authentic.

To perform correctness verification, the user first checks if zones $\mathcal{I}$ encloses the query region $\mathbf{R}$. If so, he proceeds with the following verifications in accordance with the aforementioned correctness condition used in query processing.

1. There are exactly k data records in the query result with POI locations all in $\mathbf{R}$, which correspond to the top-k POIs (i.e., kPOI) in $\mathbf{R}$. If so, the user locates the lowest attribute-k rating γ.

2. None of the POIs for which the data indexes (instead of data records) are returned satisfy the query. In particular, for each index $\phi_{i,j}$ ($i \in \mathcal{I}$), at least one of the following conditions does not hold.

   - $\phi_{i,j}$ contains a location $l'_{i,j} \in \mathbf{R}$.
   - $\phi_{i,j}$ contains an attribute$-q$ rating $A'_{i,j,q} > \gamma$.

In addition, since the query result is authentic, it must include either the data record or index for every POI in every candidate zone whose attribute-q rating is not smaller than γ. Therefore, the user considers the query result correct if the above two verifications succeed.

3.4.2.4 An Example

To better illustrate Scheme 1, we show an example in Fig. 3.2 with $M = 4$ zones, where we assume one-dimensional POI locations for simplicity, i.e., that all POIs are distributed on a straight line, and all the shown POIs have been ordered according to the attribute-q rating (q is omitted from subscripts for brevity). Suppose that the user queries the top-4 POIs in the query region that completely covers zone 2 and partially overlaps with zones 1 and 3. It follows that $\mathcal{I} = \{1, 2, 3\}$, and τ_1, τ_2, τ_3 are 3, 2, 0, respectively. For zone 1, there is one POI outside the query region with a rating higher than γ, so we have $\mathcal{S}_1 = \langle 1, D'_{1,1}, \phi_{1,2}, D'_{1,3}, \phi_{1,4}, h_{1,5} \rangle$. Similarly, we have $\mathcal{S}_2 = \langle 2, D'_{2,1}, D'_{2,2}, \phi_{2,3}, h_{2,4} \rangle$ for zone 2 and $\mathcal{S}_3 = \langle 3, \phi_{3,1}, h_{3,2} \rangle$ for zone 3. The query result includes $\mathcal{S}_1$, $\mathcal{S}_2$, $\mathcal{S}_3$, the auxiliary indexes $\{\mathcal{T}_i\}_{i=1}^{3}$, and the data collector's signature on h_{1-4} which is the root of the Merkle hash tree with depth $d = 2$. Based on $\mathcal{S}_1$, $\mathcal{S}_2$, and $\mathcal{S}_3$, the user can derive $h_{1,1}, h_{2,1}$, and $h_{3,1}$, respectively. He can further compute three Merkle root hashes using $h_{1,1}$ and $\mathcal{T}_1$, $h_{2,1}$ and $\mathcal{T}_2$, and $h_{3,1}$ and $\mathcal{T}_3$, respectively. If the three root hashes are equal and match the data collector's signature, the user considers the query result authentic. If the query result can also pass the aforementioned three correctness verifications, the user considers the query result both authentic and correct.

3.4.3 Scheme 2

Scheme 1 requires the LBSP to return some information for every candidate zone even if it has no top-k POI satisfying the query. This may incur significant communication overhead for a large query region. Given this observation, we propose Scheme 2 which works by embedding some information among nearby zones to dramatically reduce the amount of information returned to the user.

The basic idea of Scheme 2 can be better illustrated using a simple example. Assume that zones i and j are two candidate zones. But neither contains a top-k POI. Under Scheme 1, the LBSP need return both $\langle i, \phi_{i,1}, h_{i,2}, \mathcal{T}_i \rangle$ and $\langle j, \phi_{j,1}, h_{j,2}, \mathcal{T}_j \rangle$ to

prove that no POI in zones i or j satisfies the query. In contrast, if we could consider zones i and j as one virtual zone, the LBSP only need return $\langle x, \phi_{x,1}, h_{x,2}, \mathcal{T}_x \rangle$, where $x = i$ if the largest attribute-q rating in zone j is smaller than that in zone i, and $x = j$ otherwise. The amount of information returned to the user can thus be reduced.

3.4.3.1 Data Preprocessing

To implement the basic idea exemplified above, the data collector binds to every POI data index some additional information about the POIs in adjacent zones. In particular, the data collector partitions the original M zones into non-overlapping *macro zone*s, each consisting of m nearby zones, where m is a public system parameter. Assuming that M is divisible by m, we let $\mathcal{M}_e$ denote the set of zones composing the macro zone $e \in [1, M/m]$.

Consider a macro zone e as an example. As in Scheme 1, the data collector first sorts $\mathcal{D}_i$ for every zone $i \in \mathcal{M}_e$ according to the descending order of the attribute-q rating to generate an orderer list $\mathcal{D}'_i = \langle D'_{i,1}, D'_{i,2}, \ldots, D'_{i,n_i} \rangle$. Let $A'_{j,0,q} = \overline{\chi}$ and $A'_{j,n_i+1,q} = \chi$ denote two public values larger than the largest possible attribute rating and smaller than the smallest possible attribute rating, respectively. The data collector further generates $\{\mathcal{I}_{i,j}\}_{j=1}^{n_i+1}$, where $\mathcal{I}_{i,j} = \{\langle s, A'_{s,1,q} \rangle | s \in \mathcal{M}_e \setminus \{i\}\}$ with $A'_{s,1,q} \in (A'_{i,j-1,q}, A'_{i,j,q})$. In other words, $\mathcal{I}_{i,j}$ comprises all the other zones in $\mathcal{M}_e \setminus \{i\}$ and their largest attribute-q ratings in $(A'_{i,j-1,q}, A'_{i,j,q})$. Apparently, we have $|\bigcup_{j=1}^{n_i+1} \mathcal{I}_{i,j}| = |\mathcal{M}_e \setminus \{i\}| = m - 1$. The data collector then computes an index as

$$\phi_{i,j} = \langle l_{i,j}, \mathcal{I}_{i,j}, A'_{i,j,q}, H(\mathcal{I}_{i,j} || D'_{i,j}) \rangle \tag{3.4}$$

for all $j \in [1, n_i]$ and chains $\{\phi_{i,j}\}_{j=1}^{n_i}$ according to Eq. (3.2). Finally, it builds a Merkle hash tree over $\{h_{i,1}\}_{i=1}^{M}$ and signs the root as in Scheme 1. The essential difference in data preprocessing between Schemes 1 and 2 thus lies in the construction of POI indexes.

As in Scheme 1, the data collector builds a separate Merkle hash tree for every attribute $q \in [1, \lambda]$ in every POI category and signs every Merkle root hash using its

private key.

3.4.3.2 Query Processing

The LBSP purchases the original data set $\mathcal{D}$, the signatures on λ Merkle root hashes, and all the intermediate results for constructing the Merkle hash tree of every interested POI category from the data collector.

After receiving a top-k query, the LBSP first determines the top-k POIs (i.e., kPOI) in the query region $\mathbf{R}$ and also the set of candidate zones $\mathcal{I} \subseteq \{1, \ldots, M\}$. The LBSP then determines the lowest attribute-q rating γ in kPOI and τ_i as the number of POIs in zone $i \in \mathcal{I}$ with attribute-q ratings not smaller than γ. Next, the LBSP defines

$$Y_{i,j} = \begin{cases} D'_{i,j} || \mathcal{I}_{i,j} & \text{if } l'_{i,j} \in \mathbf{R}, \\ \phi_{i,j} & \text{o.w.}, \end{cases} \tag{3.5}$$

for all $i \in \mathcal{I}, j \in [1, n_i]$, where $\mathcal{I}_{i,j}$ is as defined in the data-preprocessing phase. The query result includes the following information $\mathcal{S}_i$ for each zone $i \in \mathcal{I}$ with $\tau_i > 0$.

- Case 1: if $n_i = \tau_i \geq 1$, $\mathcal{S}_i = \langle i, Y_{i,1}, \ldots, Y_{i,\tau_i} \rangle$.
- Case 2: if $n_i \geq 2$ and $n_i > \tau_i > 0$,

$$\mathcal{S}_i = \langle i, Y_{i,1}, \ldots, Y_{i,\tau_i}, \phi_{i,\tau_i+1}, h_{i,\tau_i+2} \rangle \ .$$

Moreover, let $\mathcal{M}'_e = \{i | i \in \mathcal{M}_e \bigcap \mathcal{I}, \tau_i < n_i, n_i \neq 0\}$ denote the zones with at least one attribute-q rating smaller than γ in every macro zone $e \in [1, M/m]$. There are two cases.

- Case 3: if there is zone $i \in \mathcal{M}'_e, \tau_i > 0$, nothing need be done because this case has been covered by Case 2.
- Case 4: otherwise, we have $\tau_i = 0, \forall i \in \mathcal{M}'_e$. Assuming that $A'_{j,1,q}$ is the highest attribute-q rating in $\mathcal{M}'_e$, the LBSP also adds $\mathcal{S}_j = \langle j, \phi_{j,1}, h_{j,2} \rangle$ to the query result.

Furthermore, for any candidate macro zone e, if there is no POI in zones $\mathcal{M}_e \bigcap \mathcal{I}$ with attribute-q rating not smaller than γ, we must have $n_i = \tau_i$ for all $i \in \mathcal{M}_e \bigcap \mathcal{I}$, in which case the LBSP is required to return $\mathcal{S}_i = \langle i \rangle$ for each $i \in \mathcal{M}_e \bigcap \mathcal{I}$ if $n_i = \tau_i = 0$ (Case 5). Note that the case for $n_i = \tau_i > 0$ has been covered by Case 1 above.

As in Scheme 1, the LBSP additionally returns the data collector's signature on the qth Merkle root hash and $\mathcal{T} = \bigcup_{i \in \mathcal{I}'} \mathcal{T}_i$, where $I' \subseteq \mathcal{I}$ is the set of zones in which there at least one POI data record or index has been included in the query result. In contrast to Scheme 1, $\langle i, \phi_{i,1}, h_{i,2}, \mathcal{T}_i \rangle$ need not be returned for any zone $i \in \mathcal{I}$ when $\tau_i = 0$ in most cases due to the macro-zone idea, which can lead to much lower computation and communication overhead in practice.

3.4.3.3 Query-Result Verification

After receiving the query result, the user first verifies its authenticity as in Scheme 1. If the authentication succeeds, he proceeds with correctness verification by checking whether the query result contains some information for every candidate macro zone $e \in [1, M/m]$ that overlaps with the query region $\mathcal{R}$. This verification should succeed for a correct query result according to the query-processing process. If so, the user further checks that the query result satisfies the same two conditions as in Scheme 1 (see Section 3.4.3.3) and then determines the lowest attribute-q rating γ among kPOI. Subsequently, based on the information format $\mathcal{S}_i$ for every zone i in the query result, the user determines τ_i (i.e., the number of POIs in zone i with attribute-q ratings $\geq \gamma$) and also the relationship between τ_i and n_i (the total number of POIs in zone i).

Unlike Scheme 1, Scheme 2 does not require some information to be returned for every candidate zone $i \in \mathcal{I}$ overlapping with the query region $\mathbf{R}$ if $\tau_i = 0$. The LBSP may exploit this situation and return no information for zone i even if $\tau_i > 0$. To detect this possible attack, the user conducts the following verifications for every candidate macro zone e in accordance with the five cases in query processing.

- If there is any zone $i \in \mathcal{I} \bigcap \mathcal{M}_e$ with $0 < \tau_i < n_i$ (i.e., Case 2 in query processing), the user checks whether the query result contains a valid $\mathcal{S}_x$ field corresponding to Case 1 or 2 in query processing for every zone $x \in \mathcal{I} \bigcap \mathcal{M}_e \bigcap (\bigcup_{j=1}^{\tau_i+1} \mathcal{I}_{i,j})$ that satisfies $A'_{x,1,q} \geq \gamma > A'_{i,\tau_i+1,q}$. If not, the user considers the query result incorrect. The reason is that the pair $\langle x, A'_{x,1,q} \rangle$ should have been inserted by the data collector in one of $\{\mathcal{I}_{i,j}\}_{j=1}^{\tau_i+1}$ if $x \in \mathcal{M}_e$ and $A'_{x,1,q} > A'_{i,\tau_i+1,q}$. If x is also in $\mathcal{I}$ and $A'_{x,1,q} \geq \gamma$, we have $\tau_x \geq 1$, so the LBSP should have returned a valid $\mathcal{S}_x$ for zone x corresponding to Case 1 or 2.

- If such zone i does not exist, the user checks if the query result contains $\mathcal{S}_j = \langle j, \phi_{j,1}, h_{j,2} \rangle = \langle j, l_{j,1}, \mathcal{I}_{j,1}, A'_{j,1,q}, H(\mathcal{I}_{j,1} || D'_{j,1}) \rangle$ with $A'_{j,1,q} < \gamma$ for $j \in \mathcal{I} \bigcap \mathcal{M}_e$, which corresponds to the case of $\tau_i = 0$ for all $i \in \mathcal{M}'_e = \{i | i \in \mathcal{M}_e \bigcap \mathcal{I}, \tau_i < n_i, n_i \neq 0\}$. If so, for every zone $x \in \mathcal{I} \bigcap \mathcal{M}_e \bigcap \mathcal{I}_{j,1}$ with $A'_{x,1,q} \geq \gamma > A'_{j,1,q}$, the user checks whether the query result contains a valid $\mathcal{S}_x$ corresponding to Case 1 or 2 in query processing. If not, the query result is considered incorrect. Note that this verification implicitly ensures the compliance with Case 4 in query processing, i.e., that the LBSP only returns the information for the highest attribute-q rating in $\mathcal{M}'_e$.

- If such zone j does not exist either, it must be true that $n_i = \tau_i$ for all $i \in \mathcal{I} \bigcap \mathcal{M}_e$ and that there is no attribute-q rating in zones $\mathcal{I} \bigcap \mathcal{M}_e$ smaller than γ. The user verifies this by checking if $n_i = 0$ or $n_i = \tau_i > 0$ for each zone $i \in \mathcal{I} \bigcap \mathcal{M}_e$ according to the corresponding field $\mathcal{S}_x$. If not, the user considers the query result incorrect.

If the query result pass all the above verifications, the user considers it both authentic and correct.

3.4.3.4 An Example

We continue with the example in Fig. 3.2, where we assume that zones 1 to 3 compose a macro zone. Unlike in Scheme 1, the LBSP need not return any information for

zone 3, which has been embedded into the query result along with the information from zones 1 and 2. More specifically, we can see that the highest POI rating $A'_{3,1}$ in zone 3 satisfies $A'_{1,3} > A'_{3,1} > A'_{1,4}$ and $A'_{2,2} > A'_{3,1} > A'_{2,3}$. Therefore, $\langle 3, A'_{3,1} \rangle$ must have been embedded into $\mathcal{I}_{1,4}$ and also $\mathcal{I}_{2,3}$, so there is no need to include $\langle 3, A'_{3,1}, \mathcal{T}_3 \rangle$ in the query result. After verifying the query result, the user can find that no POI in zone 3 has a rating higher than γ.

3.4.4 Discussion

Thus far we have assumed that there are at least k POIs in the query region and that no POIs have the same rating for any attribute. This section discusses the impact on our schemes if these assumptions do not hold.

3.4.4.1 Insufficient POIs in the Query Region

If there are less than k POIs in the query region $\mathbf{R}$, any POI there satisfies the top-k query. Therefore, the LBSP need prove to the user that the query result contains every POI record in $\mathbf{R}$ by returning all the POIs in candidate zones $\mathcal{I}$. Take Scheme 1 as an example. On receiving a top-k query, the LBSP includes $\mathcal{S}_i = \langle i, X_{i,1}, \ldots, X_{i,n_i} \rangle$ for each zone $i \in \mathcal{I}$ in the query result, which the user can verify in the same way. Similar modifications can be made to Scheme 2 and are omitted here.

The impact of such scenarios on our schemes is limited. Consider Scheme 1 as an example, $X_{i,j}$ equals the index $\phi_{i,j}$ for POI$_{i,j}$ outside $\mathbf{R}$ and the record $D'_{i,j}$ for POI$_{i,j}$ in $\mathbf{R}$ according to Eq. (3.3), while an index contains much fewer bits than a data record. So the additional communication overhead incurred by returning the indexes of all the POIs in $\mathcal{I}$ but not in $\mathbf{R}$ is relatively very low. Moreover, such cases can be largely avoided in practice by putting an upper limit on k and/or a lower limit on the query-region size. For example, users are not allowed to submit a top-k query with k exceeding a certain threshold and/or too small a query region.

3.4.4.2 Multiple POIs with Equal Attribute Ratings

Due to the limited rating range, multiple POIs may have an equal rating for the same attribute. The tie can be easily broken by considering additional information for compar-

ing POIs. For example, we can add the time of the last review into the index $\phi_{i,j}$ of any POI$_{i,j}$ in Schemes 1 and 2. In case there are multiple POIs with equal ratings, the one with the most recent review is preferred. The impact of such scenarios on our schemes is thus negligible.

3.5 Secure Moving Top-k Query Processing

In this section, we propose a scheme to realize secure moving top-k query processing.

3.5.1 Basics of Moving Top-k Queries

A moving top-k query asks for the top-k POIs in a moving query region $\mathbf{R}$. For example, a user may want to find the k gas stations with the lowest gas price within 5 miles radius when driving a car. In this example, $\mathbf{R}$ is a changing circle of radius 5 miles centered at the user's current location.

One may think about securely processing a moving top-k query as a sequence of snapshot top-k queries. In particular, the mobile user submits a snapshot top-k query at a sufficiently high frequency which can be processed by the LBSP using Scheme 1 or 2. Since the query results for consecutive snapshot top-k queries may largely overlap, this naive solution may incur unnecessary communication and computation overhead. This observation motivates us to develop a more efficient solution to moving top-k queries.

3.5.2 Scheme 3

Our basic idea is to let the LBSP process consecutive snapshot top-k queries involved in a moving top-k query as a whole and only return a query result if there is any update in the top-k POIs satisfying the query. An update in the top-k POIs may occur when a current top-k POI is no longer in the moving query region or when a new POI appears in the moving query region, which has an attribute-q rating higher than the lowest among the current top-k POIs. The user can directly tell when the first situation occurs based on the current top-k POIs he knows, in which case he can issue a new snapshot top-k query for the current query region. The user, however, cannot tell when the second

situation will occur. Without a sound defense in place, the LBSP can choose not to inform the user about updated top-k POIs in the second situation.

Scheme 3 aims at the second situation discussed above and can be built upon either Scheme 1 or Scheme 2. Due to space constraints, we focus on Scheme 1 and assume that the data set has been preprocessed by the data collector accordingly, and the same design principles apply when Scheme 2 is chosen instead. Without loss of generality, we assume that a user issues a moving top-k query for attribute q during time period $[0, T]$, where T may be unknown in advance. Since a moving top-k query involves a sequence of snapshot top-k queries, we denote the ath snapshot top-k query by Q_a and the corresponding query region $\mathbf{R}_a$. We also let kPOI$_a$ be the top-k POIs in $\mathbf{R}_a$ and γ_a the lowest attribute-q rating among kPOI$_a$. In what follows, we detail the additional operations in Scheme 3 in contrast to Scheme 1, including *query scheduling*, *query processing*, and *query-result verification*.

3.5.2.1 Query Scheduling

To realize a moving top-k query, the user issues a sequence of snapshot top-k queries according to a query schedule. In particular, the user issues the ath snapshot top-k query (i.e., Q_a) at time

$$t_a = \begin{cases} 0 & \text{if } a = 1, \\ \min(t_{a-1} + \triangle t, t_u, T) & \text{o.w. }, \end{cases} \tag{3.6}$$

where $\triangle t$ is his personal parameter determining the lowest frequency at which snapshot queries are issued, and t_u denotes the time when the first POI in the current top-k POIs moves out of the query region. To be more clear, after receiving kPOI$_a$ from the LBSP in response to Q_a, the user sets a timer of length $\triangle t$. Then he issues Q_{a+1} when the timer fires or when the first POI in kPOI$_a$ is no longer in his moving query region, whichever comes first.

As before, Q_a includes the interested POI category, the interested attribute q, the current query region $\mathbf{R}_a$, and an integer $k \geq 1$. To facilitate query processing at the LBSP, it also includes both an integer id uniquely identifying this moving top-k query

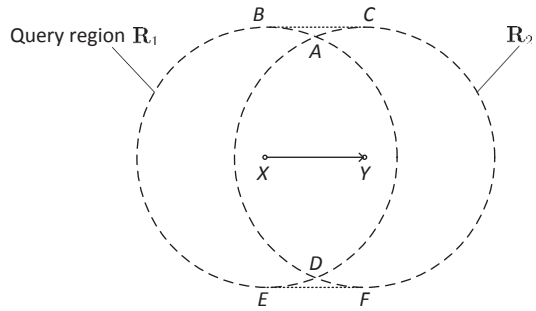


Figure 3.3: An example of two consecutive snapshot top-k queries.

and a one-bit flag indicating whether Q_a is the last snapshot query for this moving top-k query.

3.5.2.2 Query Processing

Assume that the LBSP has purchased the data set from the data collector as in under Scheme 1. It processes the sequence of snapshot top-k queries of the same moving top-k query as follows.

We first define a special region to ease our subsequent illustration. Consider two consecutive snapshot top-k queries Q_a and Q_{a+1} with query regions $\mathbf{R}_a$ and $\mathbf{R}_{a+1}$, respectively. Since the user's query region $\mathbf{R}$ is always defined with regard to his current location, we have $\mathbf{R} = \mathbf{R}_a$ at time t_a and $\mathbf{R} = \mathbf{R}_{a+1}$ at time t_{a+1}. We define the *progression region*, denoted by $\mathbf{P}_a$, as the area in $\mathbf{R}_{a+1}$ but not in $\mathbf{R}_a$. Consider Fig. 3.3 as an example where the user issues two consecutive snapshot top-k queries at locations X and Y with query regions $\mathbf{R}_1$ and $\mathbf{R}_2$, respectively. The progression region $\mathbf{P}_1$ is the area formed by arcs AD and $ACFD$.

On receiving query Q_1, the LBSP locates kPOI$_1$ in the query region $\mathbf{R}_1$ and then returns a complete query result constructed as in Scheme 1. In addition, the LBSP records id, $\mathbf{R}_1$, and kPOI$_1$ to facilitate the processing of subsequent snapshot top-k queries with the same moving top-k identifier id. Then it processes any subsequent query Q_b ($b > a$) as follows. Without loss of generality, assume that the last complete query result the LBSP returned is in response to Q_a ($a \geq 1$), which contains kPOI$_a$ in

$\mathbf{R}_a$. In other words, we assume that the top-k POIs $\{\text{kPOI}_a\}_{i=a}^{b-1}$ in the query regions $\{\mathbf{R}_i\}_{i=a}^{b-1}$ are all equal to kPOI$_a$.

First, the LBSP checks if a complete query result containing the top-k POIs (i.e., kPOI$_b$) in the current query region $\mathbf{R}_b$ need be returned by checking the following two conditions.

- kPOI$_b$ have different POIs from kPOI$_a$.
- The one-bit flag in Q_b is set, meaning that it is the last snapshot query for the current moving top-k query identified by the same id.

If neither condition holds, the LBSP returns a short ACK containing a predefined flag to the user, which means that the previously returned top-k POIs in kPOI$_a$ remain valid in the current query region $\mathbf{R}_b$. Otherwise, the LBSP constructs a complete query result as follows.

First, the LBSP locates the top-k POIs (i.e., kPOI$_b$) in the query region $\mathbf{R}_b$ whose attribute-q ratings are among the highest k. Second, the LBSP retrieves the recorded query regions $\{\mathbf{R}_x\}_{x=a}^{b}$ based on their same moving top-k identifier id, based on which to compute the progressive regions $\{\mathbf{P}_x\}_{x=a}^{b-1}$. Next, the LBSP computes a *verification region* as $\mathbf{V}_{a\to b} = \bigcup_{x=a}^{b-1} \mathbf{P}_x$ whereby to find the set of zones either completely or partially covered by $\mathbf{R}_b \bigcup \mathbf{V}_{a\to b}$, denoted by $\mathcal{I}_{a\to b}$.

Let γ_a and γ_b be the lowest attribute-q rating among kPOI$_a$ and kPOI$_b$, respectively. For each zone $i \in \mathcal{I}_{a\to b}$, we define

$$\tau_i = \begin{cases} \tau_{b,i} & \text{if zone } i \text{ only overlaps with } \mathbf{R}_b, \\ \tau_{a,i} & \text{if zone } i \text{ only overlaps with } \mathbf{V}_{a\to b}, \\ \max(\tau_{a,i}, \tau_{b,i}) & \text{if zone } i \text{ overlaps with both } \mathbf{R}_b \text{ and } \mathbf{V}_{a\to b}\ , \end{cases}$$

where $\tau_{a,i}$ and $\tau_{b,i}$ are the number of POIs in zone i with the attribute-q rating $\geq \gamma_a$ or

γ_b, respectively. We further define

$$Z_{i,j} = \begin{cases} D'_{i,j} & \text{if } l'_{i,j} \in \mathbf{R}_b \text{ and } A'_{i,j,q} \geq \gamma_b, \\ \phi_{i,j} & \text{otherwise} , \end{cases}$$

which means that the LBSP only needs to return a much shorter index instead of the complete record for any POI not in the query region $\mathbf{R}_b$ or not among the top-k. Similar as in Scheme 1, the LBSP finally returns the following information $\mathcal{S}_i$ for each zone $i \in \mathcal{I}_{a \to b}$ as part of the query result.

- Case 1: if $n_i = 0$, $\mathcal{S}_i = \langle i \rangle$.
- Case 2: if $n_i = 1$, $\mathcal{S}_i = \langle i, Z_{i,1} \rangle$.
- Case 3: if $n_i \geq 2$ and $\tau_i = 0$, $\mathcal{S}_i = \langle i, \phi_{i,1}, h_{i,2} \rangle$.
- Case 4: if $n_i \geq 2$ and $n_i > \tau_i \geq 1$,

$$\mathcal{S}_i = \langle i, Z_{i,1}, \ldots, Z_{i,\tau_i}, \phi_{i,\tau_i+1}, h_{i,\tau_i+2} \rangle \ .$$

- Case 5: if $n_i = \tau_i \geq 2$, $\mathcal{S}_i = \langle i, Z_{i,1}, \ldots, Z_{i,\tau_i} \rangle$.

In addition, the LBSP returns $\mathcal{T} = \bigcup_{i \in \mathcal{I}_{a \to b}} \mathcal{T}_i$ and the data collector's signature on the qth Merkle root hash.

3.5.2.3 Query-Result Verification

For every snapshot top-k query Q_b of the same moving top-k query, the LBSP (if benign) should return a complete query result if $b = 1$ or there has been any change in the top-k POIs, or return an ACK if $b > 1$ and the previously returned top-k POIs are still valid. Accordingly, there are three cases for the user to verify the query result in response to Q_b. First, if the user receives an ACK when Q_b is the final snapshot query, he can immediately tell that the result is incorrect. Second, if receiving an ACK when Q_b is not the final snapshot query, he marked this query result unverified and waits for the next complete query result. Third, if receiving a complete query result for Q_b (no matter whether Q_b is the final query), he verifies it as follows.

First, the user checks if the query result is authentic as in Scheme 1. If so, the user derives the set of zones $\mathcal{I}_{a\to b}$ from the POI information returned in the query result and checks if zones $\mathcal{I}_{a\to b}$ encloses the region $\mathbf{R}_b \bigcup \mathbf{V}_{a\to b}$. If so, he locates the lowest attribute-q rating γ_b in the query result whereby to check whether all the following conditions hold.

1. There are exactly k POI records in the query result.
2. Every returned POI record is in $\mathbf{R}_b$.
3. None of the POIs for which the indexes are returned satisfy the query. In particular, for each index $\phi_{i,j}, \forall i \in \mathcal{I}_{a\to b}$, at least one following condition does not hold.
   - $\phi_{i,j}$ contains a location $l'_{i,j} \in \mathbf{R}_b$.
   - $\phi_{i,j}$ contains an attribute rating $A'_{i,j,q} \geq \gamma_b$.

If so, the top-k POI records are correct.

Assume that the last complete query result the user verified is for Q_a and contains kPOI$_a$ in the region $\mathbf{R}_a$ and that $b > a+1$. The user should have accumulated $b-a-1$ unverified query results for queries $\{Q_x\}_{x=a+1}^{b-1}$ and can verify their correctness by checking whether the LBSP should have returned a complete query result instead of an ACK for each of them instead. Let γ_a again denote the lowest attribute-q rating in kPOI$_a$ and $\mathbf{S}_{a\to b} = \bigcup_{j=a}^{b-2} \mathbf{P}_j$ denote the *suspicion region*. If all the unverified query results are correct, there should not be any POI in $\mathbf{S}_{a\to b}$ with attribute-q rating higher than γ_a. According to the query-processing process, the LBSP should have returned one or multiple data indices for every zone i that overlaps with $\mathbf{S}_{a\to b}$; otherwise, the query result for Q_b would not have passed the verification. The user thus proceeds to check whether at least one following condition does not hold for any such index, say $\phi_{i,j}$.

- $\phi_{i,j}$ contains a location $l'_{i,j} \in \mathbf{S}_{a\to b}$.

- $\phi_{i,j}$ contains an attribute rating $A'_{i,j,q} > \gamma_a$.

If so, all the unverified query results are marked verified; otherwise, the LBSP has misbehaved.

3.6 Performance Analysis

In this section, we analyze Schemes 1∼3 with regard to their correctness in detecting inauthentic and/or incorrect query results and the related communication/computation overhead. To make the quantitative analysis tractable, we make the following assumptions.

- There are $n > k$ POIs uniformly distributed in each zone, i.e., $n_i = n, \forall i \in [1, M]$, where $M = 2^d$ for an integer $d > 1$.
- All attribute ratings are i.i.d. random variables uniformly distributed in the range $[0, 1]$ after proper normalization.
- The query-region size is δ times of the zone size.

3.6.1 Analysis of Scheme 1

The following theorem is for the correctness of Scheme 1.

Theorem 3.6.1. Scheme 1 can detect any incorrect and/or inauthentic query result from a misbehaving LBSP.

We give the proof in Appendix C.

The main extra computation overhead incurred by Scheme 1 on top-k query processing involves hash computations and signature generations/verifications. Consider the data collector first. For every zone $i \in [1, M]$ and every attribute, the data collector performs n hash computations to generate the indexes $\{\phi_{i,j}\}_{j=1}^{n}$ and n hash computations to derive $h_{i,1}$, which leads to totally $2Mn$ hash computations. In addition, the data collector needs $M - 1$ hash computations to construct the Merkle hash tree of every attribute and one signature generation for the root hash. Since there are q

POI attributes, the total computation overhead per POI category at the data collector is $\lambda(2Mn + M - 1)$ hash computations and λ signatures. Moreover, the computation overhead at the LBSP is negligible because the LBSP need not perform any hash or signature operations for query processing.[2] Finally, we consider the computation overhead at the user. For every query result, the user needs one signature verification for the Merkle root hash and also a certain number of hash computations given below.

Theorem 3.6.2. The expected number of hash computations the user performs to verify the query result under Scheme 1 is given by

$$\mathsf{E}[N_{hash,1}] = k + |\mathcal{I}| \cdot \frac{(k+\delta)n + 1}{\delta n + 1} + \sum_{j=1}^{d-1} 2^{j-1}(1 - (1 - 2^{-(j-1)})^{|\mathcal{I}|}) \ . \tag{3.7}$$

We give the proof in Appendix D.

Now we analyze the communication overhead associated with transmitting the necessary information for authenticity and correctness proofs from the data collector to the LBSP. Let L_{h}, L_{loc}, L_{r}, and L_{sig} denote the bit-lengths of a hash value $H(\cdot)$, a POI location, an attribute rating, and the data collector's signature, respectively. For each of λ POI attributes, the data collector sends n indexes of $L_{\mathrm{loc}} + L_{\mathrm{r}} + L_{\mathrm{h}}$ bits for each of M zones as well as a Merkle hash tree of $(M-1)L_{\mathrm{h}}$ bits. The extra communication overhead in bits per POI category Scheme 1 incurs between the data collector and LBSP is thus

$$\mathsf{S}_1 = \lambda(Mn(L_{\mathrm{loc}} + L_{\mathrm{r}} + L_{\mathrm{h}}) + (M-1)L_{\mathrm{h}} + L_{\mathrm{sig}}). \tag{3.8}$$

We also have the following theorem about the extra communication overhead associated with sending authenticity and correctness proofs of a top-k query result from the LBSP to the user.

[2] Here we ignore the LBSP's database lookup overhead which exists with or without our scheme.

Theorem 3.6.3. The additional communication overhead between the LBSP and the user incurred by Scheme 1 is given by

$$\begin{aligned}\mathsf{E}[\mathsf{T}_1] = (|\mathcal{I}| \cdot \frac{(k+\delta)n+1}{\delta n+1} - k)(L_{\mathrm{loc}} + L_{\mathrm{r}} + L_{\mathrm{h}}) + |\mathcal{I}| \cdot d \\ + \sum_{j=1}^{d-1} 2^j (1-(1-2^{-j})^{|\mathcal{I}|}) L_{\mathrm{h}} + L_{\mathrm{sig}} \ ,\end{aligned} \tag{3.9}$$

We give the proof in Appendix E.

3.6.2 Analysis of Scheme 2

The following theorem is for the correctness of Scheme 2.

Theorem 3.6.4. Scheme 2 can detect any incorrect and/or inauthentic query result from a misbehaving LBSP.

We give the proof in Appendix F.

Scheme 2 incurs the same computation overhead to the data collector and LBSP as Scheme 1, which has been analyzed before. To verify the authenticity and correctness of a top-k query result, the user performs one signature verification on the Merkle root hash and also a certain number of hash computations given in the following theorem.

Theorem 3.6.5. The expected number of hash computations the user performs to verify the query result under Scheme 2 is given by

$$\mathsf{E}[N_{hash,2}] = |\mathcal{I}|\mu_1 + \sum_{j=1}^{d-1} 2^{j-1}(1-(1-2^{-(j-1)})^{|\mathcal{I}|(1-\mu_2^n)}) \ , \tag{3.10}$$

where $\mu_1 = (n - n\mu_2 + 1 - \mu_2^n)$ and $\mu_2 = \frac{\delta n - k + 1}{\delta n + 1}$.

We give the proof in Appendix G.

Now we analyze the communication overhead incurred by Scheme 2. In Scheme 2, every zone belongs to a macro zone of m zones. For every zone i in a macro zone $\mathcal{M}_e$, the set $\{j, A'_{j,1,q}\}_{j \in \mathcal{M}_e \setminus \{i\}}$ need be transmitted along with both POI records and

indexes. Since a zone ID is of $\log_2 M = d$ bits, Scheme 2 requires the data collector to additionally transmit $2(m-1)(d+L_{\mathrm{r}})$ bits for attribute q in contrast to Scheme 1. The extra communication overhead per POI category Scheme 2 incurs between the data collector and LBSP is thus

$$\mathsf{S}_2 = \mathsf{S}_1 + 2(m-1)\lambda(d+L_{\mathrm{r}}) \ , \tag{3.11}$$

where S_1 is given in Eq. (3.8). We also have the following theorem about the communication overhead for sending authenticity and correctness proofs of a query result from the LBSP to the user.

Theorem 3.6.6. Assuming that the query region comprises $\check{m}$ zones $\mathcal{I}$ fully contained in a macro zone $\mathcal{M}_e$ with m zones. The expected additional communication overhead Scheme 2 incurs between the LBSP and user is bounded as follows,

$$\begin{aligned}\mathsf{T}_2 \leq\ & \check{m}(1-\mu^n)d + \check{m}(n-n\mu+1-\mu^n)(L_{\mathrm{loc}}+L_{\mathrm{r}}+L_{\mathrm{h}}) \\ & + (\check{m}(1-\mu^n) + \sum_{j=1}^{d-1} 2^j(1-(1-2^{-j})^{\check{m}(1-\mu^n)}))L_{\mathrm{h}} \\ & + \check{m}(1-\mu^n)(m-\check{m})(1-(\frac{n-\nu}{n+1})^n)(d+L_{\mathrm{r}}) \\ & + g(g-1)(d+L_{\mathrm{r}}) + L_{\mathrm{sig}} \ ,\end{aligned} \tag{3.12}$$

where $\mu = (\check{m}n-k+1)/(\check{m}n+1), \nu = n(1-\mu)/(1-\mu^n)$, and $g = \min(k, \check{m})$.

We give the proof in Appendix H. We have not been able to obtain a close-form solution for the more general case, which we will evaluate using simulation in the next section.

3.6.3 Analysis of Scheme 3

The following theorem is for the correctness of the Scheme 3.

Theorem 3.6.7. Any misbehavior of the LBSP, including returning incorrect/inauthentic query result and omitting complete query results, will be eventually detected under Scheme 3.

Table 3.1: Default simulation settings

Para.	Val.	Para.	Val.	Para.	Val.	Para.	Val.
M	10000	m	100	n	100	δ	10
k	5	d	14	d	20	L_{h}	160
L_{loc}	20	L_{sig}	160	L_{r}	10		

We give the proof in Appendix I. We will use simulation to evaluate the communication and computation overhead incurred by Scheme 3 in the next section.

3.7 Simulation Results

In this section, we evaluate our schemes using simulations. We assume that the data set covers 100×100 unit square zones of 1000×1000m^2, each containing 100 POIs uniformly distributed. The simulation code is written in C++, and each data point represents an average of 50 simulation runs with different random seeds. In addition, our simulations use the default parameters in Table 3.1, unless stated otherwise.

3.7.1 Snapshot Top-k Queries

We first report the simulation results for Schemes 1 and 2. Recall that δ denote the ratio of the query-region size to the zone size and that $\mathcal{I}$ represent the set of candidate zones that completely or partially overlap with the query region $\mathbf{R}$. We simulate the following two types of queries.

- **Type-1 queries**: $\mathbf{R}$ exactly covers an integer number of zones, which means that $\mathcal{I} = \mathbf{R}$ and $|\mathcal{I}| = \delta$.
- **Type-2 queries**: $\mathbf{R}$ is a circle of radius r centered at a random location, which means that $\mathcal{I} > \mathbf{R}$ and $|\mathcal{I}| > \delta$.

3.7.1.1 Type-1 Queries: $|\mathcal{I}| = \delta$

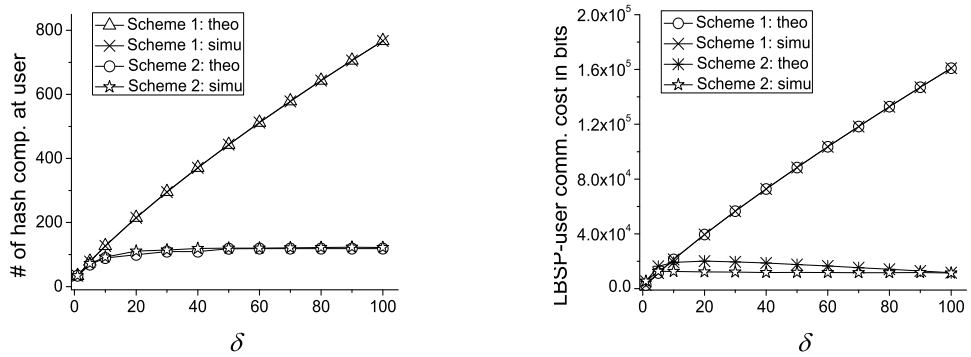
For this set of simulations, we let the query region $\mathbf{R}$ formed by δ zones randomly chosen from the same macro zone.

Fig. 3.4a shows the impact of δ on the user's computation overhead for $k = 5$, where the single signature verification is not included for brevity. Clearly, our analytical

and simulation results closely match under both schemes. In addition, the user's computation overhead increases with δ under Scheme 1, while it initially increases as δ goes from 1 to 10 and then is relatively stable under Scheme 2. The reason is that Scheme 1 requires the LBSP to return information for every zone in $\mathbf{R}$ for the user to verify. Therefore, the larger δ, the higher the user's computation overhead in Scheme 1. In contrast, Scheme 2 requires the LBSP to return information only for the zones that have at least one POI among the top-k POIs under our simulation settings, and there are at most k such zones in $\mathbf{R}$. Therefore, Scheme 2 incurs lower computation overhead on the user for small k and large δ.

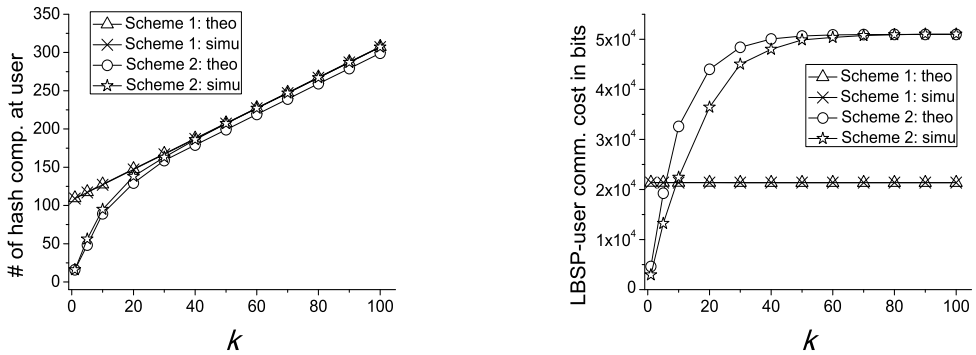
Fig. 3.4b shows the impact of δ on the LBSP-user communication overhead for $k = 5$. It is clear that the simulation results are always below the corresponding theoretical upper bounds. As in Fig. 3.4a, we can also observe that the LBSP-user communication overhead in Scheme 1 always increases with δ and is higher than that in Scheme 2. In contrast, the LBSP-user communication overhead under Scheme 2 is relatively stable and even slightly decreases when δ grows. The reason is that the kth largest attribute rating becomes large as δ increases, which means that the query result contains less information for other zones in the same macro zone with attribute ratings higher than any top-k rating.

Fig. 3.5a shows the impact of k on the user's computation overhead for $\delta = 10$. We can see that our simulation and analytical results closely match and increase with k under both schemes. The reason is that the number of hash computations increases with the number of zones with information in the query result, which itself increases with k. In addition, since Scheme 2 does not require the LBSP to return any information for zones without a top-k POI, it requires the user to perform fewer hash computations and thus incurs smaller computation overhead than Scheme 1. The difference between the two schemes gradually diminishes when k goes beyond 20, as the number of zones in $\mathbf{R}$ without a top-k POI quickly decreases for sufficiently large k.



(a) computation cost (b) communication cost

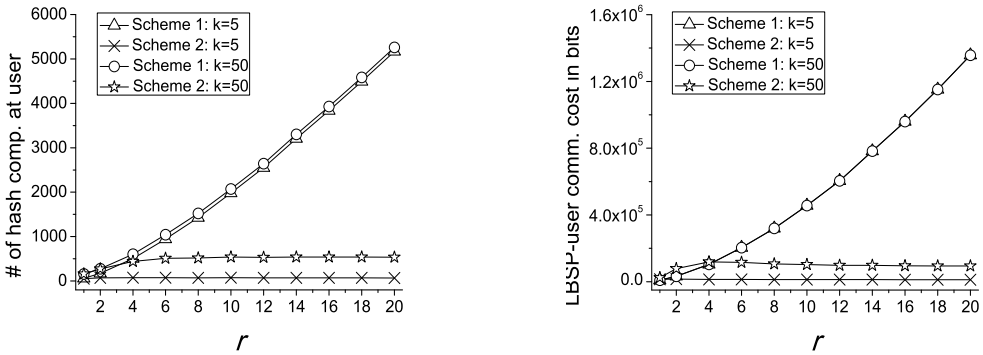
Figure 3.4: The impact of δ for Type-1 queries, where $k = 5$.



(a) computation cost (b) communication cost

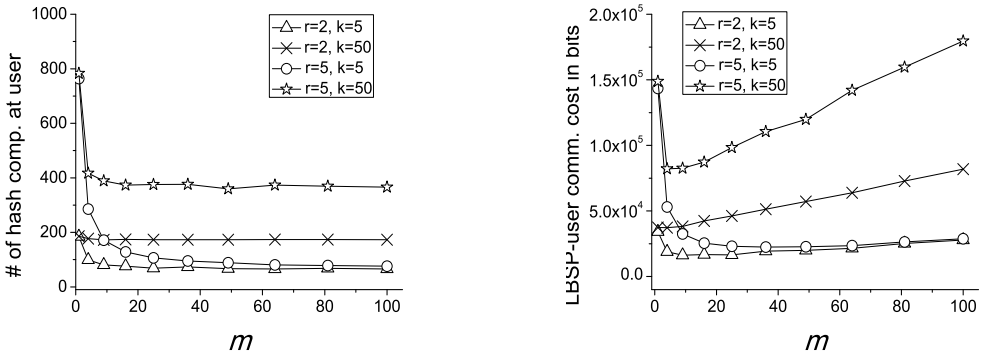
Figure 3.5: The impact of k for Type-1 queries, where $\delta = 10$.

Fig. 3.5b shows the impact of k on the LBSP-user communication overhead for $\delta = 10$. Again, our simulation and analytical results closely match. In addition, the LBSP-user communication overhead of Scheme 1 is not affected by k because it only involves transmitting $|\mathcal{I}| = \delta$ POI indexes. In contrast, the LBSP-user communication overhead of Scheme 2 always increases with k, as the number of POI records or indexes increases with k, and accordingly the information about other zones in the same macro zone returned along with every POI record or index also increases.



(a) computation cost (b) communication cost

Figure 3.6: The impact of query radius r for Type-2 queries.



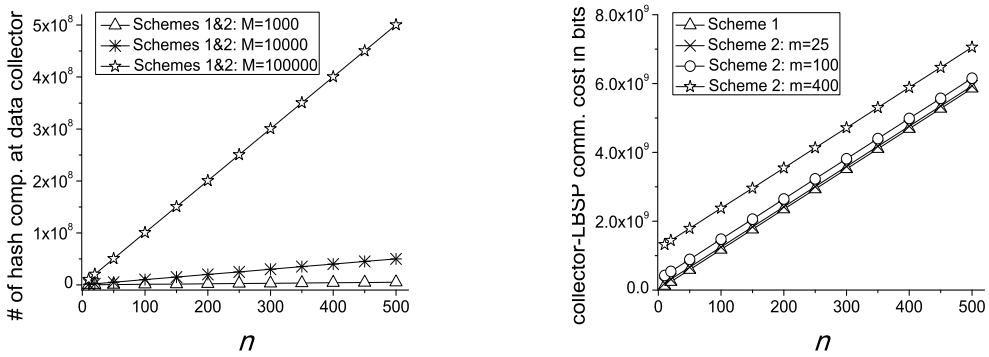
(a) computation cost (b) communication cost

Figure 3.7: The impact of m on Scheme 2.

3.7.1.2 Type-2 Queries: $|\mathcal{I}| > \delta$

For this set of simulations, we simulate a circular query region with radius r centered at a random location and only report the simulation results for simplicity.

Figs. 3.6a and 3.6b show the impact of query radius r on the user's computation overhead and the LBSP-user communication overhead, respectively, for $k = 5$ or 50. Note that $\delta = \pi r^2$ increases quadratically with r, so does the number of candidate zones. It is thus not surprising to see that the user's computation overhead and the LBSP-user communication overhead both increase as r increases under Scheme 1.



(a) computation cost (b) communication cost

Figure 3.8: The impact of n.

In contrast, both metrics are relatively insensitive to r under Scheme 2 because the number of zones having at least one top-k POI is at most k.

3.7.1.3 Impact of m on Scheme 2

Now we illustrate the impact of m, the number of zones in each macro zone, on Scheme 2. For simplicity, we show the simulation results for Type-2 queries only.

Fig. 3.7a shows that the user's computation overhead decreases rapidly as m increases from 1 to 10 and slowly as m further increases. The reason is that the LBSP returns only one index and the corresponding auxiliary set for each candidate macro zone that has no top-k POI. When k is small and $\mathbf{R}$ is large, most zones in $\mathbf{R}$ do not have any top-k POI, so the number of indexes and auxiliary sets returned is approximately proportional to the number of macro zones and thus inversely proportional to m when m is not too large. Otherwise, the number of macro zones overlapping with $\mathbf{R}$ approaches a constant, leading to relatively stable computation overhead.

Fig. 3.7b shows that the LBSP-user communication overhead quickly decreases as m increases from 1 to 10. The reason is that the larger m, the fewer POIs and corresponding auxiliary sets returned to the user. As m further increases, the communication overhead slowly increases, as a larger m requires the LBSP to return more information about other zones in the same macro zone along with every POI record or

index in the query result. In practice, a moderate m should be chosen to minimize the LBSP-user communication overhead.

3.7.1.4 Impact of n on Schemes 1 and 2

Figs. 3.8a and 3.8b show the impact of n, the number of POIs per zone, on the data collector's computation overhead and the collector-LBSP communication overhead. For brevity, we only show the simulation results which apply to both Type-1 or Type-2 queries. Fig. 3.8a shows that the data collector's computation overhead increases linearly with n under both schemes. The reason is that the data collector performs one hash computation to generate the index and chain it with adjacent indexes for each POI record in both schemes. Moreover, as anticipated, the larger M, the more POIs, and the higher the computation overhead. In addition, Fig. 3.8b shows that the collector-LBSP communication overhead under both schemes increases with n, and Scheme 2 incurs larger overhead because it requires additional information for other zones in the same macro zone to be transmitted for each POI record.

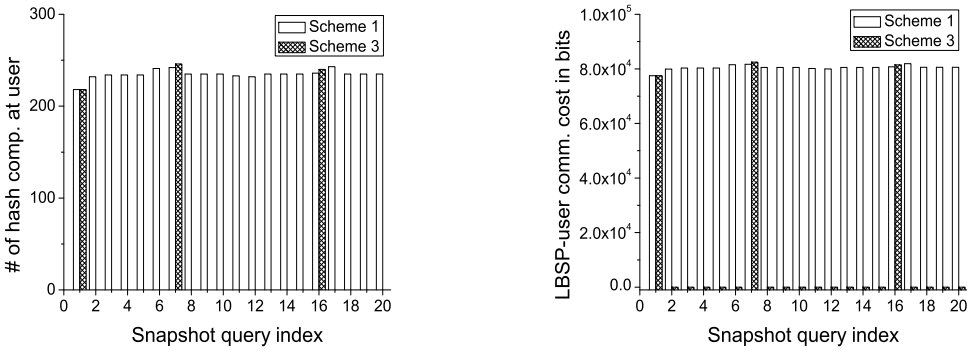
We have also simulated the impact of λ, the number of POI attributes, and observed that the data collector's computation overhead and the collector-LBSP communication overhead are both proportional to λ under both schemes.

3.7.2 Moving Top-k Queries

In this subsection, we report the simulation results for Scheme 3. In particular, we compare Scheme 3 with realizing moving top-k query via independent snapshot queries under Scheme 1. We simulate a moving top-k query in which the query region is a circular area of radius $r = 5000$m centered at the user's location. The user starts at a random location along a random direction, moves at a speed of 5m/s for a total distance of 5000m.

3.7.2.1 General Comparison between Schemes 1 and 3

Figs. 3.9a and 3.9b show the user's computation overhead and LBSP-user communication overhead incurred by the first 20 snapshot top-k queries under Schemes 1 and 3, respectively, where $\triangle t = 20$s. We can see that both schemes incur the same user-side



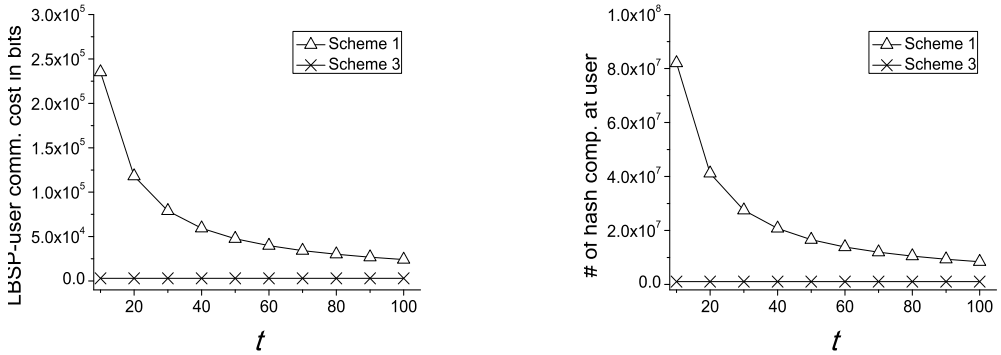
(a) computation cost (b) communication cost

Figure 3.9: Comparison of the first 20 snapshot queries in Schemes 1 and 3.

computation overhead and LBSP-user communication overhead for the first snapshot top-k query, as the LBSP need return a complete query result in both cases. Under Scheme 1, each snapshot query incurs similar computation and communication costs, while under Scheme 3, all the snapshot queries (except the 1st, 7th, and 16th) incur negligible user-side computation overhead and LBSP-user communication overhead. This is anticipated, as the LBSP always need return a complete query result for any snapshot query under Scheme 1 but does so only when there is an update in the top-k POIs from the previous ones under Scheme 3. It is also worth noticing that Scheme 3 incurs slightly higher user-side computation overhead and LBSP-user communication overhead for the 7th and 16th snapshot queries. This is because that the LBSP need provide additional information in the query response to prove that all previous returned ACKs are valid.

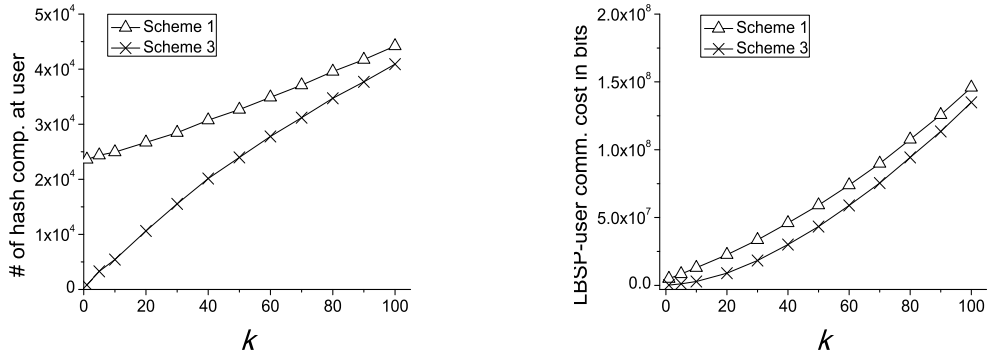
3.7.2.2 Impact of $\triangle t$

Figs. 3.10a and 3.10b compare Schemes 1 and 3 when $\triangle t$ varies. We can see that the total computation and communication cost incurred by Scheme 3 are relatively insensitive to the change in $\triangle t$, as no matter how frequently the user issues snapshot top-k queries, the LBSP only need return a complete query result when there is an update in the top-k POIs. In contrast, the total computation and communication costs incurred by Scheme 3 are inversely proportional to $\triangle t$, since the LBSP treats each snapshot query



(a) computation cost (b) communication cost

Figure 3.10: The impact of $\triangle t$ on Scheme 3.



(a) computation cost (b) communication cost

Figure 3.11: The impact of k on Scheme 3.

independently by always returning a complete query result. These results demonstrate the significant advantage of Scheme 3 over Scheme 1.

3.7.2.3 Impact of k

Figs. 3.11a and 3.11b compare Schemes 1 and 3 when k varies. We can see that the user-side computation overhead and LBSP-user communication overhead both increase as k increases under both schemes. This is because that the larger k, the more updates in the top-k POIs for the same distance that the user travels, and vice versa. Under both schemes, the LBSP need return more complete query results, which lead to higher user-side computation overhead and LBSP-user communication overhead.

When k is small, Scheme 3 incurs significantly lower user-side computation overhead and LBSP-user communication overhead than Scheme 1 does. For example, when $k = 1$ and 5, the LBSP-user communication overhead incurred by Scheme 3 is only 3.5% and 13.5% that of Scheme 1, respectively. As k increases, the benefits of using Scheme 3 gradually diminish, as the LBSP need return more complete query results under both schemes.

3.8 Summary

This chapter considers a novel distributed system for collaborative location-based information generation and sharing. We have proposed three novel schemes to enable secure top-k query processing via untrusted LBSPs for fostering the practical deployment and wide use of the envisioned system. Our schemes support both snapshot and moving top-k queries, which enable users to verify the authenticity and correctness of any top-k query result. The efficacy and efficiency of our schemes are thoroughly analyzed and evaluated through detailed simulation studies.

Chapter 4

VERIFIABLE PRIVACY-PRESERVING AGGREGATION IN PEOPLE-CENTRIC URBAN SENSING SYSTEMS

4.1 Introduction

People-centric urban sensing systems (PC-USSs) refer to using human-carried mobile devices such as smartphones and tablets with ever-growing capabilities in sensing, computation, storage, and communications for urban-scale distributed data collection, analysis, and sharing to facilitate the interaction between humans and their surrounding environments. Examples of PC-USS applications include environment monitoring [31, 86], traffic measuring and congestion avoidance [70, 108], healthcare monitoring and delivery [61], and many others [3, 8, 17, 33, 48, 55, 56, 68, 77, 85, 109]. PC-USSs are expected to open a new era of exciting scientific, social, and commercial applications.

PC-USSs differ significantly from traditional wireless sensor networks that focus on environment sensing and data collection. First, system devices are no longer owned and managed by a single authority but belong to individuals with diverse interests. Second, system devices have much more powerful resources than sensor nodes and can be charged regularly. Third, the system features dynamic node mobility. Fourth, sensing data are more related to the interactions among humans and between humans and their surroundings instead of only about some physical phenomena of interest. Fifth, but not the last, humans are no longer just passive data users but also active data contributors.

The widespread deployment and adoption of PC-USSs face many obstacles, of which user privacy and data integrity are among the most critical [17, 33, 56]. For instance, in a study of the relationship between air quality and public health, researchers desire some aggregate statistics of personal health data such as heart rates, blood pressure levels, and weights at different sections of an urban area. Individuals may be unwilling to disclose their personal data if there were no guarantee that their data would not be used to invade their privacy. As an example for data-integrity breach, consider

applications like CarTel [48] and VTrack [108] that use traffic statistics such as average speed as an indicator of congestion to help system users do route planning. A selfish and malicious driver may prevent other users from choosing his current road by manipulating the aggregation result, i.e., cheating the server into accepting a lower-than-actual average speed that indicates road congestion. These two examples highlight the necessity for verifiable privacy-preserving data aggregation techniques that can ensure strong user privacy and also aggregation integrity.[1]

Designing a verifiable privacy-preserving aggregation scheme for PC-USSs is particulary challenging. On the one hand, ensuring user privacy means that a user's original data cannot be disclosed to any other party. This requirement makes it hard to detect if a user has faithfully participated in data aggregation. On the other hand, ensuring aggregation integrity requires any misbehavior during data aggregation to be detected with overwhelming probability. This requirement is extremely difficult to satisfy without knowing users' original data.

The contribution of this chapter is the design and evaluation of VPA, a novel peer-to-peer based solution to verifiable privacy-preserving data aggregation in PC-USSs. VPA consists of the following two components.

- The first component VPA$^+$ aims at additive aggregation functions such as Sum, Average, and Variance. Its basic idea is to divide the aggregation process into two phases. In the first phase, each node submits a commitment to the aggregation server, which is a homomorphic message authentication code of its original data. The homomorphic property of commitments enables the aggregation server to compute the aggregate commitment corresponding to the final aggregate, while it is impossible for the aggregation server to recover any node's original datum. In the second phase, the original datum of each node is aggregated in a privacy-preserving manner, in which users first exchange random shares of their

[1] We use "user privacy" and "data privacy" as well as "aggregation integrity" and "data integrity" interchangeably in this chapter.

data with selected peers and then submit mixed data to the aggregation server. The aggregation server can then verify the aggregation-result integrity using the aggregate commitment derived in the first phase.

- The second component VPA$^{\oplus}$ is a non-trivial combination of the binary search and verifiable privacy-preserving Count queries and can support a wide range of non-additive aggregation functions like Max/Min, Median, Histogram, and Percentile, with accurate aggregation-results.

VPA is the first work of its kind as far as we know. The performance of VPA is thoroughly analyzed and evaluated with detailed simulations.

The rest of this chapter is organized as follows. Section 4.2 reviews the related work. Section 4.3 gives the system and adversary models and the design objectives. Section 4.4 and Section 4.5 present the solutions to additive and non-additive aggregation, respectively. Section 4.6 evaluates the performance of VPA using extensive simulation results. This chapter is finally concluded in Section 4.7.

4.2 Related Work

Although PC-USSs, also known as participatory or opportunistic sensing systems, have received extensive attention (e.g., [3, 8, 17, 33, 48, 55, 56, 68, 77, 85, 86, 109]), there is relatively little work focusing on their security and privacy aspects. Kapadia *et al.* [56] surveyed the security and privacy challenges in opportunistic sensing systems. Cornelius *et al.* [17] presented the AnonySense architecture for anonymous tasking and reporting in people-centric sensing systems. AnonySense relies on a Mix network like Minimaster [76] to ensure user privacy, which we will not assume in our scheme. Ganti *et al.* [33] proposed PoolView for computing community statistics of time-series data in a privacy-preserving manner without considering aggregation-result integrity. More recently, Cristofaro and Soriente [18] proposed PEPSI to protect data and query privacy from unauthorized subscribers. None of these schemes could achieve the same objectives as our VPA.

Privacy-preserving aggregation in traditional sensor networks has been extensively studied. The work [11,29,37,44] can support additive aggregation functions such as Sum and Average. GP^{2}S [136] can support both additive aggregation functions and non-additive ones such as Max/Min, Median, and Histogram at the sacrifice in data accuracy. The work [114] applies a particular class of encryption transformations to compute two aggregation functions, Average and "movement detection" specific to sensor networks. These schemes [11, 29, 37, 44, 114, 136] do not address aggregation-result integrity, neither could be be directly applied to PC-USSs due to different application scenarios.

There is also a big chunk of work on secure aggregation in sensor networks, see [12,13,90,93,94,118,126] for example. Such work ensures that aggregation results are not so different from the true values despite malicious intermediate aggregator nodes and does not address individual nodes' data privacy.

To the best of our knowledge, the work in [112] is the only one that simultaneously addresses data confidentiality and aggregation-result integrity. VPA differs from [112] significantly in following aspects. First, the scheme proposed in [112] targets histogram aggregates in traditional sensor networks with static topology, while VPA can support a large family of aggregates, including Sum, Average, Max/Min, Median, Histogram, and Percentile. Second, the scheme proposed in [112] can only detect ill-performed aggregation with some probability and protect users' data privacy against other users. In contrast, VPA can detect any false aggregation result with certainty and ensure user data confidentiality against both curious users and aggregation servers.

Finally, location/identity privacy of mobile users is another active topic of research, see [7, 32, 137] for example. This line of work is orthogonal to our work in this chapter and can be integrated with our VPA.

4.3 Models and Design Goals

In this section, we present the system and adversary models as well as our design goals.

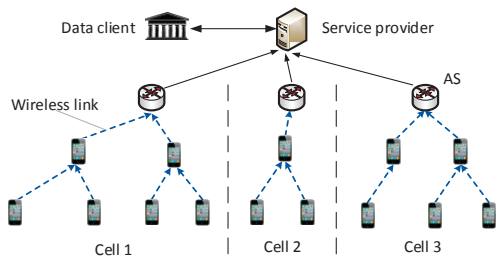


Figure 4.1: The abstract architecture of a people-centric urban sensing system (PC-USS).

4.3.1 System Model

There is no universally accepted model for a PC-USS. For ease of illustration, we assume an urban-sensing service provider which deploys a large-scale system similar to a metro-scale wireless mesh network [137], as shown in Fig. 4.1. Our solution can be easily extended to work with other system models such as cellular networks. The PC-USS features a high-speed wireless backbone consisting of M powerful aggregation servers (ASs for short) which also provide network access services for system nodes. Each AS is in charge of a certain region referred to as a *cell* and interacts with nodes therein. Here we use the term "node" to indicate a human who carries a portable device such as a smartphone and tablet. The devices have different communication and computation capabilities as well as various embedded sensors such as accelerator, digital compass, proximity sensors, and humidity sensors [68].

A node may join the system at will to participate in data sensing and sharing and also enjoy network access. To prevent fraudulent use of system resources and also provide basic privacy assurance to nodes, the system and nodes need mutually authenticate each other each time a node moves into a new cell.

We assume a similar mutual authentication protocol as in [137]. Assume that an AS, denoted by $\mathcal{A}$, can simultaneously accommodate up to 2^λ users. After achieving mutual authentication with a node, say i, $\mathcal{A}$ assigns node i a secret key k_i, a temporal integer-valued ID ID_i (which is an unused one between $[0, 2^\lambda - 1]$), and also an ID-

based private key K_i^{-1}. The pair ID_i/K_i^{-1} will serve as the temporal public/private keys of node i which are valid only in $\mathcal{A}$'s cell.

In addition, we assume an efficient method for $\mathcal{A}$ to keep track of node mobility in its cell. For example, node i need periodically notify $\mathcal{A}$ about its existence; otherwise, $\mathcal{A}$ would assume that i has left its cell and then reclaim ID_i to be allocated to new nodes. In the latter case, $\mathcal{A}$ updates all the private keys of the remaining nodes in its cell using a single broadcast message with the approach in [139]. The use of such ID-based public/private keys will be illustrated soon. Note that the mutual authentication process is performed whenever a node enters a new cell.

About the communication capabilities, we assume that each node and the AS can directly communicate with each other. In addition, each can communicate with neighboring nodes through WiFi or Bluetooth interfaces, for which very efficient protocols are available such as in [120]. Moreover, each node can transmit to the AS in a multi-hop fashion through other nodes if necessary.

Without loss of generality, we consider the following scenario throughout. We assume that the service provider, on behalf of a data client, wants to get statistical aggregates of some personal data such as heart rates, blood pressure levels, glucose levels, weights, and moving speeds. A query will be sent to selected ASs which in turn broadcast the query to the nodes inside their respective cells. If some nodes have data satisfying the query, they will participate in aggregation if provided with privacy guarantees. The ASs can then aggregate the returned data and forward the aggregation result to the service provider. The service provider often need provide extra incentives such as credits to motivate participation in data aggregation, but the design of sound incentives is outside the scope of this chapter.

4.3.2 Adversary Model

This chapter focuses on thwarting attacks on breaching nodes' data privacy as well as aggregation-result integrity. Other important issues such as DoS defenses [56] are beyond the scope of this chapter.

We assume that ASs are trusted to follow aggregation operations for generating correct aggregation results, but they may be curious about individual user data. A curious AS may collude with other curious nodes to attempt deducing the sensing data of target nodes.

In contrast, a node could be curious, malicious, or both. Like a curious AS, a curious node is interested in discovering other nodes' data but faithfully follow aggregation operations, while a malicious node intends to make the AS derive false aggregation result. More specifically, a malicious node may launch two types of *false-data injection* attacks [118]. First, a malicious node may forge its own datum. Second, a malicious node may forge a false intermediate aggregation result that could significantly affect the final aggregation result. Most recent research [24, 35, 36, 95] has shown that perhaps the only feasible defense against the former (i.e., ensuring the integrity of sensor readings from human-carried mobile devices) is to use some trusted hardware such as a Trusted Platform Module (TPM). We thus follow this line of research and assume that every participating mobile device has an embedded TPM. To keep the TPM cost as low as possible, we only require the TPM to have a minimal set of functionalities during aggregation, which include collecting sensor readings and generating a message authentication code (MAC). For this purpose, we assume that every TPM has a unique public/private key pair bound to the affiliated mobile device. After achieving mutual authentication with node i, the AS sends another secret key κ_i encrypted with the public key of node i's TPM. The TPM can then decrypt the ciphertext using its private key and store κ_i for later use. Based on the assumption about TPM, we focus on mitigating the forgery of intermediate aggregation results in our solution.

There might also be external eavesdroppers not participating in data aggregation. Since external eavesdroppers are fairly easy to defeat using end-to-end encryption (which we will use), we focus on counteracting internal attackers hereafter, which includes both curious ASs and curious/malicious participating nodes.

4.3.3 Design Goals

Given the aforementioned adversary model, VPA is designed with the following objectives.

- *Aggregation accuracy*: VPA should output accurate aggregation results in the absence of malicious attacks.
- *Aggregation/data integrity*: any attempt of injecting false data should be detected with certainty.
- *Data/user privacy*: Each user's datum should be hidden from all the other parties with high probability.
- *Efficiency*: VPA should incur low communication and computation overhead.

4.4 VPA$^+$: Verifiable Privacy-Preserving Additive Aggregation

In this section, we present VPA$^+$, a novel scheme to enable verifiable privacy-preserving additive aggregation. Without loss of generality, our discussion focuses on a cell with AS $\mathcal{A}$ and a set of n nodes, denoted by $\mathcal{U}$. We will also use Sum aggregation as an example, based on which other additive aggregation functions such as Average and Variance [11] can be easily realized.

4.4.1 Overview and Basic Idea

We observe that either of user privacy and aggregation integrity alone can be easily achieved if we ignore the other. On the one hand, if aggregation integrity is the only concern, a straightforward solution is to let each node submit its datum directly to $\mathcal{A}$ along with a message authentication code (MAC). The AS can then verify the authenticity of each datum and compute the correct sum. This naive approach, however, offers no data privacy to users. On the other hand, many existing techniques such as [29, 44] can realize privacy-preserving data aggregation, but a malicious node can launch the false-data injection attack without being detected.

Inspired by the above observation, we divide the whole aggregation process into two phases. In the first phase, each node submits to $\mathcal{A}$ a commitment, which is a homomorphic MAC of its datum and has a nice *one-way* property that $\mathcal{A}$ cannot deduce the corresponding datum. The homomorphic property of individual commitments enables $\mathcal{A}$ to compute an aggregate commitment corresponding to the Sum aggregate of all nodes' data. In the second phase, nodes perform privacy-preserving in-network data aggregation for $\mathcal{A}$ to derive the Sum aggregate without disclosing any individual datum with overwhelming probability. Finally, $\mathcal{A}$ can verify the integrity of the Sum aggregate by using the aggregate commitment derived in the first phase. In what follows, we detail the design of VPA$^+$, which includes *aggregation initialization*, *commitment submission*, *privacy-preserving in-network aggregation*, and *aggregation verification*.

4.4.2 Aggregation Initialization

The AS $\mathcal{A}$ initializes the aggregation process by selecting a large prime p and a generator g of the group $\mathbb{Z}_p^* = \{1, \ldots, p-1\}$. The parameters p and g should ensure the computational hardness of the *discrete logarithm problem*, that is, given a random $y \in \mathbb{Z}_p^*$, it is computationally infeasible to find the unique integer $x \in [0, p-2]$ such that $g^x = y \mod p$. Assume that each node has reported to $\mathcal{A}$ what kinds of data it could generate when moving into $\mathcal{A}$'s cell. Let $\mathcal{U}$ denote the set of $n = |\mathcal{U}|$ users that $\mathcal{A}$ has selected and motivated to participate in data aggregation.[2] Finally, $\mathcal{A}$ broadcasts an aggregation request $\langle p, g, \mathcal{U}, r \rangle$, where r is a random nonce for message freshness. It is worth noting that the aggregation request can be sent as part of $\mathcal{A}$'s periodic service beacons and need be authenticated properly as in [137] to prevent attackers from sending fake aggregation requests, which we have ignored here for the focus of this chapter. In addition, there can be various methods to transmit a condensed version of $\mathcal{U}$, which is also not discussed here for simplicity.

[2] How $\mathcal{A}$ selects $\mathcal{U}$ from candidate users and appropriately stimulate their participation is an orthogonal topic deserving independent investigation.

4.4.3 Commitment Submission

In this phase, each node $i \in \mathcal{U}$ submits to $\mathcal{A}$ a commitment, which is a homomorphic MAC of its datum d_i after appropriate expansion. In contrast to traditional MAC, a homomorphic MAC function $H(\cdot)$ has the additional property that the given the homomorphic MACs of two messages, say $H(m_1)$ and $H(m_2)$, anyone can derive $H(m_1 + m_2)$ without knowing m_1 or m_2. VPA$^+$ uses a simple homomorphic MAC construction as follows,

$$H(m) = g^m \mod p,$$

where $m \in [0, p-2]$. It is easy to see that $H(\cdot)$ is homomorphic because $\forall m_1, m_2 \in [0, p-2]$,

$$H(m_1 + m_2) = g^{m_1+m_2} = H(m_1)H(m_2) \mod p.$$

Before generating the commitment, each node i first need expand its datum d_i to introduce sufficient randomness. Note that the data range in many PC-USS applications is usually limited. For instance, in a traffic monitoring application, the driving speed is between 0 and 100 miles. If node i directly submits $H(d_i)$ to $\mathcal{A}$, then $\mathcal{A}$ can deduce d_i by exhaustive search. To avoid this situation, each node i expands d_i by adding a random number. In particular, assume that each datum d_i is of l bits. Node i generates a random number r_i of ϕ bits known only to itself and computes

$$e_i = 2^{l+\lfloor \log_2 n \rfloor} \cdot r_i + d_i \ . \tag{4.1}$$

where ϕ is a system parameter determining the difficulty of exhaustive search. Alternatively, we can view e_i as the concatenation of r_i, $\lfloor \log_2 n \rfloor$ zeros, and d_i as follows

$$e_i = r_i, \overbrace{0, \ldots, 0}^{\lfloor \log_2 n \rfloor}, d_i \ .$$

The reason to separate r_i and d_i by $\lfloor \log_2 n \rfloor$ zeros can be explained as follows. If we perform Sum aggregation over all e_i, then we have

$$\begin{aligned}\sum_{i\in\mathcal{U}} e_i &= 2^{l+\lfloor \log_2 n \rfloor} \cdot \sum_{i\in\mathcal{U}} r_i + \sum_{i\in\mathcal{U}} d_i \\ &\leq 2^{l+\lfloor \log_2 n \rfloor} \cdot \sum_{i\in\mathcal{U}} r_i + n(2^l - 1) \\ &< 2^{l+\lfloor \log_2 n \rfloor} \cdot \sum_{i\in\mathcal{U}} r_i + 2^{l+\lfloor \log_2 n \rfloor} \ .\end{aligned}$$

It follows that

$$\sum_{i\in\mathcal{U}} d_i = \sum_{i\in\mathcal{U}} e_i \mod 2^{l+\lfloor \log_2 n \rfloor} \ . \tag{4.2}$$

This property will be used later by $\mathcal{A}$ to derive the correct aggregation result without knowing $\{r_i\}_{i\in\mathcal{U}}$.

To prevent malicious nodes from submitting arbitrary data, we require that d_i and e_i be generated and authenticated by node i's TPM. Recall that $\mathcal{A}$ has assigned a secret key k_i to node i and another secret key κ_i to node i's TPM after mutual authentication (see Section 4.3.1). Node i submits to $\mathcal{A}$ the following message.

$$i \rightarrow \mathcal{A} : i, \langle H(e_i), h(\kappa_i || H(e_i)) \rangle_{k_i} \ ,$$

where $h(\cdot)$ denotes a good hash function, and $\langle \cdot \rangle_*$ denotes a symmetric-key encryption operation using the key on the subscript.

On receiving the message, the AS locates k_i and κ_i using node ID i. It uses k_i to decrypt the message, and then verifies $h(\kappa_i || H(e_i))$ using κ_i. If the verification succeeds, $\mathcal{A}$ considers $H(e_i)$ an authentic commitment and drops it otherwise. If $\{H(e_i)\}_{i=1}^n$ are all authentic, the AS proceeds to derive the aggregate commitment corresponding to $\sum_{i=1}^n e_i$ by computing

$$\begin{aligned}H(\sum_{i\in\mathcal{U}} e_i) &= \prod_{i\in\mathcal{U}} H(e_i) \mod p \\ &= \prod_{i\in\mathcal{U}} g^{e_i} \mod p \\ &= g^{\sum_{i\in\mathcal{U}} e_i} \mod p \ .\end{aligned}$$

4.4.4 Privacy-Preserving In-Network Data Aggregation

In this phase, nodes jointly perform in-network aggregation over their expanded data without disclosing them. This phase requires the establishment of an on-demand temporary aggregation tree. In particular, the AS $\mathcal{A}$ broadcast an *aggregation-tree formation* request, which specifies any node, say $v \in \mathcal{U}$, as the root of the aggregation tree. On receiving the request, node v rebroadcasts it via its Bluetooth or WiFi interface, depending on the particular method (e.g., [120]) it uses to communicate with neighboring nodes. Upon receiving the request for the first time, each node further rebroadcasts it and records the parent node from which this request came from. In this way, an aggregation tree is formed and rooted at node v which can directly communicate with $\mathcal{A}$.

In what follows, we present two techniques for privacy-preserving in-network Sum aggregation over all expanded data with different user-privacy guarantees and communication overhead. To facilitate presentation, we define node i's *aggregation neighbors* as i's neighboring nodes on the aggregation tree, denoted by $\mathcal{T}_i$.

4.4.4.1 Method 1: Data Perturbation (DP)

In this method, each node i perturbs its expanded datum e_i before actual aggregation. Since e_i is of $l + \lfloor \log_2 n \rfloor + \phi$ bits, we have

$$\begin{aligned}\sum_{i \in \mathcal{U}} e_i &\leq n \cdot (2^{l+\lfloor \log_2 n \rfloor + \phi} - 1) \\ &< 2^{l+2\lfloor \log_2 n \rfloor + \phi} ,\end{aligned}$$

i.e., that $\sum_{i \in \mathcal{U}} e_i$ is at most of $l + 2\lfloor \log_2 n \rfloor + \phi$ bits.

Denote by $h_1(\cdot)$ a good hash function of $l + 2\lfloor \log_2 n \rfloor + \phi$ bits. Each node i generates a perturbed datum α_i by computing

$$\alpha_i = h_1(k_i || r) + e_i \mod 2^{l+2\lfloor \log_2 n \rfloor + \phi} , \tag{4.3}$$

where k_i is the secret key shared between node i and the AS and r is the nonce broadcasted by $\mathcal{A}$.

Each node then performs in-network aggregation over its perturbed datum by adding it to the values received from its children on the aggregation tree and then transmitting the result to its parent. Finally, the AS $\mathcal{A}$ can obtain $\sum_{i\in\mathcal{U}} \alpha_i$ by summing the values received from the root of the aggregation tree, i.e., node v. Since $\mathcal{A}$ knows k_i for each $i \in \mathcal{U}$, it can compute all $h_1(k_i||r)$ and derive $\sum_{i\in\mathcal{U}} e_i$ by computing

$$\sum_{i\in\mathcal{U}} e_i = \sum_{i\in\mathcal{U}} \alpha_i - \sum_{i\in\mathcal{U}} h_1(k_i||r) \mod 2^{l+2\lfloor \log_2 n \rfloor+\phi} . \tag{4.4}$$

Since e_i is completely concealed by $h_1(k_i||r)$, which is only known by $\mathcal{A}$, other curious nodes, e.g., node i's neighbors on the aggregation tree, cannot derive e_i by monitoring i's incoming and outgoing transmission. Unfortunately, node i's data privacy can still be breached if $\mathcal{A}$ colludes with node i's aggregation neighbors.

4.4.4.2 Method 2: Peer-to-Peer Slicing and Mixing

To defend against $\mathcal{A}$ colluding with other curious nodes, we further propose another approach based on peer-to-peer data slicing and mixing. In this approach, before participating in in-network aggregation, each node i randomly divides its expanded datum e_i into multiple slices and mixes them with those from selected peers, such that data privacy can be preserved without affecting the correctness of the final aggregation result.

Specifically, before answering the query, each node i slices e_i into $t+1$ random slices $\{s_{i,j}\}_{j=1}^{t+1}$ with $t \leq n-1$, such that

$$e_i = \sum_{j=1}^{t+1} s_{i,j} \mod 2^{l+2\lfloor \log_2 n \rfloor+\phi} .$$

Then node i keeps $s_{i,t+1}$ to itself while sending each other slice to a unique peer called a *cover node*. Next, each node i adds the slices received from other nodes to its remained slice $s_{i,t+1}$ and conducts in-network aggregation as in Method 1. Finally, $\mathcal{A}$ adds up all the received values. It is easy to see that the result is exactly the Sum aggregate of interest.

This slicing technique shares the similar idea as PDA [44], while our application scenario is totally different. In particular, PDA is designed for sensor networks with relatively static network topology, where all the nodes know each other and have pairwise shared keys whereby to encrypt/decrypt data slices transmitted from any node to its chosen cover nodes. Such assumptions no longer holds in our target scenarios, where nodes in a cell are dynamically changing. Since the nodes do not know each other beforehand, they have no pre-shared keys for end-to-end encryption. This significant difference necessitates novel cover-selection strategies. In what follows, we detail two cover-selection approaches that specially tailored for our target scenario.

Random Cover Selection (RCS**)** As its name suggested, in this approach, each node randomly chooses t cover nodes from $\mathcal{U}$ and sends a data slice to each of them. The challenge is how a node can establish a shared key with each of its cover nodes for end-to-end encryption of its shares. VPA$^+$ uses the following method. Consider node i as an example with data e_i to share. It first slices e_i into $\{s_{i,j}\}_{j=1}^{t+1}$ and then randomly chooses a set of t nodes from $\mathcal{U}$ as its cover nodes, denoted by $\mathcal{C}_i \subseteq \mathcal{U}$. For any cover node $j \in \mathcal{C}_i$, node i computes a shared key $k_{i,j}$ based on its temporal public/private keys ID_i/K_i^{-1} and ID_j by using the method in our previous work [137] and then sends an encrypted unique slice $s_{i,\tau_{i,j}}$ to node j as follows.

$$i \rightarrow j : ID_i, \langle s_{i,\tau_{i,j}}, h(s_{i,\tau_{i,j}}) \rangle_{k_{i,j}}$$

Since the route to j might not be known, the packet transmission is normally preceded by an on-demand route discovery process using protocols like AODV [87]. On receiving the message, node j can derive the same key $k_{i,j}$ using its temporal public/private keys ID_j/K_j^{-1} and ID_i according to [137] and then decrypts the packet to get $s_{i,\tau_{i,j}}$. Node i repeats this process for all its cover nodes, and so does every other node in $\mathcal{U}$.

Each node waits for sufficient time to receive all the slices from other nodes choosing it as cover. Let $\mathcal{S}_i \subset \mathcal{U}$ denote the set of nodes selecting i as a cover node.

Each node i computes its share as

$$\beta_i = s_{i,t+1} + \sum_{j \in \mathcal{S}_i} s_{j,\tau_{j,i}} \mod 2^{l+2\lfloor \log_2 n \rfloor + \phi} . \tag{4.5}$$

Finally, all the nodes perform in-network aggregation over there shares as in Method 1 so that the AS $\mathcal{A}$ finally receives $\sum_{i \in \mathcal{U}} \beta_i$ which equals $\sum_{i \in \mathcal{U}} e_i$.

μ-Hop Cover Selection (μCS) Random cover selection may not be efficient because cover nodes are randomly chosen regardless of their locations. As a result, an on-demand route discovery process is often incurred to find a route to a chosen cover node multi-hop away. This may cause unnecessarily high energy consumption because a route request often involves cell-wide broadcasting.

We observe that it is unnecessary for each node i to predetermine the slices $\{s_{i,j}\}_{j=1}^{t+1}$ and send each of them to a cover node. Instead, node i can broadcast a random seed within its μ-hop neighborhood, in which every node is chosen as a cover node and can compute a slice using their shared key.

Specifically, in μ-hop cover selection, each node i initiates the slicing process by broadcasting a slicing request with a random seed r_i and a TTL value set to μ. Upon receiving a request with a TTL larger than one, each node further broadcasts it after decreasing the TTL by one. A node should only process the first copy of the same request which may be heard multiple times. In addition, each node memorizes the parent node from which this request came from. In this way, a routing tree of depth μ is formed and rooted at node i. When a node receives a request with the TTL value equal to one, the node should send a slicing response to its parent node which in turn forwards the response via the routing tree back to node i after appending its ID.

Each node waits sufficient time and then updates its share as follows. Consider node i as an example. Suppose that node i has received slicing responses from the set of nodes $\mathcal{C}_i$ and slicing requests from the set of nodes $\mathcal{S}_i$, i.e., the set of nodes choosing i as covers. Node i derives a shared key $k_{i,j}$ for each $j \in \mathcal{C}_i \bigcup \mathcal{S}_i$ according to [137]

and updates its share by computing

$$\beta_i = e_i - \sum_{j \in \mathcal{C}_i} h_1(r_i || k_{i,j}) + \sum_{j \in \mathcal{S}_i} h_1(r_j || k_{i,j}) \mod 2^{l+2\lfloor \log_2 n \rfloor + \phi} . \tag{4.6}$$

Finally, all the nodes perform in-network aggregation over their shares so that the AS $\mathcal{A}$ finally obtains$\sum_{i \in \mathcal{U}} \beta_i$ which equals $\sum_{i \in \mathcal{U}} e_i$.

Unlike in random cover selection, the number of cover nodes in μ-hop cover selection is a random variable which cannot be determined before the process is completed. Intuitively, the larger μ, the more cover nodes discovered, the higher privacy and the communication cost, and vice versa.

4.4.5 Aggregation-Result Verification

After in-network aggregation via Method 1 or 2, the AS obtain $\sum_{i \in \mathcal{U}} e_i$. It first verifies its integrity by checking if

$$g^{\sum_{i \in \mathcal{U}} e_i} = \prod_{i \in \mathcal{U}} H(e_i) \mod p.$$

If so, $\mathcal{A}$ considers $\sum_{i \in \mathcal{U}} e_i$ authentic and proceeds to derive $\sum_{i \in \mathcal{U}} d_i$ by computing

$$\sum_{i \in \mathcal{U}} d_i = \sum_{i \in \mathcal{U}} e_i \mod 2^l ,$$

which should hold according to Eq. (4.2).

4.4.6 Performance Analysis

Now we analyze the performance of VPA$^+$ with regard to its aggregation-integrity provision, data-privacy guarantee, and the associated overhead.

4.4.6.1 Aggregation Integrity

We first have the following theorem regarding the aggregation integrity of VPA$^+$.

Theorem 4.4.1. Assume that each node's datum is generated and authenticated by TPM and that $p > 2^{l+2\lfloor \log_2 n \rfloor + \phi}$. VPA$^+$ allows the AS to detect any false-data injection attack.

We give the proof in Appendix J.

4.4.6.2 Data Privacy

To evaluate the data-privacy provision of VPA$^+$, we define *exposure probability*, denoted by P_{exp}, as the probability that a node i's data d_i is disclosed during aggregation. To enable quantitative analysis, we assume that each node has N_{tree} aggregation neighbors on average. We also assume that there are M_c out of M curious ASs and n_c out of n curious nodes.

We then have the following theorems regarding the exposure probability under VPA$^+$.

Theorem 4.4.2. The exposure probability under DP is given by

$$P_{\text{exp}} = \frac{M_c}{M} \cdot \frac{\binom{n-n_c}{n_c-N_{\text{tree}}}}{\binom{n}{n_c}} \ . \tag{4.7}$$

We give the proof in Appendix K.

Theorem 4.4.3. The exposure probabilities under RCS and μCS are bounded by

$$P_{\text{exp}} \leq \frac{\binom{n-n_c}{n_c-w}}{\binom{n}{n_c}} \ , \tag{4.8}$$

where

$$w = \begin{cases} \max(N_{\text{tree}}, t) & \text{for RCS,} \\ \max(N_{\text{tree}}, \sum_{x=1}^{\mu} N_x) & \text{for } \mu\text{CS,} \end{cases} \tag{4.9}$$

is the minimum number of nodes colluding with $\mathcal{A}$.

We give the proof in Appendix L.

4.4.6.3 Overhead Analysis

Now we analyze the computation and communication overhead incurred by VPA$^+$ for each node.

For computation overhead, each node need perform one exponentiation to generate one commitment of its data. In addition, each node i need compute the shared key $k_{i,j}$ for each node $j \in \mathcal{C}_i \bigcup \mathcal{S}_i$ under RCS and μCS.

We assume that the average distance two random chosen nodes is L hops. Also denote by l_{tree}, l_{seed}, l_{hmac}, l_{h} the length of a aggregation tree formation request, a slicing request in μCS, a homomorphic MAC, and $h(\cdot)$, respectively. We then have the following theorem regarding the communication overhead incurred by VPA$^+$.

Theorem 4.4.4. The communication overhead incurred by VPA$^+$ in bits is given by

$$\mathsf{T}_{\text{VPA}^+} = nl_{\text{tree}} + \mathsf{T}_{\text{commit}} + \mathsf{T}_{\text{agg}} \ , \tag{4.10}$$

where

$$\mathsf{T}_{\text{commit}} = n(\lambda + l_{\text{hmac}} + l_{\text{h}}) \tag{4.11}$$

is the overhead incurred by transmitting commitments to $\mathcal{A}$, and

$$\mathsf{T}_{\text{agg}} = \begin{cases} nl_{\text{data}} & \text{for DP,} \\ nt(nl_{\text{req}} + L(l_{\text{rsp}} + l_{\text{data}} + \lambda)) + nl_{\text{data}} & \text{for RCS,} \\ n((1 + \sum_{x=1}^{\mu-1})N_x(\lambda + l_{\text{seed}}) + \sum_{x=1}^{\mu} N_x\lambda) + nl_{\text{data}} & \text{for } \mu\text{CS,} \end{cases} \tag{4.12}$$

is the overhead incurred by in-network aggregation, and $l_{\text{data}} = l + 2\lfloor \log_2 n \rfloor + \phi$.

We give the proof in Appendix M.

4.5 VPA$^{\oplus}$: Verifiable Privacy-Preserving Non-additive Aggregation

VPA$^+$ cannot be directly applied to non-additive aggregation functions such as Max/Min, Median, Percentile, and Histogram, which have wide applications in practice. In this section, we propose VPA$^{\oplus}$ as an extension of VPA$^+$ to support non-additive aggregation.

4.5.1 Basic Idea

Our key observation is that all the above non-additive aggregation functions are closely related to Count aggregation that ask for the number of nodes whose values are above, below, or equal to a certain value. In particular, let $\mathsf{Count}[Q]$ be the number of nodes with data satisfying the condition Q. Also denote by $d_{\max}, d_{\min}, d_{\text{med}}, d_{\sigma-\text{per}}$, the Max, Min, Median, and σ-percentile of a data set, respectively. It is easy to see that the following conditions hold.

- *Max*:

$$\begin{cases} \mathsf{Count}[d > d_{\max}] = 0, \\ \mathsf{Count}[d = d_{\max}] > 0 \ . \end{cases} \tag{4.13}$$

- *Min*:

$$\begin{cases} \mathsf{Count}[d < d_{\min}] = 0, \\ \mathsf{Count}[d = d_{\min}] > 0 \ . \end{cases} \tag{4.14}$$

- *Median*:

  – If n is odd, then

$$\begin{cases} \mathsf{Count}[d \leq d_{\mathrm{med}}] \geq \lfloor n/2 \rfloor, \\ \mathsf{Count}[d \geq d_{\mathrm{med}}] \geq \lfloor n/2 \rfloor \ . \end{cases} \tag{4.15}$$

  – If n is even, then there exists $i, j \in \mathcal{U}$, such that $d_i \leq d_j$ and

$$\begin{cases} \mathsf{Count}[d \leq d_i] \geq n/2, \\ \mathsf{Count}[d < d_i] < n/2, \\ \mathsf{Count}[d \geq d_j] \geq n/2, \\ \mathsf{Count}[d > d_j] < n/2 \ , \end{cases} \tag{4.16}$$

  and $d_{\mathrm{med}} = (d_i + d_j)/2$.

- σ*-percentile*: we only show the simplest case here

$$\begin{cases} \mathsf{Count}[d \leq d_{\sigma-\mathrm{per}}] \geq \lfloor \sigma n/100 \rfloor, \\ \mathsf{Count}[d \geq d_{\sigma-\mathrm{per}}] \geq \lfloor (100-\sigma) n/100 \rfloor \ . \end{cases} \tag{4.17}$$

Conversely, if we can find d^* such that the conditions in Eq. (4.13) (respectively, (4.14), (4.15), (4.16) (4.17)) hold, then we have $d^* = d_{\max}$ (respectively, $d_{\min}$, d_{med}, $d_{\sigma-\mathrm{per}}$). Since Count is an additive aggregation function, it can be realized by VPA$^+$. Built on the above observation, VPA$^\oplus$ combines VPA$^+$ with binary search to realize non-additive aggregation functions through a series of verifiable privacy-preserving Count aggregations.

4.5.2 Scheme Description

Given a non-additive aggregation request, $\mathcal{A}$ transforms it into a series of Count queries with conditions $Q_1, Q_2, \ldots,$ until the desired d^* is found, where Q_x is determined by the result of the previous Count query with condition Q_{x-1}. Each Count query Q_x asks how many nodes possess data above, below, or equal to a threshold, called a *count index*. Each node i with datum d_i satisfying condition Q_x gives an answer "yes", or "no" otherwise, by a single bit of value one or zero, respectively. The answers are then aggregated via VPA$^+$ to let $\mathcal{A}$ get $\mathtt{Count}(Q_x)$ with both user-privacy and aggregation-integrity guarantees.

Below we brief how to realize privacy-preserving Max/Min, Median, Histogram, and Percentile aggregation queries under the assumption that each data value d_i is an integer between $[0, 2^l - 1]$. It is easy to extend our technique to other non-additive aggregation functions.

4.5.2.1 Max/Min

Since the Min operation is opposite to the Max operation, we just illustrate the latter for brevity. Given a Max aggregation request, $\mathcal{A}$ first issues a Count query with $Q_1 = [d \geq 2^{l-1}]$ and then aggregates the received data via VPA$^+$ to get the number of "yes" answers, denoted by θ_1. If $\theta_1 \geq 1$, the maximum value should be in $[2^{l-1}, 2^l - 1]$, so $\mathcal{A}$ will send a new Count query with $Q_2 = [d \geq 2^{l-1} + 2^{l-2}]$; otherwise, the maximum value should be in $[0, 2^{l-1} - 1]$, so $\mathcal{A}$ will send a new Count query with $Q_2 = [d > 2^{l-2}]$. The *suspicion range* in which the maximum value is located is reduced by half for each additional Count query. This process continues until the suspicion range is reduced to one, in which case the last count index is exactly the maximum value, and the last query result equals the number of nodes with the maximum value.

4.5.2.2 Median/Percentile

Since Median is a special case of Percentile, we illustrate the former for simplicity, which can be easily extended to the latter. A median value is described as the number

separating the higher half of a sample, a population, or a probability distribution, from the lower half. Median aggregation can be realized in a similar fashion as Max. Here we present the case for n being odd for simplify, while the case of n being even can be realized accordingly.

Given a Median aggregation request, $\mathcal{A}$ first issues a Count query with $Q_1 = [d \geq 2^{l-1}]$ and obtains θ_1 via VPA$^+$. If $\theta_1 \geq (n+1)/2$, $\mathcal{A}$ sends the second Count query with $Q_2 = [d \geq 2^{l-1} + 2^{l-2}]$; otherwise, $\mathcal{A}$ sends the next query with $Q_2 = [d \geq 2^{l-2}]$. This process continues until the suspicion range of d_{med} is one, which takes total l queries. Suppose that the last two queries are $Q_{l-1} = [d \geq q_{l-1}]$ and $Q_l = [d \geq q_l]$ whereby $\mathcal{A}$ receives θ_{l-1} and θ_l, respectively. It follows that q_{l-1} and q_l differ by one. There are four cases.

- Case 1: if $q_{l-1} < q_l$ and $\theta_l \geq \lfloor n/2 \rfloor$, then we have $d_{\text{med}} = q_l$. The reasons are as follows. First, we must have $\theta_{l-1} < \lfloor n/2 \rfloor$, as otherwise $d_{\text{med}} \leq q_l - 1$, and q_l should not be queried. Second, there must exist a query $Q_x = [d \geq q_l + 1]$ with $x \in [1, l-2]$, due to the property of binary search. Third, it must hold that $\theta_x < \lfloor n/2 \rfloor$, as otherwise $d_{\text{med}} > q_l + 1$ and neither q_{l-1} nor q_l should be queried.
- Case 2: if $q_{l-1} > q_l$ and $\theta_l \geq \lfloor n/2 \rfloor$, then we have $d_{\text{med}} = q_l$.
- Case 3: if $q_{l-1} < q_l$ and $\theta_l < \lfloor n/2 \rfloor$, then $d_{\text{med}} = q_l + 1$.
- Case 4: if $q_{l-1} > q_l$ and $\theta_l < \lfloor n/2 \rfloor$, then $d_{\text{med}} = q_l + 1$.

The reasoning for Cases 2$\sim$4 are similar to that of Case 1 and is thus omitted.

4.5.2.3 Histogram

In statistics, a histogram is a graphical display of tabulated frequencies, shown as bars, and shows the proportion of cases falling into each of several categories. Using Count query to realize Histogram is straightforward. In particular, given a Histogram aggregation request, $\mathcal{A}$ partitions the data range $[0, 2^l - 1]$ into a certain number of consecutive, non-overlapping intervals according to the aggregation request. It then sends a Count

query for each interval, and the corresponding query result will equal the number of nodes with data in that interval.

4.5.3 Performance Analysis

Since VPA$^{\oplus}$ is built upon VPA$+$, it can also ensure perfect aggregation integrity. We thus focus on analyzing the user-privacy provision and overhead of VPA$^{\oplus}$.

4.5.3.1 Data Privacy

The exposure probability $\mathrm{P}_{\mathrm{exp}}$ used to analyze the performance of VPA^{+} can no longer precisely measure the privacy provision of non-additive aggregation. For example, even if the answer of node i to a Count query Q_x is disclosed, the adversary can only narrow down the search of d_i to a certain range instead of precisely determining d_i. Assume that the adversary knows that d_j is in a range of length ϵ after the whole query process. It is clear that the ratio $\rho = \epsilon/2^l$ can be used to analyze the privacy performance of the non-additive aggregation process: the larger ρ, the higher level of privacy provision, and vice versa.

In particular, when $\rho = 1$, the adversary has no clue about what d_i is; when $\rho = 2^{-l}$, i.e., $\epsilon = 1$, the adversary has precisely located d_i. We call ρ the *suspicion ratio* of d_i hereafter. Without loss of generality, we use Max as an example to evaluate the performance of the non-additive aggregation process. The studies about other non-additive aggregation functions can be conducted similarly. Before proceeding, we want to mention that the Max/Min aggregation functions naturally disclose some information: any user's data will be smaller or equal to $d_{\max}$ and larger or equal to $d_{\min}$. No scheme can prevent this kind of privacy breach which is due to the aggregate functions themselves. In the following, we will ignore such natural privacy breach and focus on the loss of privacy occurring in the query process.

We make the following assumptions for analytical tractability. We assume that the aggregation tree is static for the entire sequence of l Count queries. For clarity, we consider a special case where the maximum value $d_{max} = 2^l - 1$. The similar process

can be used to analyze the more general case that d_{max} may be any value in $[0, 2^l - 1]$. We then have the following theorems regarding the expected suspicion ratio of VPA$^{\oplus}$.

Theorem 4.5.1. Assume that $d_{max} = 2^l - 1$, the expected suspicion ratio of VPA$^{\oplus}$ under DP or μCS is given by

$$\mathsf{E}[\rho] = 1 - P_{\exp} + (2^{-2l} + \sum_{x=1}^{l} 2^{-2x}) P_{\exp} \ , \tag{4.18}$$

where $P_{\exp}$ is the exposure probability of VPA^{+} that given in Eq. (4.7) for DP and Eq. (4.8) for μCS.

We give the proof in Appendix N.

Theorem 4.5.2. Assume that $d_{max} = 2^l - 1$, the expected suspicion ratio under of VPA$^{\oplus}$ under RCS is given by

$$\mathsf{E}[\rho] = \sum_{x=0}^{l-1} 2^{-x-1} \mathsf{E}[\rho_x] + 2^{-l} \mathsf{E}[\rho_l] \ , \tag{4.19}$$

where

$$\mathsf{E}[\rho_x] = \sum_{k_1=0}^{x} Pr(y_e = k_1) \sum_{k_2=x+1}^{l+1} Pr(n_e = k_2) \rho[y_e, n_e] \ , \tag{4.20}$$

$$Pr(y_e = k) = \begin{cases} (1 - P_{\exp})^x & \text{if } k = 0, \\ P_{\exp}(1 - P_{\exp})^{k-1} & \text{if } 1 \leq k \leq x \ , \end{cases} \tag{4.21}$$

$$Pr(n_e = k) = \begin{cases} P_{\exp}(1 - P_{\exp})^{k-x-1} & \text{if } x + 1 \leq k \leq l, \\ (1 - P_{\exp})^{l-x} & \text{if } k = l + 1 \ , \end{cases} \tag{4.22}$$

$$\rho[y_e, n_e] = \begin{cases} 2^{-y_e} - 2^{-n_e} & \text{if } y_e + 1 \leq n_e \leq l, \\ 2^{-y_e} & \text{if } n_e = l + 1 \ , \end{cases} \tag{4.23}$$

$P_{\exp}$ is the exposure probability of VPA^{+} that given in Eq. (4.8) for RCS.

We give the proof in Appendix O.

Table 4.1: Default simulation settings

Para.	**Val.**	**Para.**	**Val.**	**Para.**	**Val.**	**Para.**	**Val.**
M	10	M_c	5	n	200	n_c	50
λ	8	ϕ	160	μ	1	t	5
N_1	20.9	N_2	39.3	N_3	48.1	N_4	48.2
l	10	L	3.39	l_{tree}	160	l_{seed}	160
l_{req}	160	l_{rsp}	160	l_{hmac}	1024	N_{tree}	1.86

4.5.3.2 Overhead Analysis

VPA$^{\oplus}$ differs from VPA$+$ mainly in the communication overhead. Since it takes l queries to complete the aggregation process, each of which incurs communication overhead of $\mathsf{T}_{\text{VPA}^+}$, we thus have

$$\mathsf{T}_{\text{VPA}^{\oplus}} = l \cdot \mathsf{T}_{\text{VPA}^+} \ ,$$

where $\mathsf{T}_{\text{VPA}^+}$ is given in Eq. (4.10).

4.6 Performance Evaluation

In this section, we evaluate VPA$+$ and VPA$^{\oplus}$ using extensive simulations.

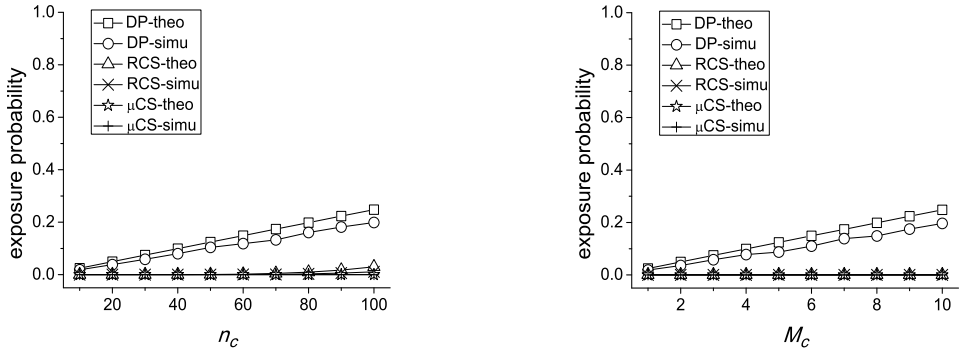
4.6.1 Simulation Setting

We simulate 10 cells of 1 km^2, each with an AS located at the center and 200 nodes randomly distributed within the cell. The transmission range of each node is 200m. This gives the average hop distance between two random nodes $L = 3.39$.

For our purpose, the simulation code is written in C++ and each data point represents an average of 50 simulation runs with different random seeds. Table 4.1 summarizes the default setting used in our simulation if not mentioned otherwise.

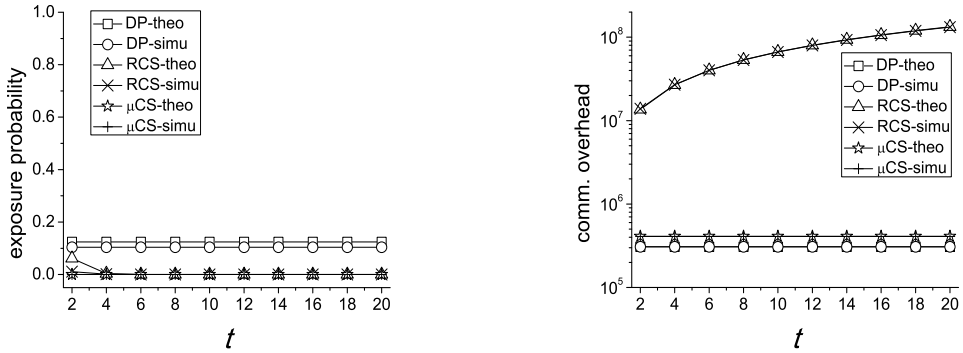
4.6.2 Evaluation of VPA$^+$

Fig. 4.2a shows both the theoretical and simulation results of the exposure probabilities of DP, RCS and μCS varying with n_c, the number of curious nodes. We can see that the exposure probabilities of all three schemes decrease as n_c increases. Among three schemes, DP has the highest exposure probability, followed by RCS and μCS. The rea-



(a) exposure probability (b) exposure probability

Figure 4.2: Impact of n_c and M_c.

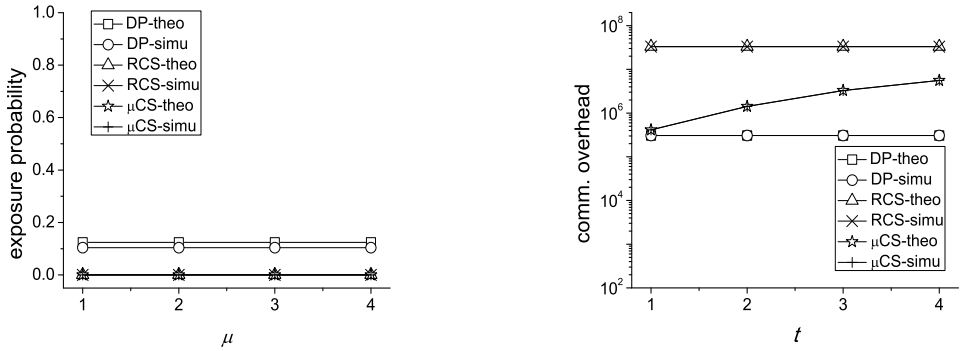


(a) exposure probability (b) communication overhead

Figure 4.3: Impact of t, the number of cover nodes on RCS.

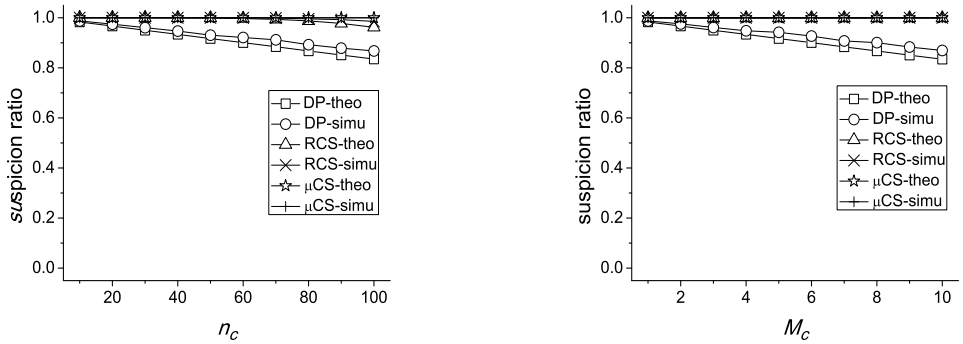
son is that on average, each node has only less than two neighbors on the aggregation tree (i.e., a spanning tree), making it easier for the adversary to compromise (or collude with) all the neighbors of a target node under DP. In contrast, it is much more difficult to compromise all the cover nodes under RCS and μCS. In addition, we can see that the P_{exp} of DP obtained via theoretical analysis is slightly higher than that obtained by simulations. The reason is that we round N_{tree} to $\lfloor N_{\text{tree}} \rfloor$ when computing $\binom{n-n_c}{n_c-N_{\text{tree}}}$ in Eq. (4.7), leading to higher P_{exp}.

Fig. 4.2b shows the impact of M_c, the number of curious ASs on the exposure probability of DP. Since curious ASs has no impact on RCS and μCS, there exposure



(a) exposure probability (b) exposure probability

Figure 4.4: Impact of μ.



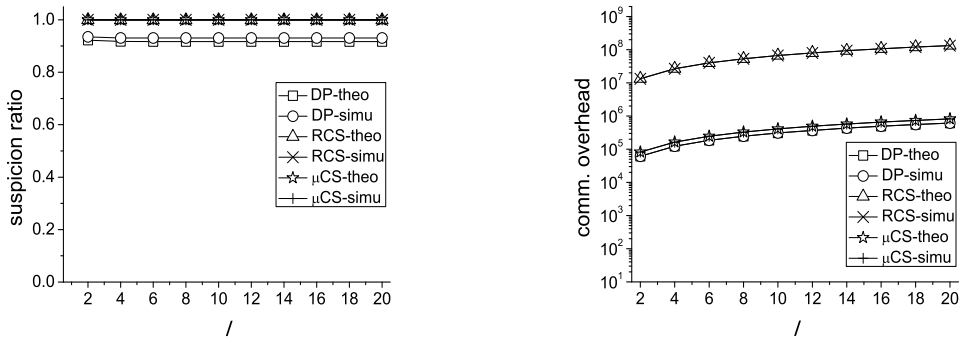
(a) suspicion ratio (b) suspicion ratio

Figure 4.5: Impact of n_c and M_c on suspicion ratio.

probabilities are shown only for references. We can see that the exposure probability of DP increases linearly with the number of curious AS increases, which is expected.

Fig. 4.3a shows the impact of t, the number of cover nodes, on the exposure probability of RCS, where the P_{exp}s of DP and μCS are shown only for reference. We can see that the P_{exp} of RCS decreases as t increases, and quickly drops to zero when $t > 4$. The reason is that the probability of all the t cover nodes being compromised decreases exponentially as t increases.

Fig. 4.3b shows the communication overhead of RCS varying with t. We can see that under the default settings, RCS incurs significantly higher communication overhead



(a) suspicion ratio (b) communication overhead

Figure 4.6: Impact of l.

than that of DP and RCS. This is anticipated since finding a cover node under RCS requires an AODV-like route discovery that involves a network-wide flooding.

Fig. 4.4 shows the impact of μ on the exposure probability and communication overhead of μCS, where the results of DP and RCS are only shown for reference. We can see from Fig. 4.4a that the exposure probability of μCS is not much affected by μ because P_{exp} is already close to zero when $\mu = 1$. In addition, We can see that the communication overhead of μCS increases moderately as μ increases, which is of no surprise.

4.6.3 Evaluation of VPA$^{\oplus}$

Fig. 4.5a shows the suspicion ratios of DP, RCS and μCS, varying with n_c. We can see that the suspicion ratios of all three schemes decrease as n_c increases. The reason is that the higher n_c, the lower P_{exp}, and the lower suspicion ratio, and vice versa. In addition, under the default setting, μCS has the highest suspicion ratio, followed by that of RCS and DP.

Fig. 4.5b shows the impact of M_c on the suspicion ratio of DP, where the performance of RCS and μCS are only shown for reference. We cans see that the larger M_c, the lower suspicion ratio, and vice versa, which is easy to understand.

Fig. 4.6 shows the impact of l on the suspicion ratio and communication over-

head of VPA$^{\oplus}$. We can see from Fig. 4.6a that the change in data range has negligible impact on the suspicion ratio of VPA$^{\oplus}$. The reason is that the suspicion ratio is determined by the last disclosed yes answer and the first disclosed no answer. Under the default setting, DP has the lowest suspicion ratio due to its highest P_{exp} among the three schemes (cf. Fig. 4.2a), while the suspicion ratios of both RCS and μCS are close to one. In addition, we can see from Fig. 4.6b that the communication overhead of VPA$^{\oplus}$ increases linearly as l increases, as it takes l Count queries to locate the desired aggregate.

4.6.4 Discussion

We summarize the evaluation results as follows.

- All three variants of VPA^{+} (i.e., DP, RCS, and μCS) can ensure aggregation integrity by detecting any false-data injection attempt.
- DP can provide user/data privacy with high probability while incurring the minimum communication overhead.
- RCS can provide user/data privacy against curious ASs with overwhelming probability while incurring the highest communication overhead.
- μCS can provide user/data privacy against curious ASs with overwhelming probability while incurring relatively low communication overhead.
- Built upon VPA^{+} and binary search, VPA$^{\oplus}$ can ensure both aggregation integrity and user/data privacy with communication overhead linear to the bit length of data.

In practice, μCS and the resulting VPA$^{\oplus}$ may be the best choices whose performance can be adjusted as needed.

4.7 Summary

In this chapter, we have presented the design and evaluation of VPA, a novel peer-to-peer approach to verifiable privacy-preserving aggregation for people-centric urban

sensing systems. VPA can support a wide range of additive and non-additive aggregation functions with strong user-privacy and aggregation-integrity guarantees. The high efficacy and efficiency of VPA are confirmed by thorough theoretical analysis and simulation results.

Chapter 5

PRIVATE MATCHING FOR PROXIMITY-BASED MOBILE SOCIAL NETWORKING

5.1 Introduction

Proximity-based mobile social networking (PMSN) becomes increasingly popular due to the explosive growth of smartphones. In particular, eMarketer estimated the US and worldwide smartphone users to be 73.3 million and 571.1 million in 2011, [1] respectively, and almost all smartphones have WiFi and Bluetooth interfaces. PMSN refers to the social interaction among physically proximate mobile users directly through the Bluetooth/WiFi interfaces on their smartphones or other mobile devices. As a valuable complement to web-based online social networking, PMSN enables more tangible face-to-face social interactions in public places such as bars, airports, trains, and stadiums [120]. In addition, PMSN may be the only feasible social networking tool when mobile users cannot access the Internet for online social networking, e.g., due to lack of Internet access minutes or very weak signals from cellular base stations or WiFi access points.

PMSN is conducted via applications running on smartphones or other mobile devices. Such applications can be offered by small independent developers. For instance, there are currently over 50 Bluetooth/WiFi chatting applications in the Android Market for Android devices and 60 in the App Store for Apple devices. Developing advanced Bluetooth/WiFi social networking applications also has recently attracted attention from the academia [120]. Moreover, online social network providers such as Facebook and Twitter may add PMSN functionalities to their future applications for smartphones and other mobile devices.

Private (profile) matching is indispensable for fostering the wide use of PMSN. On the one hand, people normally prefer to socialize with others having similar interests or background over complete strangers. Such social reality makes *profile matching* [65] the first step towards effective PMSN, which refers to two users comparing their per-

[1] http://www.emarketer.com/Report.aspx?code=emarketer_2000763

sonal profiles before real interaction. On the other hand, people have growing privacy concerns for disclosing personal profiles to arbitrary persons in physical proximity before deciding to interact with them [6, 21, 65, 69]. Although similar privacy concerns also exist in online social networking, preserving users' profile privacy is more urgent in PMSN, as attackers can directly associate obtained personal profiles with real persons nearby and then launch more targeted attacks. This situation leads to a circular dependency between personal-profile exchange and engagement in PMSN and thus necessitates *private matching*, in which two users to compare their personal profiles without disclosing them to each other.

Some elegant schemes such as [6, 65, 69] have recently been proposed to enable coarse-grained private matching for PMSN. Common to these schemes is the implicit assumption that each user's personal profile consists of multiple attributes chosen from a public set of attributes, which can be various interests [65], friends [6], or disease symptoms [69] in different contexts. Private matching is then converted into Private Set Intersection (PSI) [58, 122] or Private Set Intersection Cardinality (PSI-CA) [19, 30], whereby two mutually mistrusting parties, each holding a private data set, jointly compute the intersection [58,122] or the intersection cardinality [19,30] of the two sets without leaking any additional information to either party. These schemes [6,65,69] can enable only coarse-grained private matching and are unable to further differentiate users with the same attribute(s). For example, Alice, Bob, and Charlie all like watching movies and thus have "movie" as an attribute of their respective profile. Alice and Bob, however, both go to the cinema twice a week, while Charlie does so once every two weeks. If Alice can interact with only one of Bob and Charlie, e.g., due to time constraints, Bob is obviously a better choice. Under the existing schemes [6, 65, 69], however, Bob and Charlie appear the same to Alice. To solve this problem and thus further enhance the usability of PMSN calls for *fine-grained private matching*.

A natural first step towards fine-grained private matching for PMSN is to use fine-grained personal profiles. The basic idea is to associate a user-specific numerical

value with every attribute. For example, assume that every attribute corresponds to a different interest such as movie, sports, and cooking. The first time every user uses the PMSN application, he is prompted to create his profile by assigning a value to every attribute in the public attribute set defined by the PMSN application. Every attribute value is an integer in [0, 10] and indicates the level of interest from no interest (0) to extremely high interest (10). [2] Every personal profile is then defined as a set of attribute values, each corresponding to a unique attribute in the public attribute set.

Fine-grained personal profiles have significant advantages over traditional coarse-grained ones comprising only interested attributes from a public attribute set. First, fine-grained personal profiles enable finer differentiation among the users having different levels of interest in the same attribute. Continue with the previous example. Alice now can choose Bob over Charlie, as she and Bob have closer attribute values for "movie." In addition, fine-grained personal profiles enable personalized profile matching in the sense that two users can select the same agreed-upon metric from a set of candidate metrics to measure the similarity between their personal profiles or even different metrics according to their individual needs. The accompanying challenge is, however, how to ensure the privacy of fine-grained profile matching, which cannot be solved by existing solutions [6, 65, 69].

This chapter explores fine-grained private (profile) matching to foster the wide use of PMSN. Our main contributions can be summarized as follows.

- We motivate the requirement for and formulate the problem of fine-grained private (profile) matching for PMSN for the first time in the literature.
- We propose the notion of fine-grained personal profiles and a corresponding suite of novel private-matching protocols for different metrics measuring profile similarity. Our first three protocols are for the ℓ_1 distance, which is the sum of absolute difference in each attribute. We also propose a threshold-based protocol based

[2] Note that a user only needs to manually set the attribute values for interested attributes and leave all the other attribute values as their default values (0).

on the ℓ_1 distance, in which two users can determine whether the ℓ_1 distance between their profiles is smaller than some personally chosen threshold. We finally extend the third protocol to be a threshold-based protocol based on the MAX distance, which is the maximum absolute difference among all attributes.

- We provide thorough security analysis and performance evaluation of our proposed protocols and demonstrate their efficacy and efficiency under practical settings.

The rest of the chapter is organized as follows. Section 5.2 formulates the problem of fine-grained private matching in PMSN. Section 5.3 presents a suite of novel fine-grained private-matching protocols. Section 5.5 analyzes and evaluates the performance of the proposed protocols. Section 5.6 discusses the related work. Section 5.7 concludes this chapter.

5.2 Problem Formulation and Cryptographic Tool

In this section, we first state our assumption on PMSN and then formulates the problem of fine-grained private matching. Finally, we brief introduce Paillier's cryptosystem [82], the cryptographic tool underlying our protocol.

5.2.1 Proximity-Based Mobile Social Networking (PMSN)

We assume that each user carries a smartphone or some other mobile device with the same PMSN application installed. The PMSN application can be developed by small independent developers or offered by online social network service providers like Facebook as a function module of their applications built for mobile devices. More and more advanced PMSN applications have also been developed by the academia [120]. For convenience only, we shall not differentiate a user from his mobile device later.

A PMSN session involves two users and consists of three phases. First, two users need discover each other in the neighbor-discovery phase. Second, they need compare their personal profiles in the matching phase. Last, two matching users enter

the interaction phase for real information exchange. Our work is concerned with the first and second phases.

The PMSN application uses fine-grained personal profiles for fine-grained matching. In particular, the application developer defines a public attribute set consisting of d attributes $\{A_1, \ldots, A_d\}$, where d may range from several tens to several hundreds depending on specific PMSN applications. The attributes may have different meanings in different contexts, such as interests [65], disease symptoms [69], or friends [6]. For easier illustration, we hereafter assume that that each attribute corresponds to a personal interest such as movie, sports, and cooking. To create a fine-grained personal profile, every user selects an integer $u_i \in [0, \gamma - 1]$ to indicate his level of interest in A_i (for all $i \in [1, d]$) the first time he uses the PMSN application. As a fixed system parameter, γ could be a small integer, say 5 or 10, which may be sufficient to differentiate user's interest level. The higher u_i, the more interest the user has in A_i, and vice versa. More specifically, 0 and $\gamma - 1$ mean no interest and extremely high interest, respectively. Every personal profile is then defined as a vector $\langle u_1, \ldots, u_d \rangle$. The user can also modify his profile later on as needed.

There are two additional points worth noting. First, our scheme incurs negligible additional burden on the PMSN users in contrast to coarse-grained private matching schemes [6, 65, 69]. More specifically, a user only need rate a few selected attributes while leaving all the others as the default value 0. Second, the assumed attribute range $[0, \gamma - 1]$ is only for ease of illustration, and our protocols can directly support arbitrary attribute ranges. For example, the attribute range can be $[-\gamma + 1, \gamma - 1]$, where $-\gamma + 1$ corresponds to extreme dislike, and the default value 0 means neutral.

5.2.2 Problem Statement: Fine-Grained Private Matching in PMSN

We consider Alice with profile $\mathbf{u} = \langle u_1, \ldots, u_d \rangle$ and Bob with profile $\mathbf{v} = \langle v_1, \ldots, v_d \rangle$ as two exemplary users of the same PMSN application from here on. Assume that Alice wants to find someone to chat with, e.g., when waiting for the flight to depart. As the first step (Neighbor Discovery), she broadcasts a chatting request via the PMSN

application on her smartphone to discover proximate users of the same PMSN application. Suppose that she receives multiple responses including one from Bob who may also simultaneously respond to other persons. It should be noted that normally PMSN pseudonyms instead of real names are used in neighbor discovery. Due to time constraints or other reasons, both Alice and Bob can only interact with one stranger whose profile best matches hers or his. The next step (Profile Matching) is thus for Alice (or Bob) to compare her (or his) profile with those of others who responded to her (or whom he responded to). Our subsequent discussion will focus on the profile-matching process between Alice and Bob for the sake of simplicity. As in [21], we assume that the PMSN application is completely distributed and does not involve any third party in neighbor discovery, profile matching, and subsequent real user interactions.

Alice and Bob are both assumed to have privacy concerns about disclosing their personal profiles to complete strangers, so a privacy-preserving matching protocol is needed. In particular, let $\mathcal{F}$ denote a set of candidate matchings defined by the PMSN application developer, where each $f \in \mathcal{F}$ is a function over two personal profiles that measures their similarity. Our private-matching protocols allow Alice and Bob to either negotiate one common metric from $\mathcal{F}$ or choose different metrics according to their individual needs. We shall focus on the latter more general case henceforth, in which private matching can be viewed as two independent protocol executions, with each user initiating the protocol once according to her/his chosen matching. Assume that Alice chooses a matching metric $f \in \mathcal{F}$ and runs the privacy-matching protocol with Bob to compute $f(\mathbf{u}, \mathbf{v})$. According to the amount of information disclosed during the protocol execution, we define the following three privacy levels from Alice's viewpoint, which can also be equivalently defined from Bob's viewpoint for his chosen matching metric.

Definition 5.2.1. ***Level-I privacy****: When the protocol ends, Alice only learns $f(\mathbf{u}, \mathbf{v})$, and Bob only learns f.*

Definition 5.2.2. ***Level-II privacy****: When the protocol ends, Alice only learns $f(\mathbf{u}, \mathbf{v})$, and Bob learns nothing.*

Definition 5.2.3. ***Level-III privacy****: When the protocol ends, Alice only learns if $f(\mathbf{u}, \mathbf{v}) < \tau_A$ holds for some threshold τ_A of her own choice without learning $f(\mathbf{u}, \mathbf{v})$, and Bob learns nothing.*

For all three privacy levels, neither Alice nor Bob learns the other's personal profile. With level-I privacy, although Bob cannot learn $f(\mathbf{u}, \mathbf{v})$, he learns the matching metric f chosen by Alice. In contrast to level-I privacy, level-II privacy additionally requires that Bob learn nothing other than $f \in \mathcal{F}$. Finally, level-III privacy discloses the least amount of information by also hiding $f(\mathbf{u}, \mathbf{v})$ from Alice. We will introduce a suite of private-matching protocols satisfying one of the three privacy levels. Besides privacy guarantees, other design objectives include small communication and computation overhead, which can translate into the total energy consumption and matching time and thus are crucial for resource-constrained mobile devices and the usability of PMSN.

There might be passive attackers eavesdropping on the messages between Alice and Bob. All our protocols can ensure that the eavesdroppers are completely blind to the profiles of Alice and Bob and the matching metric(s) chosen by them, which will not be disclosed in plain text during the protocol execution. For simplicity, we will neglect passive eavesdroppers in subsequent protocol illustrations and analysis.

It is beyond the scope of this chapter to consider some other possible attacks. For example, Bob may manipulate the protocol output by using an arbitrary profile and/or not faithfully following the protocol operations (e.g., by changing intermediate computation results). It is fundamentally difficult to defend against this attack without involving a trusted third party as in [58, 65]. In particular, Alice cannot tell whether the protocol output is caused by Bob's misbehavior or they indeed having similar profiles. Our protocols, however, can guarantee one of the three privacy levels for Alice against Bob. Our protocols are also vulnerable to Denial-of-Service attacks in which an attacker

keeps sending or replying to chatting requests without finishing private matching with good users in order to consume their device resources. In addition, attackers may track PMSN users if they use static pseudonyms in neighbor discovery, private matching, and other PMSN communications. Moreover, intelligent attackers may launch the man-in-the-middle (MiM) attack by surreptitiously relay messages between Alice and Bob who are not physically proximate. These attacks are not unique to our private-matching scenario, and similar ones can apply to any wireless protocol involving message exchanges between multiple parties. The DoS attack can be mitigated by incorporating message puzzles [52] into protocol design, the tracking attack can be alleviated by letting users employ dynamic pseudonyms in PMSN communications, and the MiM attack can be tackled by using the device pairing protocol in [40] . Tight space limitations do not allow us to elaborate on these issues in detail here.

5.2.3 Cryptographic Tool: Paillier Cryptosystem

Our protocols rely on the Paillier cryptosystem [82], and we assume that every PMSN user has a unique Paillier public/private key pair which can be generated via a function module of the PMSN application. How the keys are generated and used for encryption and decryption are briefed as follows to help illustrate and understand our protocols.

- **Key generation.** An entity chooses two primes p and q and compute $N = pq$ and $\lambda = \mathrm{lcm}(p-1, q-1)$. It then selects a random $g \in \mathbb{Z}^*_{N^2}$ such that $\gcd(\mathsf{L}(g^\lambda \mod N^2), N) = 1$, where $\mathsf{L}(x) = (x-1)/N$. The entity's Paillier public and private keys are $\langle N, g\rangle$ and λ, respectively.

- **Encryption.** Let $m \in \mathbb{Z}_N$ be a plaintext to be encrypted and $r \in \mathbb{Z}_N$ be a random number. The ciphertext is given by

$$\mathsf{E}(m \mod N, r \mod N) = g^m r^N \mod N^2 , \tag{5.1}$$

  where $\mathsf{E}(\cdot)$ denotes the Paillier encryption operation on two integers modulo N. To simplify our expressions, we shall hereafter omit the modular notation inside $\mathsf{E}(\cdot)$.

- **Decryption.** Given a ciphertext $c \in \mathbb{Z}_{N^2}$, the corresponding plaintext can be derived as

$$\mathsf{D}(c) = \frac{\mathsf{L}(c^\lambda \mod N^2)}{\mathsf{L}(g^\lambda \mod N^2)} \mod N \ , \tag{5.2}$$

where $\mathsf{D}(\cdot)$ denotes the Paillier decryption operation hereafter.

The Paillier's cryptosystem has two very useful properties.

- **Homomorphic.** For any $m_1, m_2, r_1, r_2 \in \mathbb{Z}_N$, we have

$$\mathsf{E}(m_1, r_1)\mathsf{E}(m_2, r_2) = \mathsf{E}(m_1 + m_2, r_1 r_2) \mod N^2,$$
$$\mathsf{E}^{m_2}(m_1, r_1) = \mathsf{E}(m_1 m_2, r_1^{m_2}) \mod N^2 \ .$$

- **Self-blinding.**

$$\mathsf{E}(m_1, r_1) r_2^N \mod N^2 = \mathsf{E}(m_1, r_1 r_2) \ ,$$

which implies that any ciphertext can be changed to another without affecting the plaintext.

The Paillier cryptosystem is semantically secure for sufficiently large N and g, which means that it is infeasible for a computationally bounded adversary to derive significant information about a message (plaintext) when given only its ciphertext and the corresponding public key. To facilitate our illustrations, we assume that N and g are of 1024 and 160 bits, respectively, for sufficient semantical security of the Paillier cryptosystem [21]. Under this assumption, a public key $\langle N, g \rangle$ is of 1184 bits, a ciphertext is of $2\log_2 N$=2048 bits, a Paillier encryption needs two 1024-bit exponentiations and one 2048-bit multiplication, and a Paillier decryption costs essentially one 2048-bit exponentiation.

5.3 Fine-Grained Private Matching Protocols

In this section, we present three private-matching protocols to support different matching metrics and offer different levels of privacy. In particular, Protocol 1 is for the ℓ_1-distance matching metric and can offer level-I privacy, Protocol 2 supports a family of

additively separable matching metrics and can offer level-II privacy, and Protocol 3 is an enhancement of Protocol 3 for supporting level-III privacy.

A complete matching process involves Alice with profile $\mathbf{u} = \langle u_1, \ldots, u_d \rangle$ and Bob with profile $\mathbf{v} = \langle v_1, \ldots, v_d \rangle$, each running an independent instance of the same or even different private-matching protocol. Let f denote any matching metric supported by Protocols 1 to 3. The larger $f(\mathbf{u}, \mathbf{v})$, the less similar $\mathbf{u}$ and $\mathbf{v}$, and vice versa. We can thus consider $f(\mathbf{u}, \mathbf{v})$ some kind of distance between $\mathbf{u}$ and $\mathbf{v}$. Assume that Alice has a threshold τ_A and will accept Bob if $f(\mathbf{u}, \mathbf{v}) < \tau_A$. Similarly, Bob has a threshold τ_B and will accept Alice if $f(\mathbf{u}, \mathbf{v}) < \tau_B$. If both accept each other, they can start real information exchange. Our subsequent protocol illustrations and analysis will be from Alice's viewpoint, which can be similarly done from Bob's viewpoint. We assume that Alice has a Paillier public key $\langle N, g \rangle$ and the corresponding private key λ, which are generated as in Section 5.2.3. A practical security protocol often involves some routines such as using timestamps to mitigate replay attacks and message authentication codes for integrity protection. To focus on explaining our key ideas, we will neglect such security routines in protocol illustrations.

5.3.1 Protocol 1 for Level-I Privacy

Protocol 1 is designed for the ℓ_1 distance as the matching metric. Recall that every personal profile is a vector of dimension d. As probably the most straightforward matching metric, the ℓ_1 distance (also called the Manhattan distance) is computed by summing the absolute value of the element-wise subtraction of two profiles and is a special case of the more general ℓ_α distance defined as

$$\ell_\alpha(\mathbf{u}, \mathbf{v}) = (\sum_{i=1}^{d} |v_i - u_i|^\alpha)^{\frac{1}{\alpha}} , \tag{5.3}$$

where $\alpha \geq 1$. When $\alpha = 1$, we have $\ell_1(\mathbf{u}, \mathbf{v}) = \sum_{i=1}^{d} |v_i - u_i|$. The ℓ_1 distance allows a user to evaluate whether the overall absolute difference between his and another user's profiles is above a threshold chosen by himself.

Protocol 1 is designed to offer level-I privacy from Alice's viewpoint with regard to Bob. It is a nontrivial adaptation from the protocol in [91] with significantly lower computation overhead to be shown shortly. Our basic idea is to first convert $\ell_1(\mathbf{u}, \mathbf{v})$ into the ℓ_2 distance between the unary representations of $\mathbf{u}$ and $\mathbf{v}$ and then compute the ℓ_2 distance using a secure dot-product protocol.

In particular, for all $x \in [0, \gamma-1]$, we define a binary vector $h(x) = \langle x_1, \ldots, x_{\gamma-1} \rangle$, where x_i is equal to one for $1 \leq i \leq x$ and zero for $x < i \leq \gamma - 1$. We also abuse the notation by defining another binary vector $\hat{\mathbf{u}} = h(\mathbf{u}) = \langle h(u_1), \ldots, h(u_d) \rangle = \langle \hat{u}_1, \ldots, \hat{u}_{(\gamma-1)d} \rangle$ and $\hat{\mathbf{v}} = h(\mathbf{v}) = \langle h(v_1), \ldots, h(v_d) \rangle = \langle \hat{v}_1, \ldots, \hat{v}_{(\gamma-1)d} \rangle$. It follows that

$$\begin{aligned} \ell_1(\mathbf{u}, \mathbf{v}) &= \sum_{i=1}^{d} |u_i - v_i| \\ &= \sum_{i=1}^{(\gamma-1)d} |\hat{u}_i - \hat{v}_i| \\ &= \sum_{i=1}^{(\gamma-1)d} |\hat{u}_i - \hat{v}_i|^2 = \ell_2^2(\hat{\mathbf{u}}, \hat{\mathbf{v}}) \ . \end{aligned} \tag{5.4}$$

The correctness of the above equation is straightforward. We can further note that

$$\begin{aligned} \ell_2^2(\hat{\mathbf{u}}, \hat{\mathbf{v}}) &= \sum_{i=1}^{(\gamma-1)d} |\hat{u}_i - \hat{v}_i|^2 \\ &= \sum_{i=1}^{(\gamma-1)d} \hat{u}_i^2 - 2 \sum_{i=1}^{(\gamma-1)d} \hat{u}_i \hat{v}_i + \sum_{i=1}^{(\gamma-1)d} \hat{v}_i^2 \\ &= \sum_{i=1}^{(\gamma-1)d} \hat{u}_i^2 - 2\hat{\mathbf{u}} \cdot \hat{\mathbf{v}} + \sum_{i=1}^{(\gamma-1)d} \hat{v}_i^2 \ . \end{aligned} \tag{5.5}$$

Since Alice and Bob know $\sum_{i=1}^{(\gamma-1)d} \hat{u}_i^2$ and $\sum_{i=1}^{(\gamma-1)d} \hat{v}_i^2$, respectively, we just need a secure dot-product protocol for Bob to compute $\hat{\mathbf{u}} \cdot \hat{\mathbf{v}}$ without knowing Alice's profile $\mathbf{u}$ or disclosing his profile $\mathbf{v}$ to Alice. Subsequently, Bob can return $-2\hat{\mathbf{u}} \cdot \hat{\mathbf{v}} + \sum_{i=1}^{(\gamma-1)d} \hat{v}_i^2$ for Alice to finish computing $\ell_2^2(\hat{\mathbf{u}}, \hat{\mathbf{v}})$ and thus $\ell_1(\mathbf{u}, \mathbf{v})$.

5.3.1.1 Protocol Details

The detailed operations of Protocol 1 are as follows.

1. Alice does the following in sequence.

a. Construct a vector $\hat{\mathbf{u}} = h(\mathbf{u}) = (h(u_1), \ldots, h(u_d)) = (\hat{u}_1, \ldots, \hat{u}_{(\gamma-1)d})$, where $\hat{u}_j$ is equal to one for every $j \in \mathcal{J}_{\mathbf{u}} = \{j | (i-1)(\gamma-1) < j \leq (i-1)(\gamma-1) + u_i, 1 \leq i \leq d\}$ and zero otherwise.

b. Choose a distinct $r_j \in \mathbb{Z}_N$ and compute $\mathsf{E}(\hat{u}_j, r_j)$ for every $j \in [1, (\gamma-1)d]$ using her public key.

c. Send $\{\mathsf{E}(\hat{u}_j, r_j)\}_{j=1}^{(\gamma-1)d}$ and her public key to Bob.

2. Bob does the following after receiving Alice's message.

a. Construct a vector $\hat{\mathbf{v}} = h(\mathbf{v}) = (h(v_1), \ldots, h(v_d)) = (\hat{v}_1, \ldots, \hat{v}_{(\gamma-1)d})$, where $\hat{v}_j$ is equal to one for every $j \in \mathcal{J}_{\mathbf{v}} = \{j | (i-1)(\gamma-1) < j \leq (i-1)(\gamma-1) + v_i, 1 \leq i \leq d\}$ and zero otherwise.

b. Compute

$$\begin{aligned} \mathsf{E}(\hat{\mathbf{u}} \cdot \hat{\mathbf{v}}, \prod_{j \in \mathcal{J}_{\mathbf{v}}} r_j) &= \mathsf{E}(\sum_{j \in \mathcal{J}_{\mathbf{v}}} \hat{u}_j, \prod_{j \in \mathcal{J}_{\mathbf{v}}} r_j) \\ &= \prod_{j \in \mathcal{J}_{\mathbf{v}}} \mathsf{E}(\hat{u}_j, r_j) \mod N^2 , \end{aligned} \tag{5.6}$$

where the first equality sign is very obvious, and the second is due to the homomorphic property of the Paillier cryptosystem.

c. Compute

$$\mathsf{E}((N-2)\hat{\mathbf{u}} \cdot \hat{\mathbf{v}}, s) = \mathsf{E}^{N-2}(\hat{\mathbf{u}} \cdot \hat{\mathbf{v}}, \prod_{j \in \mathcal{J}_{\mathbf{v}}} r_j) \mod N^2 ,$$

where $s = (\prod_{j \in \mathcal{J}_{\mathbf{v}}} r_j)^{N-2} \mod N$. This equation holds again due to the homomorphic property of the Paillier cryptosystem.

d. Compute $\mathsf{E}(\sum_{j=1}^{d(\gamma-1)} \hat{v}_j^2, r)$ with a random $r \in \mathbb{Z}_N$.

e. Compute

$$\begin{aligned} \mathsf{E}(\sum_{j=1}^{d(\gamma-1)} \hat{v}_j^2 - 2\hat{\mathbf{u}} \cdot \hat{\mathbf{v}}, rs) &= \mathsf{E}(\sum_{j=1}^{d(\gamma-1)} \hat{v}_j^2 + (N-2)\hat{\mathbf{u}} \cdot \hat{\mathbf{v}}, rs) \\ &= \mathsf{E}(\sum_{j=1}^{d(\gamma-1)} \hat{v}_j^2, r) \cdot \mathsf{E}((N-2)\hat{\mathbf{u}} \cdot \hat{\mathbf{v}}, s) \mod N^2 , \end{aligned} \tag{5.7}$$

and send it back to Alice. Note that the first equality sign is because $\hat{v}_j^2 - 2\hat{\mathbf{u}} \cdot \hat{\mathbf{v}} = \hat{v}_j^2 + (N-2)\hat{\mathbf{u}} \cdot \hat{\mathbf{v}} \mod N$, and that the second is again due to the homomorphic property of the Paillier cryptosystem.

3. Alice decrypts $\mathsf{E}(\sum_{j=1}^{d(\gamma-1)} \hat{v}_j^2 - 2\hat{\mathbf{u}} \cdot \hat{\mathbf{v}}, rs)$ using her private key to get $\sum_{j=1}^{d(\gamma-1)} \hat{v}_j^2 - 2\hat{\mathbf{u}} \cdot \hat{\mathbf{v}}$ and finally computes

$$\ell_1(\mathbf{u}, \mathbf{v}) = \sum_{j=1}^{d(\gamma-1)} \hat{v}_j^2 - 2\hat{\mathbf{u}} \cdot \hat{\mathbf{v}} + \sum_{j=1}^{d(\gamma-1)} \hat{u}_j^2. \tag{5.8}$$

5.3.1.2 Protocol Analysis

We now analyze the privacy provision of Protocol 1 and the related computation and communication overhead.

Theorem 5.3.1. Protocol 1 ensures level-I privacy if the Paillier cryptosystem is semantically secure and a personal profile is a vector of dimension $d \geq 2$.

Proof. Bob receives and operates only on ciphertexts $\{\mathsf{E}(\hat{u}_1, r_1)\}_{j=1}^{(\gamma-1)d}$ and does not know Alice's private key. Since the Paillier cryptosystem is semantically secure, computationally bounded Bob cannot decrypt the ciphertexts to learn anything about Alice's profile $\mathbf{u}$. As to Alice, she only get $\sum_{j=1}^{d(\gamma-1)} \hat{v}_j^2 - 2\hat{\mathbf{u}} \cdot \hat{\mathbf{v}}$. If she wants to find out Bob's profile $\mathbf{v}$, she must solve an equation with d unknowns, which is infeasible for $d \geq 2$. Therefore, Alice knows nothing about $\mathbf{v}$ other than the result $\ell_1(\mathbf{u}, \mathbf{v})$. □

The computation overhead incurred by Protocol 1 is mainly related to modular exponentiations and multiplications. In particular, Alice need perform $(\gamma - 1)d$ Paillier encryptions in Step 1.a, each costing two 1024-bit exponentiations and one 2048-bit multiplication according to Eq. (5.1). Note that Alice can preselect many random numbers and precompute the corresponding ciphertexts in an offline manner to reduce the online matching time.[3] In addition, Alice need perform one Paillier decryption in Step 3, which is essentially a 2048-bit exponentiation. As for Bob, he need perform $\sum_{i=1}^{d} v_i - 1$ 2048-bit multiplications in Step 2.b, one 2048-bit exponentiation in

[3] Alice can even do such offline computations on her regular computer and then synchronize the results to her mobile device.

Step 2.c, two 1024-bit exponentiations and one 2048-bit multiplication (i.e., one Paillier encryption) in Step 2.d, and one 2048-bit multiplication in Step 2.e. Considering Alice and Bob together, we can approximate the online computation cost of Protocol 1 to be $\sum_{i=1}^{d} v_i + 1$ 2048-bit multiplications, two 2048-bit exponentiations, and two 1024-bit exponentiations. In contrast, a direct application of the secure dot-protocol in [91] will require Bob to perform totally $(\gamma - 1)d - 1$ more 2048-bit exponentiations in Steps 2.b and 2.c.

The communication overhead incurred by Protocol 1 involves Alice sending her public key $\langle N, g \rangle$ and $(\gamma - 1)d$ ciphertexts in Step 1.c and Bob returning one ciphertext in Step 2.e. Since a public key and a ciphertext are of 1184 and 2048 bits, respectively, the total net communication cost of Protocol 1 is of $2048(\gamma - 1)d + 3232$ bits without considering message headers and other fields.

5.3.2 Protocol 2 for Level-II Privacy

We now introduce Protocol 2 which can satisfy level-II privacy. In contrast to Protocol 1 working only for the ℓ_1 distance, Protocol 2 can apply to a family of additively separable matching metrics and also hide the matching metric chosen by one user from the other. The secrecy of a user's selected matching metric can help prevent an attacker from generating better tailored profiles to deceive the victim user into a successful matching.

To illustrate Protocol 2, we first introduce the definition of *additively separable* functions as follows.

Definition 5.3.1. *A function $f(\mathbf{u}, \mathbf{v})$ is additively separable if it can be written as $f(\mathbf{u}, \mathbf{v}) = \sum_{i=1}^{d} f_i(u_i, v_i)$ for some functions $f_1(\cdot), \ldots, f_n(\cdot)$.*

Many common matching metrics are additively separable. For example, the ℓ_1 distance can be written as $\ell_1(\mathbf{u}, \mathbf{v}) = \sum_{i=1}^{d} |u_i - v_i|$, the dot product is $\mathbf{u} \cdot \mathbf{v} = \sum_{i=1}^{d} u_i v_i$, and the ℓ_α norm is $\ell_\alpha^\alpha = \sum_{i=1}^{d} |u_i - v_i|^\alpha$. In addition, assuming that Alice assigns a weight w_i to attribute i, we can define the weighted ℓ_1 distance as $\sum_{i=1}^{d} w_i |u_i - v_i|$ which is also additively separable.

Protocol 2 works by first converting any additively separable function into a dot-product computation. In particular, given an additively separable similarity function f of interest, Alice constructs a vector $\tilde{\mathbf{u}} = \langle \tilde{u}_1, \ldots, \tilde{u}_{\gamma d} \rangle$, where $\tilde{u}_j = f_i(u_i, k)$, $i = \lfloor (j-1)/\gamma \rfloor + 1$, and $k = (j-1) \mod \gamma$, for all $j \in [1, \gamma d]$. Assume that Bob also relies on his profile $\mathbf{v}$ to construct a binary vector $\tilde{\mathbf{v}} = (\tilde{v}_1, \ldots, \tilde{v}_{\gamma d})$, where the jth bit $\tilde{v}_j$ equals one for all $j \in \mathcal{J}'_{\mathbf{v}} = \{j | j = (i-1)\gamma + v_i + 1, 1 \leq i \leq d\}$ and zero otherwise. It follows that $\tilde{u}_j \tilde{v}_j = \tilde{u}_j = f_i(u_i, v_i)$ for all $j \in \mathcal{J}'_{\mathbf{v}}$ and zero otherwise. We then can easily obtain the following result.

$$\begin{aligned} f(\mathbf{u}, \mathbf{v}) &= \sum_{i=1}^{d} f_i(u_i, v_i) \\ &= \sum_{j \in \mathcal{J}'_{\mathbf{v}}} \tilde{u}_j \\ &= \sum_{j=1}^{\gamma d} \tilde{u}_j \tilde{v}_j = \tilde{\mathbf{u}} \cdot \tilde{\mathbf{v}} \end{aligned} \tag{5.9}$$

So we can let Alice run a secure dot-protocol protocol with Bob to obtain $\tilde{\mathbf{u}} \cdot \tilde{\mathbf{v}} = f(\mathbf{u}, \mathbf{v})$ without disclosing $\mathbf{u}$ or f to Bob.

5.3.2.1 Protocol Details

The detailed operations of Protocol 2 are as follows.

1. Alice first constructs a vector $\tilde{\mathbf{u}}$ as discussed above and then chooses a distinct random $r_j \in \mathbb{Z}_N$ to compute $\mathsf{E}(\tilde{u}_j, r_j)$ for all $j \in [1, \gamma d]$ using her public key. Finally, she sends $\{\mathsf{E}(\tilde{u}_j, r_j)\}_{j=1}^{\gamma d}$ and her public key to Bob.

2. Bob constructs a vector $\tilde{\mathbf{v}}$ as described above after receiving Alice's message. He then computes

$$\begin{aligned} \mathsf{E}(\tilde{\mathbf{u}} \cdot \tilde{\mathbf{v}}, \prod_{j \in \mathcal{J}'_{\mathbf{v}}} r_j) &= \mathsf{E}(\sum_{j \in \mathcal{J}'_{\mathbf{v}}} \hat{u}_j, \prod_{j \in \mathcal{J}'_{\mathbf{v}}} r_j) \\ &= \prod_{j \in \mathcal{J}'_{\mathbf{v}}} \mathsf{E}(\hat{u}_j, r_j) \mod N^2 , \end{aligned} \tag{5.10}$$

which holds due to Eq. (5.9) and the homogenous property of the Paillier cryptosystem. Next, he selects a random number $r_B \in \mathbb{Z}_N$ to compute

$$\mathsf{E}(\tilde{\mathbf{u}} \cdot \tilde{\mathbf{v}}, r_B \prod_{j \in \mathcal{J}'_{\mathbf{v}}} r_j) = \mathsf{E}(\tilde{\mathbf{u}} \cdot \tilde{\mathbf{v}}, \prod_{j \in \mathcal{J}'_{\mathbf{v}}} r_j) \cdot r_B^N \mod N^2 , \tag{5.11}$$

which holds due to the self-blinding property of the Paillier cryptosystem introduced in Chapter 5.2.3. Finally, Bob returns $\mathsf{E}(\tilde{\mathbf{u}} \cdot \tilde{\mathbf{v}}, r_B \prod_{j \in \mathcal{J}'_{\mathbf{v}}} r_j)$ to Alice.

3. Alice uses her private key to decrypt $\mathsf{E}(\tilde{\mathbf{u}} \cdot \tilde{\mathbf{v}}, r_B \prod_{j \in \mathcal{J}'_{\mathbf{v}}} r_j)$ and finally get $\tilde{\mathbf{u}} \cdot \tilde{\mathbf{v}}$, i.e., $f(\mathbf{u}, \mathbf{v})$.

Note that it is necessary for Bob to perform one more encryption in Step.2 using a random number r_B unknown to Alice. Otherwise, Alice may be able to easily infer Bob's profile $\mathbf{v}$ by purposefully choosing her profile $\mathbf{v}$ and random numbers $\{r_j\}_{j=1}^{\gamma d}$.

5.3.2.2 Protocol Analysis

We now analyze the privacy provision of Protocol 2 and the related computation and communication overhead.

Theorem 5.3.2. Protocol 2 ensures level-II privacy if the Paillier cryptosystem is semantically secure and a personal profile is a vector of dimension $d \geq 2$.

Proof. The proof is similar to that of Theorem 5.3.1 except the additional point that Bob does not know the matching metric f employed by Alice. It is thus omitted here for lack of space. □

The computation overhead incurred by Protocol 2 also mainly relates to modular exponentiations and multiplications. In particular, Alice need perform γd Paillier encryptions in Step 1, each requiring two 1024-bit exponentiations and one 2048-bit multiplication. As in Protocol 1, Alice can do these encryptions beforehand in an offline manner. In addition, Alice need do one Paillier decryption in Step 3, corresponding to one 2048-bit exponentiation. Moreover, Bob need perform $d - 1 = |\mathcal{J}'_{\mathbf{v}}| - 1$ 2048-bit

multiplications in Eq. (5.10) plus one 1024-bit exponentiation and one 2048-bit multiplication in Eq. (5.11). In summary, the total online computation overhead of Protocol 2 can be approximated by d 2048-bit multiplications, one 2048-bit exponentiation, and one 1024-bit exponentiation.

The communication overhead incurred by Protocol 2 involves Alice sending her public key $\langle N, g \rangle$ and γd ciphertexts in Step 1 and Bob returning one ciphertext in Step 2. Similar to that of Protocol 1, the total net communication cost of Protocol 2 can be computed as $2048(\gamma d + 1) + 1184$ bits without considering message headers and other fields.

5.3.3 Protocol 3 for Level-III Privacy

Protocol 3 is designed to offer level-III privacy. In contrast to Protocol 2, it only lets Alice know whether $f(\mathbf{u}, \mathbf{v})$ is smaller than a threshold τ_A of her own choice, while hiding $f(\mathbf{u}, \mathbf{v})$ from her. Protocol 3 is desirable if Bob does not want Alice to know the actual similarity score $f(\mathbf{u}, \mathbf{v})$ between their profiles.

Protocol 3 is based on a special trick. In particular, assuming that there are three arbitrary integers δ, δ_1, and δ_2 such that $\delta > \delta_1 > \delta_2 \geq 0$, we have $0 < (\delta_1 - \delta_2)/\delta < 1$. Since we assume $f(\mathbf{u}, \mathbf{v})$ and τ_A both to be integers, $f(\mathbf{u}, \mathbf{v}) < \tau_A$ is equivalent to $f(\mathbf{u}, \mathbf{v}) + (\delta_1 - \delta_2)/\delta < \tau_A$ and thus $\delta f(\mathbf{u}, \mathbf{v}) + \delta_1 < \delta\tau_A + \delta_2$. On the other hand, if $f(\mathbf{u}, \mathbf{v}) \geq \tau_A$, we would have $f(\mathbf{u}, \mathbf{v}) + (\delta_1 - \delta_2)/\delta > \tau_A$. According to this observation, Bob can choose random δ, δ_1, and δ_2 unknown to Alice and then send encrypted $\delta f(\mathbf{u}, \mathbf{v}) + \delta_1$ and $\delta\tau_A + \delta_2$ to Alice. After decrypting the ciphtertexts, Alice can check whether $\delta f(\mathbf{u}, \mathbf{v}) + \delta_1$ is smaller than $\delta\tau_A + \delta_2$ to learn whether $f(\mathbf{u}, \mathbf{v}) < \tau_A$.

5.3.3.1 Protocol Details

The detailed operations of Protocol 3 are as follows.

1. Alice first constructs a vector $\tilde{\mathbf{u}}$ as in Step 1 of Protocol 2. She then chooses a distinct random $r_j \in \mathbb{Z}_N$ to compute $\mathsf{E}(\tilde{u}_j, r_j)$ for all $j \in [1, \gamma d]$ and also another

distinct random r_{τ_A} to compute $\mathsf{E}(\tau_A, r_{\tau_A})$. Finally, she sends $\{\mathsf{E}(\tilde{u}_j, r_j)\}_{j=1}^{\gamma d}$, $\mathsf{E}(\tau_A, r_{\tau_A})$, and her public key to Bob.

2. Bob first construct a binary vector $\tilde{\mathbf{v}}$ whereby to compute $\mathsf{E}(\tilde{\mathbf{u}} \cdot \tilde{\mathbf{v}}, \prod_{j \in \mathcal{J}'_{\mathbf{v}}} r_j)$ (i.e., $\mathsf{E}(f(\mathbf{u}, \mathbf{v}), \prod_{j \in \mathcal{J}'_{\mathbf{v}}} r_j)$) as in Step 2 of Protocol 2. He then randomly choose $r'_1, r'_2, \delta, \delta_1, \delta_2 \in \mathbb{Z}_N$ such that $\delta > \delta_1 > \delta_2$ to compute

$$\mathsf{E}(\delta \tilde{\mathbf{u}} \cdot \tilde{\mathbf{v}} + \delta_1, s_1) = \mathsf{E}(\tilde{\mathbf{u}} \cdot \tilde{\mathbf{v}}, \prod_{j \in \mathcal{J}'_{\mathbf{v}}} r_j)^{\delta} \mathsf{E}(\delta_1, r'_1) \mod N^2 \tag{5.12}$$

and

$$\mathsf{E}(\delta \tau_A + \delta_2, r'_2 r_{\tau_A}) = \mathsf{E}(\tau_A, r_{\tau_A})^{\delta} \cdot \mathsf{E}(\delta_2, r'_2) \mod N^2 , \tag{5.13}$$

where $s_1 = r'_1 (\prod_{j \in \mathcal{J}'_{\mathbf{v}}} r_j)^{\delta} \mod N$. Both equations hold due to the homomorphic property of the Paillier cryptosystem. Finally, Bob returns $\mathsf{E}(\delta \tilde{\mathbf{u}} \tilde{\mathbf{v}} + \delta_1, s_1)$ and $\mathsf{E}(\delta \tau_A + \delta_2, r'_2 r_{\tau_A})$ to Alice.

3. Alice decrypts the ciphertexts to get $\delta \tilde{\mathbf{u}} \cdot \tilde{\mathbf{v}} + \delta_1$ and $\delta \tau_A + \delta_2$. If the former is smaller than the latter, Alice knows $f(\mathbf{u}, \mathbf{v}) < \tau_A$. Otherwise, she knows $f(\mathbf{u}, \mathbf{v}) \geq \tau_A$.

5.3.3.2 Protocol Analysis

We now analyze the privacy provision of Protocol 3 and the related computation and communication overhead.

Theorem 5.3.3. Protocol 3 ensures level-III privacy if the Paillier cryptosystem is semantically secure and a personal profile is a vector of dimension $d \geq 2$.

Proof. The proof is similar to that of Theorem 5.3.2 except the additional points that Bob does not know Alice's threshold τ_A and that Alice does know the comparison result $f(\mathbf{u}, \mathbf{v})$. □

The computation overhead incurred by Protocol 3 also mainly relates to modular exponentiations and multiplications. In particular, Alice need perform $\gamma d + 1$ Paillier encryptions in Step 1, each requiring two 1024-bit exponentiations and one 2048-bit multiplication. As in Protocol 2, Alice can do these encryptions beforehand in an offline

manner. In addition, Alice need do two Paillier decryptions in Step 3, each corresponding to one 2048-bit exponentiation. Moreover, Bob need perform $d - 1 = |\mathcal{J}'_{\mathbf{v}}| - 1$ 2048-bit multiplications in Eq. (5.10) plus one 1024-bit exponentiation and one 2048-bit multiplication in Eq. (5.11). Furthermore, Bob need do one 2048-bit exponentiation, one Paillier encryption, and one 2048-bit multiplication in each of Eqs. (5.12) and (5.13). In summary, the total online computation cost of Protocol 3 is $d+3$ 2048-bit multiplications, four 2048-bit exponentiations, and four 1024-bit exponentiations.

The communication overhead incurred by Protocol 3 involves Alice sending her public key $\langle N, g\rangle$ and $\gamma d + 1$ ciphertexts in Step 1 and Bob returning two ciphertexts in Step 2. Similar to that of Protocol 2, the total net communication cost of Protocol 3 can be computed as $2048(\gamma d + 3) + 1184$ bits without considering message headers and other fields.

5.4 Extension: MAX-Distance Matching

In this section, we present another private-matching protocol based on the MAX distance. Given two personal profiles $\mathbf{u}$ and $\mathbf{v}$, the MAX distance between them is defined as follows.

$$\ell_{\max}(\mathbf{u}, \mathbf{v}) = \max\{|v_1 - u_1|, \ldots, |v_d - u_d|\} \tag{5.14}$$

Protocols 1 to 3 all enable a user to check whether the overall absolute difference between her and another user's profiles is below a personally chosen threshold. In contrast, Protocol 4 allows the user to check whether the maximum attribute-wise absolute difference does not exceed her personal threshold.

At the first glance, $\ell_{\max}(\mathbf{u}, \mathbf{v})$ is not an additively separable function, so it cannot be computed using Protocol 2 or 3. In what follows, we first show how to convert $\ell_{\max}(\mathbf{u}, \mathbf{v})$ into an additively separable function based on a concept called *similarity matching* and then present the protocol details and analysis.

5.4.1 From the MAX Distance to an Additively Separable Function

The conversion from $\ell_{\max}(\mathbf{u}, \mathbf{v})$ to an additively separable function relies on similarity matching defined as follows.

Definition 5.4.1. *Given two user's personal profiles* $\mathbf{u} = \langle u_1, \ldots, u_d \rangle$ *and* $\mathbf{v} = \langle v_1, \ldots, v_d \rangle$*, their* i *attributes are considered* ***similar*** *if* $|u_i - v_i| \leq \tau$ *for a specific threshold* τ*.*

Definition 5.4.2. *The* ***similarity score*** *of* $\mathbf{u}$ *and* $\mathbf{v}$*, denoted by* $\Phi(\mathbf{u}, \mathbf{v}, \tau)$*, is defined as the total number of similar attributes, i.e.,*

$$\Phi(\mathbf{u}, \mathbf{v}, \tau) = \sum_{i=1}^{d} \phi(u_i, v_i, \tau) \ ,$$

where

$$\phi(u_i, v_i, \tau) = \begin{cases} 1 & \textit{if } |u_i - v_i| \leq \tau, \\ 0 & \textit{otherwise} \ . \end{cases}$$

The similarity score has three essential properties. First, it is additively separable, implying that Alice can run Protocol 2 with Bob to compute $\Phi(\mathbf{u}, \mathbf{v}, \tau)$ or Protocol 3 to check whether $\Phi(\mathbf{u}, \mathbf{v}, \tau) < \tau_A$. Second, it is directly affected by the value of τ. In particular, the larger τ, the higher $\Phi(\mathbf{u}, \mathbf{v}, \tau)$, and vice versa. Last, it relates to $\ell_{\max}(\mathbf{u}, \mathbf{v})$ based on the following theorem.

Theorem 5.4.1. For all $\tau \geq \ell_{\max}(\mathbf{u}, \mathbf{v})$, we have $\Phi(\mathbf{u}, \mathbf{v}, \tau) = d$; likewise, for all $\tau < \ell_{\max}(\mathbf{u}, \mathbf{v})$, we have $\Phi(\mathbf{u}, \mathbf{v}, \tau) < d$.

Proof. By the definition of the MAX distance, we have $|u_i - v_i| \leq \ell_{\max}(\mathbf{u}, \mathbf{v})$ for all $1 \leq i \leq d$. It follows that $\phi(u_i, v_i, \tau) = 1$ for all $1 \leq i \leq d$ if $\tau \geq \ell_{\max}(\mathbf{u}, \mathbf{v})$. Therefore, we have therefore have $\mathbf{s}(\mathbf{u}, \mathbf{v}, \tau) = d$ for all $\tau \geq \ell_{\max}(\mathbf{u}, \mathbf{v})$. Similarly, by the definition of MAX distance, there exists k such that $|u_k - v_k| = \ell_{\max}(\mathbf{u}, \mathbf{v})$. It follows that $\phi(u_k, v_k, \tau) = 0$, so we have $\Phi(\mathbf{u}, \mathbf{v}, \tau) < d$ for all $\tau < \ell_{\max}(\mathbf{u}, \mathbf{v})$. □

5.4.2 Protocol 4: MAX-Distance Matching for Level-III Privacy

Protocol 4 depends on Protocol 3 for level-III privacy. Let $\tau_{\max}$ to denote Alice's MAX-distance threshold kept secret from Bob. According to Theorem 4, checking whether $\ell_{\max}(\mathbf{u}, \mathbf{v}) < \tau_{\max}$ is equivalent to checking whether $\Phi(\mathbf{u}, \mathbf{v}, \tau_{\max}) = d$.

5.4.2.1 Protocol Details

1. Alice first constructs a vector $\tilde{\mathbf{u}} = \langle \tilde{u}_1, \ldots, \tilde{u}_{\gamma d} \rangle$, where $\tilde{u}_j = \phi_i(u_i, k, \tau_{\max})$, $i = \lfloor j/\gamma \rfloor + 1$, and $k = (j-1) \mod \gamma$ for all $j \in [1, \gamma d]$. He then chooses a random $r_{\max} \in \mathbb{Z}_N$ to compute E$(d, r_{\max})$ and a distinct random $r_j \in \mathbb{Z}_N$ to compute E$(\tilde{u}_j, r_j)$ for all $j \in [1, \gamma d]$. Finally, she sends $\{$E$(\tilde{u}_j, r_j)\}_{j=1}^{\gamma d}$, E$(d, r_{\tau_{\max}})$, and her public key to Bob.

2. Bob performs almost the same operations as in Step 2 of Protocol 2 (except replacing r_A by $r_{\max}$) and returns E$(\delta \tilde{\mathbf{u}} \cdot \tilde{\mathbf{v}} + \delta_1, s_1)$ and E$(\delta d + \delta_2, r_2' r_{\tau_{\max}})$ to Alice. As in Protocol 2, we have $\tilde{\mathbf{u}} \cdot \tilde{\mathbf{v}} = \Phi(\mathbf{u}, \mathbf{v}, \tau_{\max})$.

3. Alice does the same as in Step 3 of Protocol 3 to check whether $\delta \tilde{\mathbf{u}} \cdot \tilde{\mathbf{v}} + \delta_1 < \delta d + \delta_2$. If so, she learns $\tilde{\mathbf{u}} \cdot \tilde{\mathbf{v}} < d$ (i.e., $\Phi(\mathbf{u}, \mathbf{v}, \tau_{\max}) < d$) and thus $\ell_{\max}(\mathbf{u}, \mathbf{v}) > \tau_{\max}$. Otherwise, Alice knows $\ell_{\max}(\mathbf{u}, \mathbf{v}) \leq \tau_{\max}$.

Since Protocol 4 is a special case of Protocol 3 and thus can also ensure level-III privacy with the same communication and computation overhead as that of Protocol 3.

Table 5.1: Comparison of private-matching protocols

Protocol	Metric	Privacy	Offline Comp.	Online Comp.	Comm. (in bits)
RSV [91]	$\ell_1(\mathbf{u}, \mathbf{v})$	Level-I	$2(\gamma-1)d$ exp$_1$, $(\gamma-1)d$ mul$_2$	$(\gamma-1)d+3$ exp$_1$, $(\gamma-1)d+3$ mul$_2$	$2048(\gamma-1)d+3232$
Protocol 1	$\ell_1(\mathbf{u}, \mathbf{v})$	Level-I	$2(\gamma-1)d$ exp$_1$, $(\gamma-1)d$ mul$_2$	2 exp$_2$, 2 exp$_1$, $\sum_{j=1}^{d} v_i + 1$ mul$_2$	$2048(\gamma-1)d+3232$
Protocol 2	$f(\mathbf{u}, \mathbf{v})$	Level-II	$2\gamma d$ exp$_1$, γd mul$_2$	1 exp$_2$, 1 exp$_1$, d mul$_2$	$2048\gamma d+3232$
Protocol 3	$f(\mathbf{u}, \mathbf{v}) < \tau$	Level-III	$2\gamma d+2$ exp$_1$, $\gamma d+1$ mul$_2$	4 exp$_2$, 4 exp$_1$, $d+3$ mul$_2$	$2048\gamma d+7328$
Protocol 4	$\ell_{\max}(\mathbf{u}, \mathbf{v}) < \tau$	Level-III	$2\gamma d+2$ exp$_1$, $\gamma d+1$ mul$_2$	4 exp$_2$, 4 exp$_1$, $d+3$ mul$_2$	$2048\gamma d+7328$

5.5 Performance Evaluation

In this section, we evaluate the communication and computation overhead as well as overall execution time of our protocols in contrast to previous work. Since previous work [6,21,65,69] on private matching for PMSN is not applicable to fine-grained private matching addressed by our protocols, we only compare our work with RSV, which refers to the ℓ_1 distance protocol in [91] and can satisfy level-I privacy. We are not aware of any existing work that can offer level-III privacy as our Protocol 3. We omit the rather straightforward derivation process for the communication and computation costs of RSV and refer interested readers to [91] for details.

Table 5.1 summarizes the theoretical performance of Protocols 1$\sim$4 and RSV, where $\mathsf{mul}_1, \mathsf{mul}_2, \mathsf{exp}_1$, and exp_2 denote one 1024-bit multiplication, 2048-bit multiplication, 1024-bit exponentiation, and 2048-bit exponentiation, respectively. It is clear that all our protocols incur significantly lower online computation overhead than RSV with similar communication overhead.

5.5.1 Implementation

We implement our four protocols and RSV on LG P-970 smartphones, which has a 1GHz Cortex-A8 processor, 512 MB RAM, Android v2.2 Operating System, a 802.11 b/g/n WiFi interface, and Bluetooth v2.1 with Enhanced Data Rate (EDR). As in [21], we use a publicly available Java implementation of Paillier cryptosystem [51]. The whole PMSN application consists of 5000+ lines of Java code, in which our four protocols share the majority of the codes. Since Android platform currently does not support WiFi ad-hoc mode, we let two smartphones communicate with each other through Bluetooth. In our experiments, we are only able to achieve a transmission rate of approximately 800 kb/s, lthough Bluetooth v2.1 with EDR is expected to operate at a transmission rate of 2.1 Mb/s. If better Bluetooth implementations are available, the execution time of our protocols can be significantly reduced.

In our implementation, assuming that Alice initiates the matching protocol with Bob, each protocol consists of five main steps as follows.

1. Alice prepares the message through offline computation, e.g., generating a number of ciphertexts according to our protocol specifications;
2. Alice sends the message to indicate the start of the protocol;
3. Bob receives and buffers the message;
4. Once the transmission completes, Bob computes the intermediate result according to our protocol specifications, and sends it back to Alice;
5. On receiving the intermediate result, Alice computes the final matching result.

We use custom message headers in the application layer to distinguish these messages.

5.5.2 Experimental Results

We first measure the computation time of different basic operations of Paillier cryptosystem on LG P-970 and a Dell XPS 9100 desktop with Intel Core i7 920 2.6GHZ CPU, 9GB RAM, and Windows 7 Operating System. The desktop is used for offline computation. Tables 5.2 and 5.3 show the mean, maximum, minimum, medium, and standard deviation of the execution time of mul_1, mul_2, exp_1, exp_2, Enc, and Dec on different platforms, where Enc and Dec denote one Paillier encryption and one decryption, respectively, and each value is computed statistically from 10,000 runs. We can see that it takes much less time to perform the same operation on Dell XPS 9100 than on LG P-970. For example, one Paillier encryption takes on average 167.21 ms and 37.53 ms on LG P-970 and Dell XPS 9100, respectively. In what follows, we assume that the offline computation is performed on desktop and is not counted into the protocol execution time.

In our experiments, we generate random profiles with each having d attributes, where every attribute value is chosen from $[0, \gamma - 1]$ uniformly at random. The perfor-

Table 5.2: Execution time of different operations (ms) on LG P-970

Operation	Mean	Max	Min	Median	Std.
mul_1	0.73	40.25	0.58	0.61	1.16
exp_1	81.08	112.0	77.0	78.0	6.22
mul_2	0.88	43.00	0.73	0.76	1.14
exp_2	159.06	197.0	153.0	154.0	9.75
Enc	167.21	295.0	158.0	159.0	17.53
Dec	165.6	227.0	160.0	161.0	10.27

Table 5.3: Execution time of different operations (ms) on Dell XPS 9100

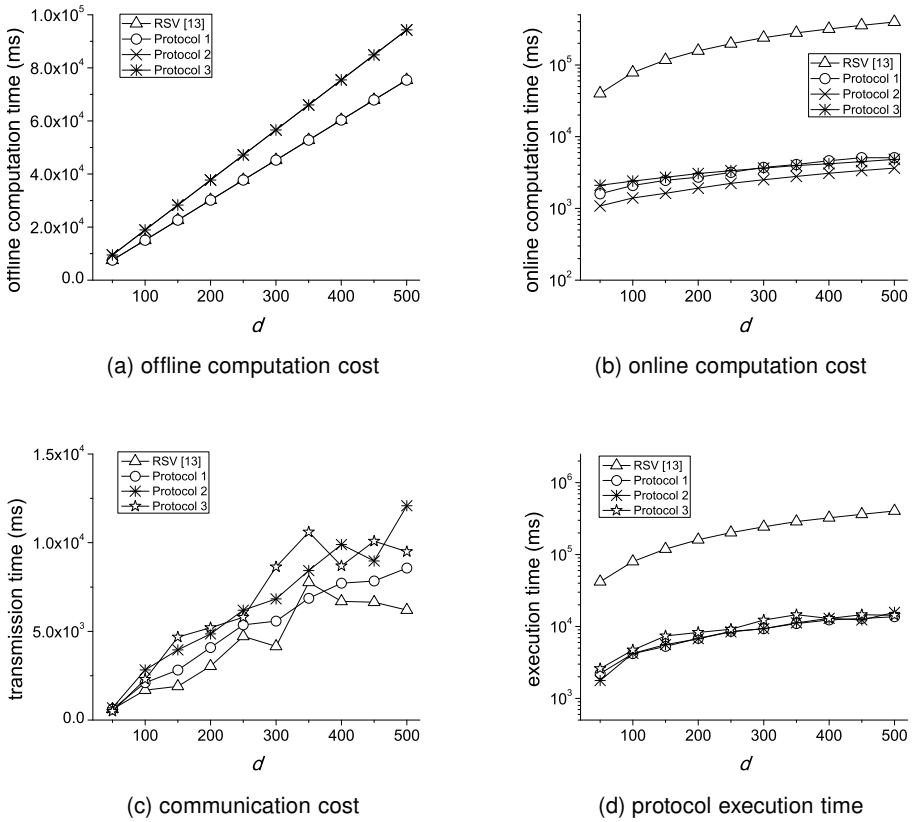
Operation	Mean	Max	Min	Median	Std.
mul_1	0.0076	0.10	0.0042	0.0062	0.0055
exp_1	18.84	28.0	17.0	18.0	1.54
mul_2	0.033	0.28	0.031	0.031	0.0080
exp_2	36.26	40.0	34.0	36.0	1.46
Enc	37.53	40.0	35.0	37.0	1.16
Dec	37.66	41.0	36.0	37.0	1.20

mance metrics used include the offline computation time on the desktop, online computation time on the smartphone, the total net communication cost in bits, and the total online execution time including the online computation, communication, and internal processing time. Note that a complete matching process involves two independent executions of the same or even different private-matching protocols, initiated by Alice and Bob, respectively. For simplicity, we assume that Alice and Bob choose the same protocol and only show the results for one protocol execution. The total matching time thus should be twice the shown total online execution time. Finally, since Protocol 4 has the same communication and computation overhead as Protocol 3, its performance results are not shown for brevity.

5.5.2.1 Impact of d

We first check the case when $\gamma = 5$ and d varies. It is not surprising to see from Fig. 5.1a that the offline computation costs of all the four protocols are proportional to d. In addition, Protocols 2 and 3 incur comparable offline computation overhead higher than that of Protocol 1 and RSV which incur the same computation overhead. The main reason is that both Protocols 2 and 3 require $\gamma d + 1$ offline Paillier encryptions,[4] while

[4]Recall that one Paillier encryption corresponds to two 1024-bit exponentiation and one 2048-bit multiplication.



(a) offline computation cost

(b) online computation cost

(c) communication cost

(d) protocol execution time

Figure 5.1: Impact of the profile dimension d, where $\gamma = 5$.

RSV and Protocol 1 require $(\gamma - 1)d$. Since users can do such offline computations on their regular computers and then synchronize the results to their mobile devices, the offline computation cost thus does not contribute to the total protocol execution time.

Fig. 5.1b shows the online computation costs of all the protocols in the $\log 10$ scale for a fixed $\gamma = 5$ and varying d. It is clear that Protocols 1 to 3 all incur much lower online computation overhead than RSV. The main reasons are that 1024-bit and 2048-bit exponentiations dominate the online computation costs of all the protocols and that Protocols 1 to 3 all require a constantly small number of modular exponentiations, while RSV requires a much larger number of modular exponentiations that increases almost linearly with d.

Fig. 5.1c compares the total net communication costs of all the protocols for a fixed $\gamma = 5$ and varying d. We can see that all the protocols incur comparable communication costs which all increase almost linearly as d increases, which is of no surprise.

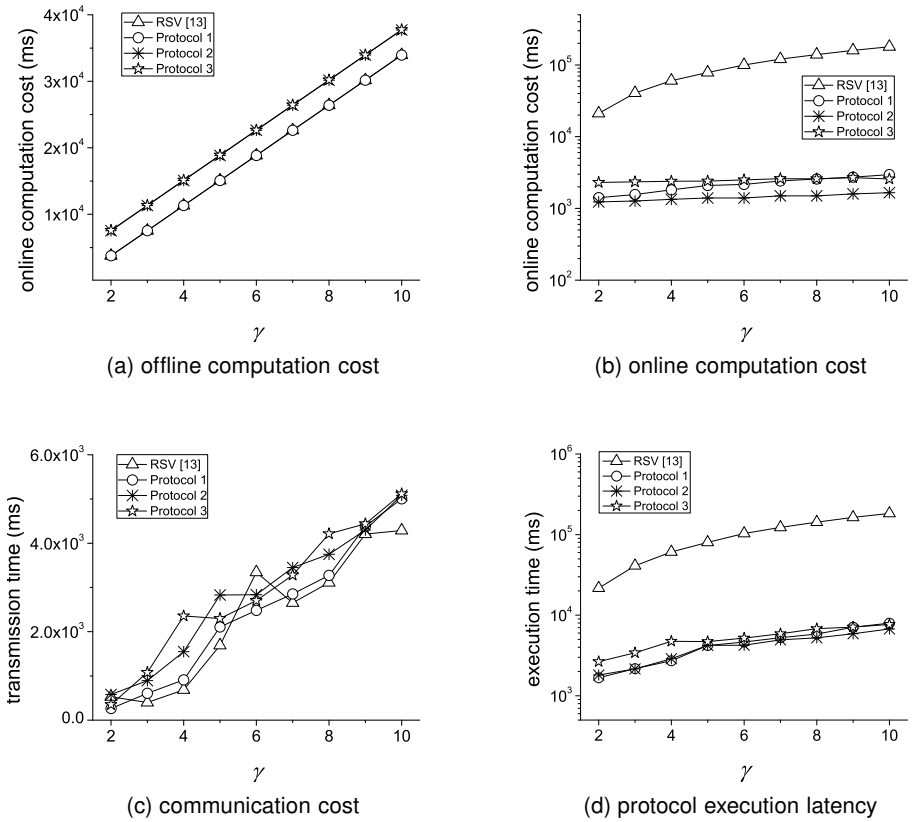
Fig. 5.1d shows the total protocol execution time for a fixed $\gamma = 5$, which comprise the online computation, communication, and internal processing time and is dominated by the former. We can see that there are some fluctuations in the protocol execution time, mainly due to unstable transmission rate of the Bluetooth interface. In addition, Protocol 2 has the shortest execution time among Protocols 1 to 3, while Protocol 3 has the longest for achieving level-III privacy. All our protocols, however, can finish within 15 seconds under all simulated scenarios in contrast to the much longer execution time required by RSV. For example, when $d = 100$, RSV require 80.1 seconds to finish, while Protocols 1 to 3 only require 4.2, 4.2, and 4.7 seconds, respectively. Recall that a complete private-matching process involves two protocol executions. Our three protocols are thus much more feasible and user-friendly solutions to private matching for PMSN.

5.5.2.2 Impact of γ

The impact of γ on the protocol performance is shown in Fig. 5.2, where d is fixed to be 100. Similar results can be observed as in Fig. 5.1. In particular, the online computation time of Protocols 1 to 3 are relatively insensitive to the increase in γ while that of RSV increases linearly as γ increases.

5.5.2.3 Energy Consumption

As in [21], we measure the energy consumption of our protocols using PowerTutor [131]. Table III shows that the energy consumption of one execution of RSV and Protocols 1 to 3, where $d = 100$ and $\gamma = 5$. We can see that all our protocols consumes about one eighth of the energy RSV does. In addition, a fully charged LG P-970 has 20,160J and one execution of any of our protocols only consumes less than 0.06% of the total



(a) offline computation cost (b) online computation cost

(c) communication cost (d) protocol execution latency

Figure 5.2: Impact of the highest attribute value γ, where $d = 200$.

Table 5.4: Comparison of energy consumption for four protocols, where $d = 100$ and $\gamma = 5$

	Energy Consumption (J)				
Protocol	**Alice**	**Percentage**	**Bob**	**Percentage**	**Total**
RSV [91]	100	0.46%	98	0.45%	198
Protocol 1	12	0.055%	9.6	0.047%	21.6
Protocol 2	12.3	0.057%	9.6	0.047%	21.9
Protocol 3	13.2	0.061%	13.9	0.064%	27.1

energy, which indicate that our private matching protocols are very practical in terms of power consumption.

5.5.3 Discussion

The experimental results show that all our protocols incur similar offline computation and communication overhead but significantly lower online computation overhead and thus

total protocol execution latency in comparison with RSV, making them more practical than RSV to realize private matching in PMSN.

To further reduce the total execution latency of our protocols, there are two directions to explore. First, since a significant portion of the total protocol execution time is the transmission time, it is possible to reduce the total execution latency by using advanced wireless interface with higher transmission rate. For example, Bluetooth 3.0 and 4.0 have a promised speed of 25 Mb/s, and Wi-Fi Direct has the maximum transmission rate of up to 250 Mb/s, while our current achievable transmission rate via Bluetooth interface on LG P-970 is only 800 kb/s. As more and more emerging mobile devices support these advanced interfaces, the transmission time and total protocol execution latency of our protocols will be significantly reduced. For instance, when $d = 100$ and $\gamma = 5$, with a transmission rate of 25 Mb/s, the total execution times of Protocols 1 to 3 will be close to the online computation time, i.e., 2.1s, 1.4s, and 2.4s, respectively. Second, our currently implementation uses the publicly available Java implementation of Paillier cryptosystem [51] without any optimization, further optimization is expected to further reduce the online computation time.

5.6 Related Work

In this section, we briefly discuss some work in several areas which is most germane to our work in this chapter.

Private matching for PMSN. As mentioned in Chapter 5.1, the private matching schemes proposed in [6, 65, 69] aim at coarse-grained personal profiles and match two users based on a privacy-preserving computation of the intersection (cardinality) of their attribute sets. In contrast, our protocols support fine-grained personal profiles and thus much finer user differentiation, which is important for fostering the much wider use of PMSN. To our best knowledge, Dong *et al.* presented the only piece of work in [21] that does not match two users in PMSN using the intersection (cardinality) of their attribute sets. Instead, they proposed using the social proximity between two users as the matching metric, which measures the distance between their social coordinates with

each being a vector precomputed by a trusted central server to represent the location of a user in an online social network. By comparison, our work does not rely on the affiliation of PMSN users with a single online social network and addresses a more general private matching problem for PMSN by supports fine-grained personal profiles and a wide spectrum of matching metrics.

Secure multi-party computation. Private matching for PMSN can also be viewed as special instances of secure two-party computation, which was initially introduced by Yao in [121] and later generalized to secure multi-party computation by Goldreich *et al.* [39] and many others. In secure two-party computation, two users with private inputs x and y, respectively, both want to compute some function $f(x, y)$ without any party learning information beyond what can be inferred from the result $f(x, y)$. It was shown that all secure multi-party computation problems can be solved using the general approach in using the general approach [39], which is nevertheless too inefficient to use in practice. The existing literature on secure multi-party computation thus focused on devising more efficient solutions for specific functions. Our work in this chapter belong to this category and gives efficient solutions to many PMSN matching metrics.

Privacy-preserving data mining and scientific computation. Securely computing some function over two vectors has also been investigated in the context of privacy-preserving data mining and scientific computation. In particular, secure dot-product computation was studied in [23, 38, 50, 97]. As in [21], we adopt the method in [38] as a component of our protocols and make significant contributions on relating the computation of many PMSN matching metrics to secure dot-product computation. Privacy-preserving correlation computation was studied in [57, 89] and is loosely related to our work here. Moreover, some novel methods were proposed in [91] for securely computing the approximate ℓ_1 distance of two private vectors. As said before, our Protocol 1 is adapted from the protocols [91] but with significantly smaller computation overhead. In addition, Du *et al.* proposed a set of protocols based on commutative encryptions for securely computing the difference between two private vectors based on different met-

rics [22], including the ℓ_1 distance, the ℓ_2 distance, and a more general function. Since all known commutative encryption schemes are deterministic, i.e., the same plaintext always leads to the same cyphertext, it is less secure than Paillier cryptosystem [65]. It is not clear how to apply their protocols to our problem here in an efficient and secure fashion.

5.7 Summary

In this chapter, we formulated the problem of fine-grained private (profile) matching for proximity-based mobile social networking and presented a suite of novel solutions that support a variety of private-matching metrics at different privacy levels. Detailed performance analysis and experimental evaluation confirmed the high efficiency of our protocols over prior work under various practical settings.

Chapter 6

SECURE CROWDSOURCING-BASED COOPERATIVE SPECTRUM SENSING

6.1 Introduction

Cooperative spectrum sensing (CSS) is a key function for dynamic spectrum access and is essential for avoiding interference with licensed primary users and identifying spectrum holes [4]. It relies on spatially distributed cognitive radio (CR) users to jointly detect the occupancy of a licensed channel in a specific location and time range. In contrast to spectrum sensing by individual CR users, CSS could largely mitigate many factors such as multipath fading and shadowing [4] and thus has considerably better performance.

Centralized CSS involves a centralized fusion center (FC) which instructs selected CR users to sense a specific channel and then makes a global decision about channel occupancy by aggregating received local sensing results. Local spectrum sensing at cooperative CR users normally relies on energy detection, matched filter detection, or cyclostationary-feature detection. Data fusion can be in the form of either soft or hard combination, which requires the CR users to report raw sensing data or local decisions to the FC, respectively. As in [16, 26, 53, 64, 73], we consider soft combining in this chapter.

A promising method for effective CSS over a large geographic region is to explore the emerging *crowdsourcing* paradigm, in which special *spectrum-sensing provider*s (SSPs) [26–28, 98] outsource spectrum-sensing tasks to distributed mobile users called *mobile detector*s who themselves may also be secondary CR users. The feasibility of crowdsourcing-based CSS (CCSS for short) is deeply rooted in the ubiquitous penetration of mobile devices into everyday life. Specifically, according to a recent Cisco report [1], the number of mobile devices such as smartphones and tablets will exceed the world population in 2012 and hit 10 billion in 2016, which implies sufficient geographic coverage especially in highly populated regions such as metropolitan areas. Moreover, they can always accurately self-localize based on hybrid GPS, WiFi, and cellular po-

sitioning techniques. Since dynamic spectrum access is expected to be pervasive in future wireless systems, it is widely expected that future mobile devices can perform spectrum sensing via either internal spectrum sensors or external ones acquired from other parties like the SSP [4, 26–28, 74].

CCSS, though appealing, is vulnerable to false sensing reports, each containing a power measurement much larger (or smaller) than the true value to inflate (or deflate) the final average. A false sensing report can come from a normal mobile detector with a faulty spectrum sensor, a dishonest one wishing to save energy by faking data without actual sensing, or a malicious one aiming to prevent other users from using the channel by submitting an extremely high (or low) power measurement. Without sound defenses in place, the SSP may be misled by false sensing reports to falsely determine that the channel is busy (or vacant). It is thus critical to ensure secure soft combination such that the impact of possible false sensing reports can be minimized.

The prior work on secure CSS against false sensing reports can be generally classified into three categories. The first category such as [26, 28, 53, 64, 73] uses various anomaly detection techniques to identify false sensing reports and would fail if they constitute the majority, as discussed in [27]. The second category such as [16, 53, 130] uses reputations to differentiate malicious mobile detectors from legitimate ones and is unable to handle sudden change in mobile detectors' behaviors. More recent work [27] relies on some trusted nodes to detect false sensing reports which requires real signal propagation data from primary users (PUs) that are often difficult to obtain. A sound soft combination scheme that can withstand a majority of malicious mobile detectors without too strong assumptions remains an open challenge.

As the first work of its kind, we propose a novel scheme to realize secure CCSS in the presence of malicious mobile detectors possibly being the majority without requiring real signal propagation data from PUs. Our scheme relies on using a few trusted *anchor* detectors to evaluate the instantaneous trustworthiness of mobile detectors in combination with their reputation scores. Our scheme can enable robust PU detection

even when the majority of mobile detectors are malicious as long as there are enough trustworthy sensing reports submitted from legitimate mobile detectors. Our contribution in this chapter can be summarized as follows.

- We propose a novel metric to measure the instantaneous trustworthiness of a sensing report based on trusted anchor detectors and the relationship between receiving power and distance.
- We design a novel secure soft combination scheme based on the prioritized weighted sequential probability ratio test, in which sensing reports are assigned different weights based on their reputation scores and prioritized according to their instantaneous trustworthiness.
- We confirm the high efficacy and efficiency of our scheme by extensive simulation studies.

The rest of this chapter is organized as follows. Section 6.2 introduces the system and adversary models. Section 6.3 presents our proposed solution. Section 6.4 reports the performance evaluation based on detailed simulation studies. Section 6.5 discusses the related work. Section 6.6 concludes this chapter.

6.2 System and Adversary Models

In this section, we introduce our system, signal propagation, local spectrum sensing, and adversary models.

6.2.1 System Model

Fig. 6.1 shows the CCSS architecture under consideration. The SSP divides its service region into equally-sized cells and deploys some *anchor* detectors at strategic locations, e.g., the corners or center of each cell. Similar to the trusted nodes in [27], anchor detectors can be remotely attested by the SSP and excluded if they are detected as compromised. Due to cost constraints, the SSP cannot afford to deploy too many anchor detectors. As a result, although anchor detectors can provide the most trustworthy

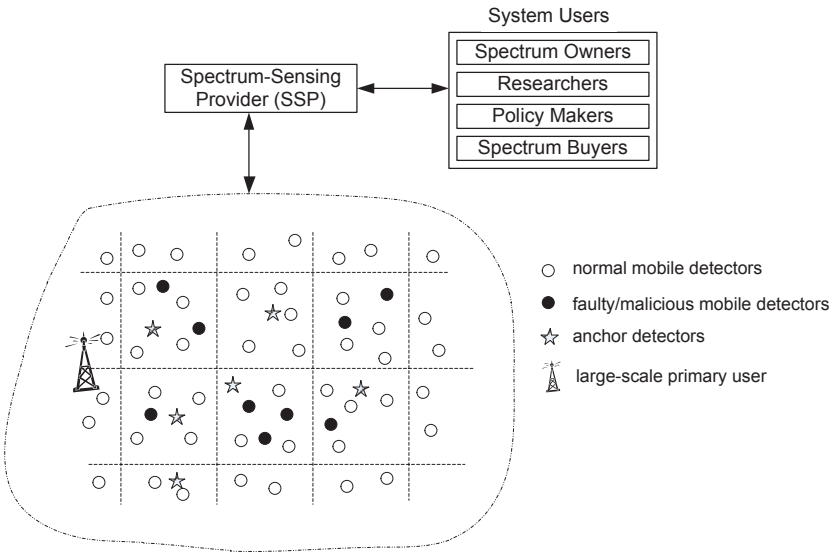


Figure 6.1: A crowdsourcing-based cooperative spectrum sensing architecture.

spectrum-sensing reports, the SSP still relies on the majority of mobile detectors to reach sufficiently high detection accuracy. Our subsequent discussion will focus on a cell with anchor detectors denoted by Θ_a and mobile detectors denoted by Θ, where $|\Theta_a| \ll |\Theta|$.

Mobile detectors correspond to humans using mobile devices such as smartphones and tablets to participate in CSS, and they can perform spectrum sensing via either spectrum sensors embedded into mobile devices or external ones which are provided by the SSP and can communicate with mobile devices. Finally, we assume that mobile detectors can accurately self-localize via hybrid GPS, cellular, and WiFi positioning techniques.

We consider large-scale PUs such as TV stations and cellular base stations (expected in the future [54, 78]) with a large transmission range and known fixed locations. Extending our work to support small-scale and/or mobile PUs such as wireless microphones is left as our future work.

6.2.2 Signal Propagation and Spectrum-sensing Models

We adopt the signal propagation model in [96], under which the received primary signal strength at mobile detector i can be expressed as

$$P_i = P_0(\frac{d_0}{d_i})^{\alpha} e^{X_i} e^{Y_i} \quad (\text{Watt}) \tag{6.1}$$

where d_0 is the reference distance, P_0 is the received primary signal strength at d_0, d_i is the distance from mobile detector i to the primary user, α is the pathloss exponent with typical value between 2 and 5, e^{X_i} and e^{Y_i} represent the effect of shadowing fading and multi-path fading, respectively, where $X_i \sim \mathcal{N}(0, \sigma^2)$.

Assuming that the channel bandwidth is much larger than the coherent bandwidth, the effect of multi-path fading is negligible, i.e., $Y_i = 0$ for all i. In addition, we assume that X_i and X_j are independent for all $i \neq j$, i.e., each mobile detector experiences i.i.d. Gaussian shadowing and fading, which holds when the distance between mobile detectors i and j exceeds decorrelation distance [5].

We assume energy detection for local spectrum sensing at mobile detectors, which is the most widely-used detection technique for its simplicity and efficiency. In particular, on receiving a sensing task from the SSP, each mobile detector collects m RSS (received signal strength) samples. The sensing report from detector i is denoted as $\mathbf{x}_i = (x_{i,1}, \cdots, x_{i,m})$. The test statistic of the energy detector is the average RSS (including the noise power), i.e., $S_i = \frac{1}{m}\sum_{k=1}^{m} x_{i,k}$, which can be approximated as a Gaussian random variable using the Central Limit Theorem (CLT) [100, 107] as

$$S_i \sim \begin{cases} \mathcal{N}(N_o, \frac{2N_o^2}{m}) & \mathcal{H}_0 : \text{Primary user is absent} \\ \mathcal{N}(N_o + \bar{P}_i, \frac{2(\bar{P}_i + N_o)^2}{m}) & \mathcal{H}_1 : \text{Primary user is present} \end{cases}, \tag{6.2}$$

where $\bar{P}_i = \mathsf{E}(P_i)$ is the average received power at detector i, and N_o is the noise power, e.g., -96 dBm for a 6MHz TV channel.

6.2.3 Adversary Model

We assume that the adversary is aware of our scheme and has full control over multiple malicious detectors who may launch the following attacks.

- A malicious mobile detector may report high RSS values when the primary signal is absent, aiming at increasing the probability of false alarm and preventing CR users from using the channel.
- A malicious mobile detector may also report low RSS values when the primary signal is present, aiming at increasing the probability of miss detection and causing increased interference to the primary user.

Malicious mobile detectors could be the majority in a cell. We, however, assume that there are enough normal detectors submitting faithful sensing reports. Otherwise, it is fundamentally difficult to realize robust PU detection with desired miss detection and false alarm probabilities.

It is beyond the scope of this chapter to consider other possible attacks against cooperative sensing. For example, a powerful adversary may jam the channel to prevent mobile detectors from communicating with the SSP. These attacks are not unique to cooperative sensing and can be mitigated by spread-spectrum techniques [88, 132].

6.3 Secure Combination for CCSS

In this section, we first outline our secure combination scheme for CCSS and then detail its design.

6.3.1 Overview

Our scheme relies on using trusted anchor detectors to evaluate the *instantaneous trustworthiness* of mobile detectors in combination with their *reputation scores*. The key insight is that a mobile detector's reputation score and the instantaneous trustworthiness of his sensing report have different trust implications. On the one hand, the reputation score is to predict his future performance based on his past long-term behavior

and is nevertheless incapable of handling sudden change in his current behavior. On the other hand, the instantaneous trustworthiness of his sensing report only reflects the level of fitting with the trusted reports from anchor detectors in the current sensing task while is unable to incorporate his long-term behavior. Although the reports from anchor detectors are trusted to have not undergone malicious modifications, they may be inaccurate due to possible multi-path fading/shadowing and other channel impairments. We thus propose to explore both the instantaneous trustworthiness and reputation scores of mobile detectors to realize robust PU detection.

Specifically, our scheme uses instantaneous trustworthiness and reputation scores of mobile detectors in different ways. To enable robust data fusion, we propose a prioritized weighted-probability-ratio test to combine sensing reports, in which the sensing reports are ordered according to their instantaneous trustworthiness and assigned different weights according to their reputation scores. The sensing reports are fed to the algorithm one at a time, and a decision is made when certain criterion is reached. By doing so, as long as there are sufficient normal mobile detectors, the final decision will not be misled even if malicious mobile detectors are the majority.

In what follows, we detail the design of our scheme, including *instantaneous trustworthiness measure*, *prioritized weighted sequential probability ratio test*, and *fine-grained reputation management*.

6.3.2 Instantaneous Trustworthiness Measure

We first introduce a novel metric to evaluate the instantaneous trustworthiness of any mobile detectors $i \in \Theta \cup \Theta_A$ (or equivalently, their reports). For convenience only, we abuse the notation by letting Θ and Θ_a denote the mobile and anchor detectors who all submitted a sensing report to the SSP, where the cardinality $|\Theta|$ is normally much larger than $|\Theta_a|$.

Our key insight can be explained as follows. According to Eqs. (6.1) and (6.2), we know that the receiving powers at two honest mobile detectors either are both close to noise if the primary user is absent, or satisfy certain condition with respect to their

distances to the primary user if the primary user is present. Consider any two mobile detectors i and j with their distances to the primary user d_i and d_j, respectively. Their test statistics are denoted by S_i and S_j, respectively, which are assumed to be independent Gaussian random variables. We define the following random variable

$$Z_{i,j} = \rho(d_i^\alpha(S_i - N_o) - d_j^\alpha(S_j - N_o)) \ , \tag{6.3}$$

where $\rho = \sqrt{\frac{1}{2(d_i^{2\alpha}+d_j^{2\alpha})}}$.

When the primary user is absent, we have

$$\mathsf{E}(Z_{i,j}|\mathcal{H}_0) = \mathsf{E}(\rho(d_i^\alpha(S_i - N_o) - d_j^\alpha(S_j - N_o))) = 0 \ ,$$

and

$$\begin{aligned}
\mathsf{VAR}(Z_{i,j}|\mathcal{H}_0) &= \rho^2\mathsf{VAR}(d_i^\alpha(S_i - N_o) - d_j^\alpha(S_j - N_o)) \\
&= \rho^2(\mathsf{VAR}(d_i^\alpha S_i) + \mathsf{VAR}(d_j^\alpha S_j)) \\
&= \frac{2d_i^{2\alpha}N_o^2 + 2d_j^{2\alpha}N_o^2}{2m(d_i^{2\alpha} + d_j^{2\alpha})} \\
&= \frac{N_o^2}{m} \ .
\end{aligned}$$

Similarly, when the primary user is present, we have

$$\begin{aligned}
\mathsf{E}(Z_{i,j}|\mathcal{H}_1) &= \mathsf{E}(\rho d_i^\alpha(S_i - N_o) - d_j^\alpha(S_j - N_o)) \\
&= \mathsf{E}(d_i^\alpha \bar{P}_i - d_j^\alpha \bar{P}_j) \\
&= \mathsf{E}(d_i^\alpha P_0(\frac{d_0}{d_i})^\alpha e^{X_i} e^{Y_i} - d_j^\alpha P_0(\frac{d_0}{d_j})^\alpha e^{X_j} e^{Y_j}) \\
&= \mathsf{E}(P_0 d_0^\alpha e^{X_i} - P_0 d_0 e^{X_j}) \\
&= P_0 d_0^\alpha(\mathsf{E}(e^{X_i}) - \mathsf{E}(e^{X_j})) \\
&= 0 \ ,
\end{aligned}$$

where the third equality holds because $Y_i = 0$ (cf. Section 6.2.2). Since FCC requires that unlicensed CR devices reliably detect incumbent signals at very low SNRs (e.g., as low as -22 dB in the IEEE 802.22 standard [99]), it is typically assumed that $N_o \gg P_i$

for all $i \in \Theta \cup \Theta_a$. It follows that

$$\begin{aligned}
\text{VAR}(Z_{i,j}|\mathcal{H}_1) &= \rho^2 \text{VAR}(d_i^\alpha(S_i - N_o) - d_j^\alpha(S_j - N_o)) \\
&= \rho^2(\text{VAR}(d_i^\alpha S_i) + \text{VAR}(d_j^\alpha S_j)) \\
&= \frac{2d_i^{2\alpha}(\bar{P}_i + N_o)^2 + 2d_j^{2\alpha}(\bar{P}_j + N_o)^2}{2m(d_i^{2\alpha} + d_j^{2\alpha})} \\
&\approx \frac{N_o^2}{m} \ .
\end{aligned}$$

Therefore, no matter whether the primary user is present or not, $Z_{i,j}$ is approximately Gaussian distributed with zero mean and variance N_o^2/m if the reports S_i and S_j are highly correlated with the distance d_i and d_j. Otherwise, the distribution of $Z_{i,j}$ is unpredictable.

Assume that S_j is provided by a trustworthy anchor detector $j \in \Theta_a$. We can thus assess the trustworthiness of S_i with regard to S_j through the likelihood of $Z_{i,j}$ being generated from the Gaussian distribution $\mathcal{N}(0, N_o^2/m)$. In particular, assume that detectors i and j report s_i and s_j as an observation of S_i and S_j, respectively, based on which the SSP constructs an observation $z_{i,j}$ of $Z_{i,j}$. The likelihood of $z_{i,j}$ being generated from $\mathcal{N}(0, N_o^2/m)$ is given by

$$\text{L}(z_{i,j}|\mathcal{N}(0, N_o^2/m)) = \frac{1}{\sqrt{2\pi N_o^2/m}} e^{-\frac{m z_{i,j}^2}{N_o^2}} \ , \tag{6.4}$$

which monotonically decreases as $|z_{i,j}|$ increases. Therefore, we define the *relative instantaneous trustworthiness* of s_i with regard to s_j as $|z_{i,j}|$. The smaller $|z_{i,j}|$, the more trustworthy s_i with regard to s_j, and vice versa. In addition, we have $|z_{j,j}| = 0$ for any anchor detector $j \in \Theta_a$.

We then measure the overall instantaneous trustworthiness of s_i by combining all the relative instantaneous trustworthiness values $\{|z_{i,j}|\}_{j \in \Theta_a}$. In particular, we view detector s_i's $|\Theta_a|$ relative instantaneous trustworthiness values as a point in the $|\Theta_a|$-dimensional space $P_i = (|z_{i,1}|, \cdots, |z_{i,|\Theta_a|}|)$ and define the overall instantaneous trustworthiness of s_i as the Euclidean distance between P_i and the origin, which is

given by

$$t_i = (\sum_{j \in \Theta_a} |z_{i,j}|^2)^{\frac{1}{2}} \ . \tag{6.5}$$

The smaller t_i, the more trustworthy of s_i, and vice versa.

We have a few remarks about the instantaneous trustworthiness measure t_i. First, when there is only one anchor detector, say j, we have $t_j = 0$ as $z_{j,j} = 0$, meaning that s_j is the most trustworthy sensing report. Second, it is possible that an anchor detector j generates a bad sensing report due to temporal channel impairments. In this case, a malicious mobile detector i with false sensing report may gain high relative instantaneous trustworthiness with regard to anchor detector j, i.e., low $|z_{i,j}|$. It is, however, impossible for him to simultaneously gain high instantaneous trustworthiness at the other anchor detectors. It is therefore necessary to have multiple anchor detectors. Finally, when the primary user is present, a malicious mobile detector may submit a false sensing report along with a falsified location aiming at cheating the SSP into computing a wrong but high instantaneous trustworthiness. Our instantaneous trustworthiness measure is resilient to this attack, as if the false sensing report and location could together lead to a lower t_i (i.e., high instantaneous trustworthiness), it is equivalent to a sensing report submitted by a good mobile detector i' at the reported location.

6.3.3 Prioritized Weighted Sequential Probability Ratio Test

Once the SSP evaluates the instantaneous trustworthiness of all anchor and mobile detectors in $\Theta \cup \Theta_a$, it applies the Weighted Sequential Probability Ratio Test (WSPRT) technique [16] to aggregate the sensing reports by prioritizing those with higher instantaneous trustworthiness and also assigning higher weights to those from detectors with higher reputation scores, which we call Prioritized Weighted Sequential Probability Ratio Test (PWSPRT).

To perform PWSPRT, the SSP first ranks all the sensing reports according to their instantaneous trustworthiness t_i in an ascending order. We then define the follow-

ing decision variable

$$\mathbb{V} = \sum_{i \in \underline{\Theta}} \ln(\frac{\mathsf{P}(S_i|\mathcal{H}_1)}{\mathsf{P}(S_i|\mathcal{H}_0)})^{\mathsf{w}_i} \ , \tag{6.6}$$

where $\mathsf{P}(S|\mathcal{H}_k)$ refers to the probability density function of a random variable S under $\mathcal{H}_k$ ($k = 0$ or 1), $\underline{\Theta}$ denotes a subset of detectors in $\Theta \cup \Theta_a$ whose reports have been aggregated, and $\mathsf{w}_i \in [0, 1]$ is the normalized reputation score of detector i used as the weight here, which will be explained in Section 6.3.4.

The SSP's decision is based on the following criterion:

- Accept $\mathcal{H}_1$ and terminate if $\mathbb{V} \geq A$;
- Accept $\mathcal{H}_0$ and terminate if $\mathbb{V} \leq B$;
- Aggregate an additional report and add the corresponding detector index to $\underline{\Theta}$ if $A < \mathbb{V} < B$,

where A and B are two decision thresholds derived from the desired miss detection and false alarm probabilities. In particular, let χ and ψ denote the desired miss detection and false alarm probabilities, respectively. The decision thresholds are given in [111] as

$$A = \ln(\frac{1-\chi}{\psi}) \quad \text{and} \quad B = \ln(\frac{\chi}{1-\psi}) \ . \tag{6.7}$$

In each iteration, the SSP chooses a sensing report with the lowest rank from the remaining reports, updates $\mathbb{V}$ according to Eq. (6.6), and checks if a final decision can be reached. In addition, in case a decision cannot be reached after aggregating all the sensing reports, the SSP permissively accepts $\mathcal{H}_0$ to avoid potential interference with the primary user. In the end, the SSP updates the reputation profile for each mobile detector (see Section 6.3.4).

Our scheme obviously has greater resilience to false sensing reports. In particular, a sensing report from a less reputable mobile detector will be assigned a smaller weight and is thus less likely to drastically affect the final decision. In addition, a sensing report with low instantaneous trustworthiness will be counted only if a final decision

cannot be reached after combining all the other sensing reports with higher instantaneous trustworthiness (i.e., smaller t_i). As long as there are sufficient mobile detectors in the cell, a robust decision can still be reached even if there are much more malicious mobile detectors.

6.3.4 Fine-Grained Reputation Management

As discussed in Section 6.3.3, the reputation scores of mobile detectors are used to combine their sensing reports in PWSPRT. Now we present a novel reputation system for the SSP to record the past sensing performance of mobile detectors.

Our reputation system will be built upon our previous work [138] which is firmly rooted in the classical Bayesian inference theory used to estimate one or more unknown quantities from the results of a sequence of multinomial trials. For clarity, we outline the adopted Dirichlet-Multinomial model as follows and refer to [34] for more details. A multinomial trial process is a sequence of independent, identically distributed (i.i.d.) random variables U_1, U_2, ..., each taking one of ϖ possible outcomes $\{o_i\}_{i=1}^{\varpi}$. We then denote the common probability density function (PDF) of the trial variables by $p_i = \mathsf{P}(\mathsf{U}_j = o_i)$ for $1 \leq i \leq \varpi$, where $p_i > 0$ and $\sum_{i=1}^{\varpi} p_i = 1$. Let $\mathbf{p} = (p_1, ..., p_\varpi)$ and $\mathbf{z} = (z_1, ..., z_\varpi)$ which is the vector of observation counts of each outcome after N multinomial trials, namely, $\sum_{i=1}^{k} z_i = N$. The multinomial sampling distribution [34] states that

$$f(\mathbf{z}|\mathbf{p}) = \mathsf{Mult}(N|p_1, ..., p_\varpi) = \frac{N!}{\prod_{i=1}^{\varpi} z_i!} \prod_{i=1}^{\varpi} p_i^{z_i} \ .$$

It is commonly assumed in Bayesian inference that $\mathbf{p}$ has a conjugate prior distribution[1] known as the Dirichlet,

$$f(\mathbf{p}) = \mathsf{Dir}(\mathbf{p}|\alpha_1, ..., \alpha_\varpi) = \frac{\Gamma(\sum_{i=1}^{\varpi} \alpha_i)}{\prod_{i=1}^{\varpi} \Gamma(\alpha_i)} \prod_{i=1}^{\varpi} p_i^{\alpha_i - 1} \ ,$$

where $p_i \neq 0$ if $\alpha_i < 1$ and Γ is the gamma function[2]. The positive parameters α_i can be interpreted as "prior observation counts" for events governed by p_i. Then the

[1] The property that the posterior distribution follows the same parametric form as the prior distribution is called *conjugacy* [34].

[2] If x is an integer, $\Gamma(x) = (x-1)!$.

posterior distribution is also Dirichlet [34],

$$\begin{aligned} f(\mathbf{p}|\mathbf{z}) &= \frac{f(\mathbf{z}|\mathbf{p}) \times f(\mathbf{p})}{f(\mathbf{z})} \\ &= \frac{\Gamma(\sum_{i=1}^{\varpi}(\alpha_i + z_i))}{\prod_{i=1}^{\varpi}\Gamma(\alpha_i + z_i)} \prod_{i=1}^{\varpi} p_i^{\alpha_i + z_i - 1} \\ &= \mathsf{Dir}(\mathbf{p}|\alpha_1 + z_1, ..., \alpha_\varpi + z_\varpi), \end{aligned} \tag{6.8}$$

which can be used to make statements about $\mathbf{p}$ considered as a set of random quantities. The posterior mean of p_i, which maybe be interpreted as the posterior probability of observing outcome o_i in a future multinomial trial, is

$$\mathsf{E}[p_i|\mathbf{z}] = \frac{\alpha_i + z_i}{\sum_{i=1}^{\varpi}(\alpha_i + z_i)} \ . \tag{6.9}$$

In what follows, we detail how to apply the Dirichlet-Multinomial model in our scheme.

Let $\varpi = 2(q+1)$ for some integer $q \geq 1$. The SSP first divides the range $(-\infty, \infty)$ into $2q+2$ intervals, denoted by $I_1, \cdots, I_{2q+2}$. Recall that A and B ($B < 0 < A$) are the decision thresholds used in PWSPRT, which correspond to $\mathcal{H}_1$ and $\mathcal{H}_0$, respectively (cf. Eq. (6.7)). The jth interval is given by

$$I_j = \begin{cases} (-\infty, B] & \text{if } j = 1, \\ (\frac{(k^{q+2-j}-1)B}{k^q-1}, \frac{(k^{q+1-j}-1)B}{k^q-1}] & \text{if } 2 \leq j \leq q+1, \\ (\frac{(k^{j-q-2}-1)A}{k^q-1}, \frac{(k^{j-q-1}-1)A}{k^q-1}] & \text{if } q+2 \leq j \leq 2q+1, \\ (A, \infty) & \text{if } j = 2q+2 \ , \end{cases} \tag{6.10}$$

where $k > 1$ is a system parameter. Let $|I_j|$ denote the length of the jth interval. It follows that $|I_j| = k|I_{j+1}|$čň for all $2 \leq j \leq q$, and $|I_j| = k|I_{j-1}|$čň for all $q+2 \leq j \leq 2q$. The reason to have the length of intervals form two geometric sequences is that most normal detectors will have a relative small negative or positive contribution in low SNR environment. By choosing small length for the intervals in the middle, we can better differentiate the performance levels among different detectors.

After each sensing task, the SSP maps the performance of each mobile detector into one of the ϖ levels. In particular, for each $i \in \Theta$, let $\mathbf{c}_i = \ln \mathsf{P}(S_i|\mathcal{H}_1) -$

$\ln \mathrm{P}(S_i|\mathcal{H}_0)$ be the potential contribution of mobile detector i in PWSPRT, regardless of its weight $\mathbf{w}_i$. The SSP first maps $\mathbf{c}_i$ into one of the ϖ intervals, say I_{η_i}. The performance level of mobile detector i is then given by

$$l_i = \begin{cases} \eta_i & \text{if } \mathcal{H}_1 \text{ is accepted,} \\ \varpi + 1 - \eta_i & \text{if } \mathcal{H}_0 \text{ is accepted.} \end{cases} \tag{6.11}$$

In other words, if mobile detector $i \in \underline{\Theta}$ has a positive (or negative) contribution to the final decision, its sensing performance will be mapped into one of the higher (lower) $q+1$ levels.

The SSP maintains a reputation profile for every mobile detector $i \in \Theta \cup \Theta_a$, represented by ϖ counters $\{c_{i,s}\}_{s=1}^{\varpi}$. Each counter $c_{i,s}$ corresponds to the sth performance level and is initialized to c_0. After every sensing task, the SSP increases the corresponding counter of every mobile detector $\underline{\Theta}$ involved in PWSPRT according to his performance level.

The SSP then computes $w_{i,s} = \frac{c_{i,s}}{\sum_{s=1}^{\varpi} c_{i,s}}$ for all $s \in [1, \varpi]$, where $w_{i,s}$ refers to the expected probability that detector i will have level-s sensing performance (cf. Eq. (6.9)). If the SSP desires a performance level no less than $l \in [1, \varpi]$, it computes the reputation score for detector i as

$$w_i = \sum_{s=1}^{l} w_{i,s} \ .$$

Let $w_{\max}$ be maximum reputation score among all mobile detectors $\Theta \cup \Theta_a$. The normalized reputation score of detector i is then given by

$$\mathrm{w}_i = w_i / w_{\max} \ , \tag{6.12}$$

which will be used as the weight of detector i in PWSPRT.

The choice of l is important. In particular, the higher l is, the lower the weight w_i for each $i \in \Theta \cup \Theta_a$, the fewer mobile detectors with a non-zero weight, and vice versa. We will study the tradeoff between sensing quality and resilience to malicious mobile detectors using simulations in Section 6.4.

In addition, past performance may not always be relevant for determining the current performance of mobile detectors who may update their devices and/or vary their behaviors over time. To deal with this situation, the SSP could choose a *discount factor* ν between $[0, 1.0]$ to assign more weight to recency. At regular intervals, the SSP updates $\{c_{i,s}\}_{s=1}^{\varpi} := \{\nu c_{i,s}\}_{s=1}^{\varpi}$. Discounting the past not only helps identify mobile detectors who behave well initially and poorly afterwards, but also permits a disreputable mobile detector to reform by starting to have good performance.

6.4 Performance Evaluation

In this section, we evaluate the proposed scheme using extensive simulation. As the only piece of prior work that targets malicious detectors being the majority, the solution in [27] relies on real signal propagation data which may be very difficult to obtain especially in urban environments. It is thus less meaningful to directly compare our work with [27] because our scheme does not require real signal propagation data. Instead, we compare our work with the techniques proposed in [16] (denoted by CPB) and [53] (denoted by KKB) that are under similar signal-propagation models and assumptions as well as the standard SPRT [141] that corresponds to no defense in place.

6.4.1 Simulation Setup

As in [73], we consider an IEEE 802.22 WRAN environment with a single DTV transmitter with 6MHz bandwidth and 150.3 km transmission range [100]. We simulate a rectangle cell of $5 \times 5\ \mathrm{km}^2$. The distance between the center of the cell to the primary user is 145 km. We set the minimum distance between any two detectors to be $200\ \mathrm{m}$ to decorrelate their shadow fading X_i [5]. In addition, we call a malicious mobile detector i has an attack strength T (dB) if it reports a $s_i + T$ where s_i is the true average of the RSSI values [73]. We assume that there are 100 mobile detectors in the cell, among which M are malicious.

Table I lists the default parameters used in our simulation unless stated otherwise. The simulation is done in Matlab, and each point is the average of 10000 runs, each with a random seed. In addition, since both CPB [16] and our scheme use reputa-

Table 6.1: Default simulation setting

Para.	Value	Description
$\|\Theta\|$	100	Number of mobile detectors
M	50	Number of malicious mobile detectors
$\|\Theta_a\|$	5	Number of anchor detectors
d_0	1m	Reference distance
P_0	90 dBm	The received power at d_0
m	6×10^3 [73]	Number of samples
α	3.7	Path loss exponent
χ	0.01	Desired miss detection probability
ψ	0.1	Desired false alarm probability
σ	5.5 dB [100]	The standard deviation of shadow fading X_i
α_s	1	Initial value for each counter in reputation profile
ϖ	22	Total number of service levels
k	1.4	Ratio between adjacent performance intervals
l	12	Minimum desired service level

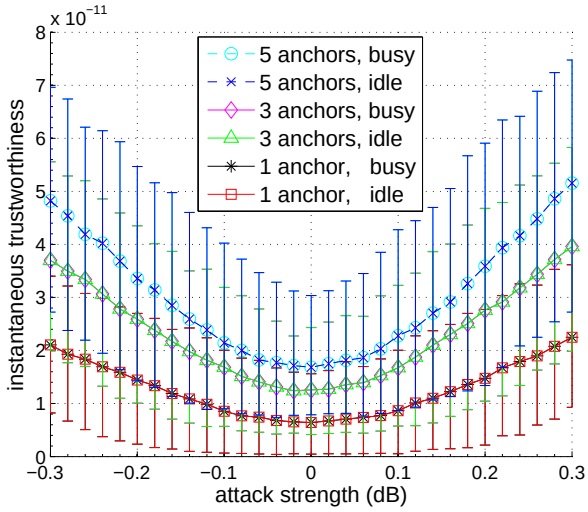


Figure 6.2: Instantanoes trustworthiness vs. attack strength.

tion scores, meaning that the outcomes of later rounds are partially determined by those of previous rounds, we divide the total 10000 rounds into 100 groups, each containing 100 rounds, and the reputation score of each mobile detector is reset at the beginning of each group.

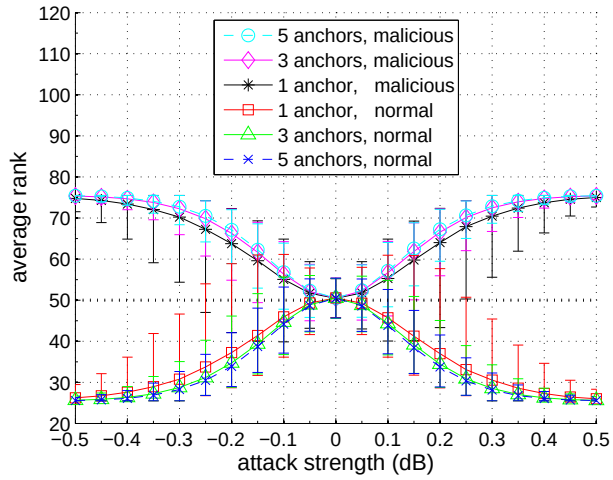


Figure 6.3: The average rank of malicious mobile detectors vs. attack strength.

6.4.2 Simulation Results

6.4.2.1 Instantaneous Trustworthiness

We first report the simulation result for the proposed instantaneous trustworthiness measure, which is one of the key components of our scheme.

Fig. 6.2 shows the instantaneous trustworthiness of a malicious mobile detector with its attack strength T varying between -0.3 and 0.3 dB. We can see that the instantaneous trustworthiness increases as the attack strength $|T|$ increases. The reason is that the higher the attack strength, the larger deviation from the expected received power at its location, the larger $|z_{i,j}|$ with regard to each anchor node j, the larger t_i, and vice versa. In addition, instantaneous trustworthiness also increases as the number of anchor detectors increases, because each additional anchor detector corresponds to one relative instantaneous trustworthiness value that will be counted in the overall instantaneous trustworthiness. As a result, the more anchor detectors, the more sensitive the instantaneous trustworthiness measure to false sensing data attack.

Fig. 6.3 shows the average ranks of 50 malicious mobile detectors and 50 normal detectors varying with attack strength. We can see that the average rank of a

malicious detector increases rapidly from 50 and converges to 75 as the attack strength increases, which means that the 50 malicious detectors constantly rank between 51 and 100, leading to an average rank of 75. In contrast, the average rank of 50 normal detectors decreases and converges to 25, meaning that the 50 normal detectors consistently rank between 1 and 50. These results are expected since malicious (or normal) detectors will have low (or high) instantaneous trustworthiness with high probability. In addition, we can see that the more anchor detectors, the smaller variance of the average rank for both malicious and normal mobile detectors. This means that by aggregating the relative instantaneous trustworthiness with regard to multiple anchor nodes, the rank based on instantaneous trustworthiness can effectively differentiate malicious detectors from normal detectors.

6.4.2.2 Reputation System

Fig. 6.4 shows the average normalized reputation scores of 50 normal detectors and 50 malicious detectors in each round with a different desired service level l. We can see that the average reputation score of normal mobile detectors increases slowly after each round, while that of malicious mobile detectors remains stable. The reason is that in each round, most detectors involved in PWSPRT will be normal detectors with high probability, so only the reputation counters of a subset of normal detectors will be updated, leading to a slow increase in their average reputation scores. For the same reason, the reputation counters of malicious detectors will not be updated, whose average reputation score will remain stable. Such updates can still guarantee good detectors having higher reputation scores than malicious ones. In addition, we can see that the higher l is, the smaller the initial reputation score for each detector, and the larger difference between the average reputation scores of normal detectors and malicious ones after sufficient rounds. This represents the tradeoff between the desired service level and convergence time as well as the final difference in the reputation scores of normal detectors and malicious ones.

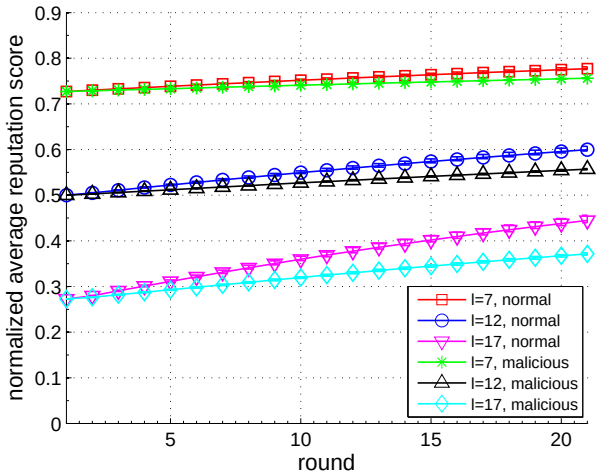


Figure 6.4: The progressive average reputation scores of normal and malicious mobile detectors.

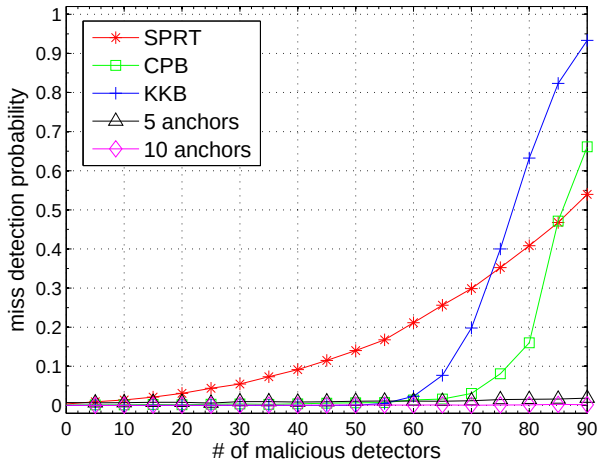


Figure 6.5: Miss detection probability vs. # of malicious detectors.

6.4.2.3 Resilience to Malicious Mobile Detectors

Fig. 6.5 shows the miss detection probabilities of our scheme, CPB, KKB, and SPRT varying with the number of malicious detectors, where the attack strength of malicious detectors is -0.1 dB. We can see that the miss detection probability of SPRT increases as the number of malicious detectors increases, which is expected. In addition, the miss

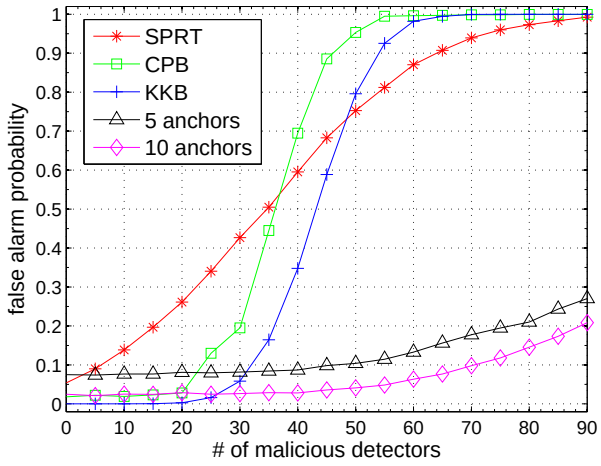


Figure 6.6: False alarm probability vs. # of malicious detectors

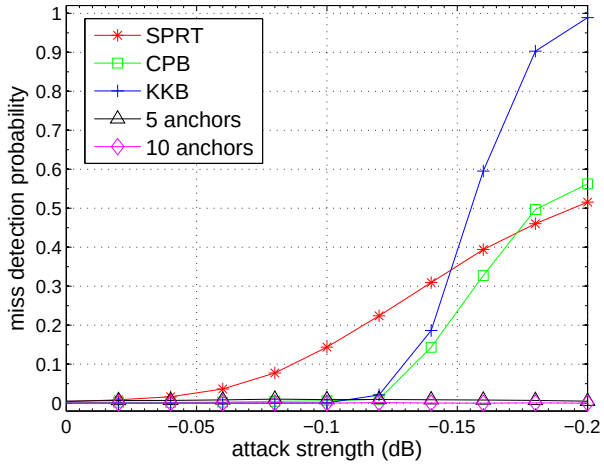


Figure 6.7: Miss detection probability vs. attack strength.

detection probabilities of CPB and KKB are close to zero when the number of malicious detectors is below 60, meaning both of them are resilient to false sensing data attack when the malicious detectors do not constitute the majority. As the number of malicious detectors further increases, the miss detection probabilities of CPB increases and eventually exceeds that of SPRT. The reason is that once the malicious detectors dominate the cell, they will always determine the final decision and have their reputation scores increased, while the normal detectors will always make the “wrong" decision and have

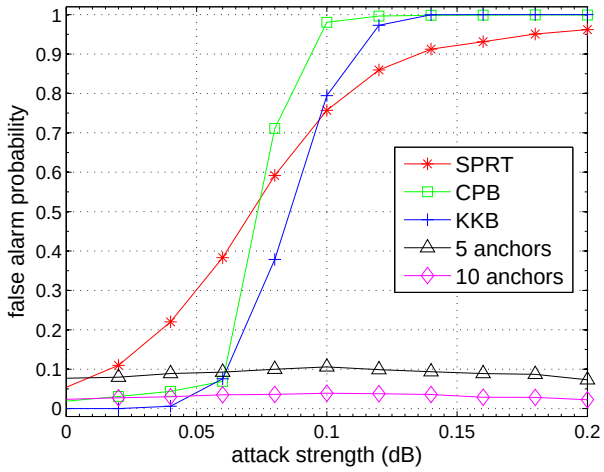


Figure 6.8: False alarm probability vs. attack strength.

their reputation scores decreased. Similar trend can be observed about KKB, because normal detectors' sensing reports will be considered as outliers and filtered out once the malicious detectors constitute the majority. In contrast, the miss detection probability of our scheme is insensitive to the increase in the number of malicious detectors and remains below 0.05 even when the number of malicious detectors exceeds 90. The reason is that malicious detectors will be ranked after normal detectors with high probability using our instantaneous trustworthiness measure, so the SSP can make a correct decision as long as there are sufficient normal detectors.

Fig. 6.6 shows the false alarm probabilities of our scheme with CPB, KKB, and SPRT varying with the number of malicious detectors, where the attack strength of malicious detectors is 0.1 dB. Similar to Fig. 6.5, the false alarm probability of SPRT increases as the number of malicious detectors increases, which is of no surprise. The false alarm probabilities of CPB and KKB are both close to zero when the number of malicious detectors is smaller than 20 and increase as the number of malicious detectors further increases, which further demonstrate that they are effective against small fraction of malicious detectors but ineffective when the malicious detectors become the majority. In contrast, the false alarm probability of our scheme is much less sensitive

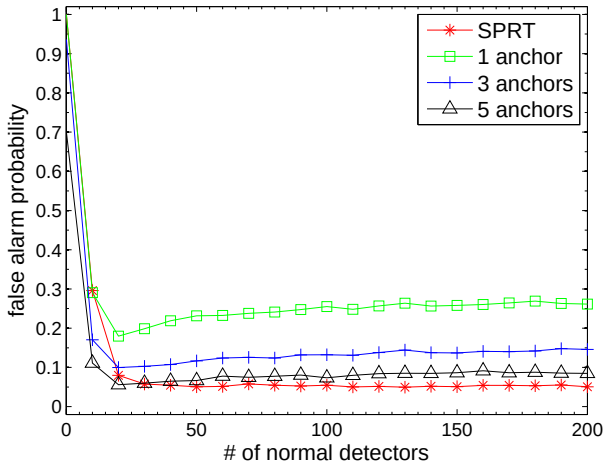


Figure 6.9: False alarm probability vs. # of mobile detectors under no attack.

to the increase in the number of malicious detectors. In addition, the more anchor detectors, the lower the false alarm probability, and vice versa.

Fig. 6.7 compares the miss detection probabilities of our scheme with CPB, KKB, and SPRT with the attack strength varying between 0 to -0.2 dB, where the number of malicious detectors is 50. We can see that the miss detection probabilities of CPB, KKB, and SPRT all increase as the attack strength increases. The reason is that neither CPB nor KKB can withstand the malicious mobile detectors being the majority, not to mention SPRT. In contrast, the miss detection probability of our scheme is relatively insensitive to the increase in attack strength as our instantaneous trustworthiness measure can effectively differentiate malicious detectors from the normal ones. As long as there are enough normal detectors, the SSP can make a correct decision under our scheme.

Fig. 6.8 compares the false alarm probabilities of our scheme with CPB, KKB, and SPRT. The result is very similar to that in Fig. 6.7 and is omitted here.

6.4.2.4 Performance under No Attack

Fig. 6.9 compares the false alarm probabilities of our scheme and SPRT varying with the number of mobile detectors where there is no malicious detectors. We can see that the

false alarm probability of SPRT decreases from 1 to below 0.1 as the number of mobile detectors increases from 0 to 20 and remains stable as the number of mobile detectors further increases, which is expected for SPRT. In contrast, the false alarm probability of our scheme first decreases as the number of mobile detectors increases from 0 to 20 and then slightly increases as the number of mobile detectors further increases. The reason is that our scheme relies on the instantaneous trustworthiness measure to order all the sensing reports, which further relies on a few anchor detectors. The order for aggregating sensing reports in our scheme is thus different from the random order used in SPRT. Since an anchor detector may also report an inaccurate sensing report due to temporal channel impairment, a mobile detector with similar inaccurate sensing report will obtain high instantaneous trustworthiness (i.e., low t_i) if there are only a few anchor detectors, leading to the increase in the false alarm probability. In addition, we can see that using multiple anchor detectors can largely mitigate this limitation, because it is very unlikely for an inaccurate sensing report to simultaneously attain high relative instantaneous trustworthiness with regard to all the anchor detectors. We have also simulated the miss detection probability of our scheme under no attack. The result is very similar to that of Fig. 6.9 and is thus omitted here.

6.4.3 Discussion

We summarize the simulation results as follows.

- Our instantaneous trustworthiness metric can effectively differentiate normal detectors from malicious ones with the help of a small number of anchor detectors.
- Our scheme can enable robust PU detection even when the malicious detectors constitute the majority as long as there are sufficient number of normal detectors.
- When there are too few anchor detectors and too many mobile detectors, our scheme has a slightly worse performance than SPRT in case there is no attack. It is thus necessary to have multiple anchor detectors, say five, to achieve robust PU detection in practice.

The simulation results clearly demonstrate the significant advantage of our scheme over existing schemes under the similar models and assumptions.

6.5 Related Work

In this section, we briefly discuss some work in several areas which is most germane to our work in this chapter.

As mentioned in Section 6.1, previous works can be generally classified into three categories. The schemes proposed in [16, 26, 28, 53, 64, 73] use various anomaly detection techniques to identify false sensing reports, which would fail if false sensing reports constitute the majority. The second category such as [16, 53, 130] relies on reputation system to differentiate malicious CR users from legitimate users based on their past behaviors, but is unable to handle sudden change in mobile detectors' behaviors. The only piece of work that targets majority of false sensing reports appears in [27], which assumes that neighboring cells can help overturn the decision by the real signal propagation data from primary users. In contrast, our scheme does not rely on inter-cell crosscheck nor requires real signal propagation data from primary users. In addition, detecting false sensing reports in a distributed sensing architecture has been studied in [116, 125], which are orthogonal to our work in this chapter.

Another line of work is to mitigate the Primary User Emulation attack, i.e., testing whether the legitimate primary user is using a licensed channel or whether an attacker is impersonating the primary user to use the channel. Proposed solutions include primary user location estimation [15], authenticating primary user's signal via physically collocated helper node [67] or properly manipulating channel coding and modulation at the physical layer.

In addition, detecting possible spectrum misuse by arbitrary secondary users has been studied in [66, 117]. ALDO [66] uses statistical significance testing to detect illegitimate secondary users based on RSS measurements and the characteristics of radio propagation. The authors of [117] propose to let every legitimate channel user embed a cryptographic spectrum permit into its physical-layer cyclostationary features,

which can be verified by mobile “police devices" dispatched by the spectrum owner. These works target different type of attack and are thus orthogonal to our work in this chapter.

6.6 Summary

In this chapter, we have presented a novel secure crowdsourcing-based cooperative spectrum sensing scheme. The key idea behind our scheme is to jointly consider the instantaneous trustworthiness of mobile detectors in combination with their reputation scores during data fusion. Our scheme can enable robust cooperative sensing even if the malicious CR users are the majority. Extensive simulation results have demonstrated the effectiveness of our proposed scheme.

Chapter 7

CONCLUSION AND FUTURE WORK

In this dissertation, we offer novel solutions to five challenging security and privacy issues in heterogeneous mobile wireless networks, spanning wireless sensor networks, mobile crowdsourcing, and mobile social networking. In particular, we design a secure and loss-resilient code dissemination scheme for wireless sensor networks deployed in hostile and harsh environments. Moreover, we devise a novel scheme to enable mobile users to detect any inauthentic or unsound location-based top-k query query result returned by an untrusted location-based service providers. In addition, we develop a novel verifiable privacy-preserving aggregation scheme for people-centric mobile sensing systems. Furthermore, we present a suite of privacy-preserving profile matching protocols for proximity-based mobile social networking, which can support a wide range of matching metrics with different privacy levels. Finally, we introduce a secure combination scheme for crowdsourcing-based cooperative spectrum sensing systems.

7.1 Future Work

As our future work, we plan to further evaluate the performance of our solutions via prototype implementations on real network testbeds or platforms such as smartphones and Universal Software Radio Peripheral (USRP). We also plan to extend our solutions by exploring the following directions.

First, we intend to extend LR-Seluge to defend against denial-of-receipt attack, in which a compromised sensor node keeps requesting new pages from a target node. In addition, besides the location-based top-k queries, we also intend to extend our solution introduced in Chapter 3 to support other types of location-based queries, e.g., location-based range queries. Moreover, we intend to improve VPA to make it resilient to denial-of-service attack, in which a malicious user keeps submitting false data to disrupt the aggregation process. Furthermore, we plan to explore novel private matching mechanisms that do not rely on public key cryptosystem. Last but not the least, we plan to extend our secure combination scheme to support hard combination.

REFERENCES

[1] Cisco visual networking index global mobile data traffic forecast update 2011-2016.

[2] TinyOS: An open-source operating system designed for wireless embedded sensor networks. http://www.tinyos.net/, 2007.

[3] Tarek Abdelzaher, Yaw Anokwa, Peter Boda, Jeff Burke, Deborah Estrin, Leonidas Guibas, Aman Kansal, Samuel Madden, and Jim Reich. Mobiscopes for human spaces. *IEEE Pervasive Computing*, 6(2):20–29, 2007.

[4] I. Akyildiz, B. Lo, and R. Balakrishnan. Cooperative spectrum sensing in cognitive radio networks: A survey. *Physical Communication*, 4(1):40–62, Mar. 2011.

[5] A. Algans, K. Pedersen, and P. Mogensen. Experimental analysis of the joint statistical properties of azimuth spread, delay spread, and shadow fading. *IEEE Journal on Selected Areas in Communications*, 20(3):523–531, Apr. 2002.

[6] Marco Arb, Matthias Bader, Michael Kuhn, and Roger Wattenhofer. VENETA: Serverless friend-of-friend detection in mobile social networking. In *WIMOB'08*, pages 184–189, Avignon, France, Oct. 2008.

[7] Giuseppe Ateniese, Amir Herzberg, Hugo Krawczyk, and Gene Tsudik. Untraceable mobility or how to travel incognito. *Computer Networks*, 31(8):871–884, Apr. 1999.

[8] A. T. Campbell, S. B. Eisenman, N. D. Lane, E. Miluzzo, and R. A. Peterson. People-centric urban sensing. In *WICON'06*, Boston, Massachusetts, Aug. 2006.

[9] Ning Cao, Cong Wang, Ming Li, Kui Ren, and Wenjing Lou. Privacy-preserving multi-keyword ranked search over encrypted cloud data. In *IEEE INFOCOM*, Shanghai, China, Apr. 2011.

[10] Ning Cao, Zhenyu Yang, Cong Wang, Kui Ren, and Wenjing Lou. Secure ranked keyword search over encrypted cloud data. In *IEEE ICDCS'11*, Minnesapolis, MN, June 2011.

[11] Claude Castelluccia, Einar Mykletun, and Gene Tsudik. Efficient aggregation of encrypted data in wireless sensor networks. In *MobiQuitous'05*, pages 109–117, San Diego, CA, July 2005.

[12] Haowen Chan, Adrian Perrig, and Dawn Song. Secure hierarchical in-network aggregation in sensor networks. In *ACM CCS'06*, pages 278–287, Alexandria, Virginia, USA, Oct. 2006.

[13] Binbin Chen and Haifeng Yu. Secure aggregation with malicious node revocation in sensor networks. In *ICDCS'11*, pages 581–592, June 2011.

[14] Fei Chen and Alex Liu. SafeQ: Secure and efficient query processing in sensor networks. In *INFOCOM'10*, pages 1–9, San Diego, CA, Mar. 2010.

[15] R. Chen, J. Park, and J. Reed. Defense against primary user emulation attacks in cognitive radio networks. *IEEE Journal on Selected Areas in Communcations Special Issue on Cognitive Radio Theory and Applications*, 26(1):25–37, Jan. 2008.

[16] Ruiliang Chen, J. Park, and Kaigui Bian. Robust distributed spectrum sensing in cognitive radio networks. In *INFOCOM'08*, pages 1876–1884, April 2008.

[17] C. Cornelius, A. Kapadia, D. Kotz, D. Peebles, M. Shin, and N. Triandopoulos. Anonysense: privacy-aware people-centric sensing. In *ACM MobiSys'08*, pages 211–224, Breckenridge, CO, June 2008.

[18] Emiliano Cristofaro and Claudio Soriente. PEPSI: privacy enhancing participatory sensing infrastructure. In *WiSec'11*, Hamburg, Germany, June 2011.

[19] Emiliano Cristofaro and Gene Tsudik. Practical private set intersection protocols with linear complexity. In *FC'10*, volume 6052, pages 143–159, Tenerife, Canary Islands, Spain, Jan. 2010.

[20] Jing Deng, Richard Han, and Shivakant Mishra. Secure code distribution in dynamically programmable wireless sensor networks. In *IPSN'06*, pages 292–300, Nashville, Tennessee, Apr. 2006.

[21] Wei Dong, Vacha Dave, Lili Qiu, and Yin Zhang. Secure friend discovery in mobile social networks. In *INFOCOM'11*, Shanghai, China, Apr. 2011.

[22] Wenliang Du and Mikhail Atallah. Protocols for secure remote database access with approximate matching. In *WSPEC'00*, Athens, Greece, Nov. 2000.

[23] Wenliang Du and Mikhail Atallah. Privacy-preserving cooperative statistical analysis. In *ACSAC'01*, pages 102–110, New Orleans, Louisiana, Dec. 2001.

[24] Akshay Dua, Nirupama Bulusu, Wu chang Feng, and Wen Hu. Towards trustworthy parcitipatory sensing. In *USENIX HotSec'09*, Montreal, Canada, Aug. 2009.

[25] Prabal Dutta, Jonathan Hui, David Chu, and David Culler. Securing the deluge network programming system. In *IPSN'06*, pages 326–333, Apr. 2006.

[26] O. Fatemieh, R. Chandra, and C. Gunter. Secure collaborative sensing for crowd-sourcing spectrum data in white space networks. In *DySPAN'10*, Singapore, Apr. 2010.

[27] O. Fatemieh, M. LeMay, and C. Gunter. Reliable telemetry in white spaces using remote attestation. In *ACSAC'11*, pages 323–332, Orlando, FL, 2011.

[28] Omid Fatemieh, Ali Farhadi, Ranveer Chandra, and Carl Gunter. Using classification to protect the integrity of spectrum measurements in white space networks. In *NDSS'11*, San Diego, CA, Feb. 2011.

[29] Taiming Feng, Chuang Wang, Wensheng Zhang, and Lu Ruan. Confidentiality protection for distributed sensor data aggregation. In *IEEE INFOCOM'08*, pages 475–483, Phoneix, AZ, Apr. 2008.

[30] Michael Freedman, Kobbi Nissim, and Benny Pinkas. Efficient private matching and set intersection. In *EUROCRYPT'04*, pages 1–19, Interlaken, Switzerland, May 2004.

[31] Jon Froehlich, Tawanna Dillahunt, Predrag Klasnja, Jennifer Mankoff, Sunny Consolvo, Beverly Harrison, and James Landay. UbiGreen: investigating a mobile tool for tracking and supporting green transportation habits. In *CHI'09*, pages 1043–1052, Boston, MA, 2009.

[32] Xinwen Fu, Nan Zhang, Aniket Pingley, Wei Yu, Jie Wang, and Wei Zhao. The digital Marauder's map: A new threat to location privacy. In *ICDCS'09*, pages 589–596, Montreal, Quebec, Canada, June 2009.

[33] Raghu K. Ganti, Nam Pham, Yu-En Tsai, and Tarek F. Abdelzaher. Poolview: stream privacy for grassroots participatory sensing. In *ACM SenSys'08*, pages 281–294, Raleigh, NC, Nov. 2008.

[34] Andrew Gelman, John B. Carlin, Hal S. Stern, and Donald B. Rubin. *Bayesian Data Analysis*. Chapman & Hall/CRC, 1st edition, June 1995.

[35] Peter Gilbert, Landon Cox, Jaeyeon Jung, and David Wetherall. Toward trustworthy mobile sensing. In *ACM HotMobile'10*, pages 31–36, Annapolis, MA, Feb. 2010.

[36] Peter Gilbert, Jaeyeon Jung, Kyungmin Lee, Henry Qin, Daniel Sharkey, Anmol Sheth, and Landon Cox. YouProve: Authenticity and fidelity in mobile sensing. In *SenSys'11*, Seattle, WA, Nov. 2011.

[37] Joao Girao, Markus Schneider, and Dirk Westhoff. CDA: Concealed data aggregation in wireless sensor networks. In *ACM WiSe'04*, Philadelphia, PA, Oct. 2004.

[38] Bart Goethals, Sven Laur, Helger Lipmaa, and Taneli Mielikainen. On private scalar product computation for privacy-preserving data mining. In *ICISC'04*, volume 3506, pages 23–25, Seoul, Korea, Dec. 2004.

[39] Oded Goldreich, Siltnb Micali, and Avi Wigderson. How to play ANY mental game. In *STOC'87*, pages 218–229, New York, NY, 1987.

[40] Shyam Gollakota, Nabeel Ahmed, Nickolai Zeldovich, and Dina Katabi. Secure in-band wireless pairing. In *USENIX Security*, San Francisco, CA, Aug. 2011.

[41] Hakan Hacigümüş, Bala Iyer, Chen Li, and Sharad Mehrotra. Executing SQL over encrypted data in the database-service-provider model. In *ACM SIGMOD'02*, pages 216–227, Madison, Wisconsin, 6 2002.

[42] Hakan Hacigümüs, Sharad Mehrotra, and B. Iyer. Providing database as a service. In *IEEE ICDE*, Aalborg, Denmark, Feb. 2002.

[43] Andrew Hagedorn, David Starobinski, and Ari Trachtenberg. Rateless deluge: Over-the-air programming of wireless sensor networks using random linear codes. In *IPSN'08*, pages 457–466, Washington, DC, Apr. 2008.

[44] Wenbo He, Xue Liu, Hoang Nguyen, Klara Nahrstedt, and Tarek F. Abdelzaher. PDA: Privacy-preserving data aggregation in wireless sensor networks. In *IEEE INFOCOM'07*, pages 2045–2053, Anchorage, Alaska, USA, May 2007.

[45] Bijit Hore, Sharad Mehrotra, and Gene Tsudik. A privacy-preserving index for range queries. In *VLDB'04*, pages 720–731, Toronto, Canada, Aug. 2004.

[46] I-Hong Hou, Yu-En Tsai, Tarek Abdelzaher, and Indranil Gupta. Adapcode: Adaptive network coding for code updates in wireless sensor networks. In *INFOCOM'08*, pages 1517–1525, Phoenix, AZ, Apr. 2008.

[47] Jonathan Hui and David Culler. The dynamic behavior of a data dissemination protocol for network programming at scale. In *SenSys'04*, pages 81–94, Baltimore, MD, Nov. 2004.

[48] Bret Hull, Vladimir Bychkovsky, Yang Zhang, Kevin Chen, Michel Goraczko, Allen Miu, Eugene Shih, Hari Balakrishnan, and Samuel Madden. CarTel: A distributed mobile sensor computing system. In *ACM SENSYS'06*, pages 125–138, Boulder, CO, Oct. 2006.

[49] Sangwon Hyun, Peng Ning, An Liu, and Wenliang Du. Seluge: Secure and dos-resistant code dissemination in wireless sensor networks. In *IPSN'08*, pages 445–456, April 2008.

[50] Ioannis Ioannidis, Ananth Grama, and Mikhail Atallah. A secure protocol for computing dot-products in clustered and distributed environments. In *ICPP'02*, pages 379–384, Vancouver, British Columbia, Canada, Aug. 2002.

[51] Java implementation of Paillier cryptosystem. available at http://www.csee.umbc.edu/∼kunliu1/research/Paillier.html.

[52] Ari Juels and John Brainard. Client puzzles: A cryptographic countermeasure against connection depletion attacks. In *NDSS'99*, pages 151–165, San Diego, CA, Feb. 1999.

[53] P. Kaligineedi, M. Khabbazian, and V.K. Bhargava. Secure cooperative sensing techniques for cognitive radio systems. In *ICC'08*, pages 3406–3410, may 2008.

[54] T. Kamakaris, M.M. Buddhikot, and R. Iyer. A case for coordinated dynamic spectrum access in cellular networks. In *DySPAN'05*, pages 289–298, Nov. 2005.

[55] Aman Kansal, Suman Nath, Jie Liu, and Feng Zhao. SenseWeb: An infrastructure for shared sensing. *IEEE MultiMedia*, 14(4):8–13, 2007.

[56] Apu Kapadia, David Kotz, and Nikos Triandopoulos. Opportunistic sensing: Security challenges for the new paradigm. In *COMSNETS'09*, Bangalore, India, Jan 2009.

[57] Aggelos Kiayias, Bülent Yener, and Moti Yung. Privacy-preserving information markets for computing statistical data. In *FC'09*, pages 32–50, Berlin, Heidelberg, Feb. 2009.

[58] Lea Kissner and Dawn Song. Privacy-preserving set operations. In *CRYPTO'05*, pages 241–257, Santa Barbara, CA, Aug. 2005.

[59] Wei-Shinn Ku, Ling Hu, Cyrus Shahabi, and Haixun Wang. Query integrity assurance of location-based services accessing outsourced spatial databases. In *Int. Sym. on Advances in Spatial and Temporal Databases*, Aalborg, Denmark, July 2009.

[60] Patrick Lanigan, Rajeev Gandhi, and Priya Narasimhan. Sluice: Secure dissemination of code updates in sensor networks. In *ICDCS'06*, pages 53–62, Washington, DC, July 2006.

[61] Peter Leijdekkers and Valerie Gay. Personal heart monitoring and rehabilitation system using smart phones. In *ICMB'06*, pages 29–36, Washington, DC, USA, 2006.

[62] Philip Levis, Nelson Lee, Matt Welsh, and David Culler. TOSSIM: accurate and scalable simulation of entire tinyos applications. In *SenSys'03*, pages 126–137, Los Angeles, CA, Nov. 2003.

[63] Philip Levis, Neil Patel, David Culler, and Scott Shenker. Trickle: a self-regulating algorithm for code propagation and maintenance in wireless sensor networks. In *NSDI'04*, pages 15–28, San Francisco, CA, Mar. 2004.

[64] Husheng Li and Zhu Han. Catch me if you can: An abnormality detection approach for collaborative spectrum sensing in cognitive radio networks. *IEEE Transactions on Wireless Communications*, 9(11):3554–3565, Nov. 2010.

[65] Ming Li, Ning Cao, Shucheng Yu, and Wenjing Lou. FindU: Privacy-preserving personal profile matching in mobile social networks. In *INFOCOM'11*, Shanghai, China, Apr. 2011.

[66] Song Liu, Larry Greenstein, Yingying Chen, and Wade Trappe. ALDO: An anomaly detection framework for dynamic spectrum access networks. In *IEEE INFOCOM'09*, Rio de Janeiro, Brazil, Apr. 2009.

[67] Yao Liu, Peng Ning, and Huaiyu Dai. Authenticating primary users' signals in cognitive radio networks via integrated cryptographic and wireless link signatures. In *S&P'10*, pages 286–301, Washington, DC, USA, 2010.

[68] Hong Lu, Wei Pan, Nicholas D. Lane, Tanzeem Choudhury, and Andrew T. Campbell. Soundsense: Scalable sound sensing for people-centric applications on mobile phones. In *ACM MobiSys'09*, pages 165–178, Wroclaw, Poland, June 2009.

[69] Rongxing Lu, Xiaodong Lin, Xiaohui Liang, and Xuemin Shen. A secure handshake scheme with symptoms-matching for mhealthcare social network. *Mobile Networks and Applications*, pages 1–12, 2010.

[70] Suhas Mathur, Tong Jin, Nikhil Kasturirangan, Janani Chandrasekaran, Wenzhi Xue, Marco Gruteser, and Wade Trappe. ParkNet: drive-by sensing of road-side parking statistics. In *MobiSys'10*, pages 123–136, San Francisco, CA, 2010.

[71] R. Merkle. Protocols for public key cryptosystems. In *IEEE S&P'80*, pages 122–134, Oakland, CA,USA, Apr. 1980.

[72] Ralph Merkle. A certified digital signature. In *CRYPTO*, pages 218–238, Santa Barbara, CA, Aug. 1989.

[73] A. Min, K. Shin, and Xin Hu. Attack-tolerant distributed sensing for dynamic spectrum access networks. In *ICNP'09*, pages 294–303, Princeton, NJ, oct. 2009.

[74] A. Min, Xinyu Zhang, and K. Shin. Detection of small-scale primary users in cognitive radio networks. *IEEE Journal on Selected Areas in Communications*, 29(2):349 –361, Feb. 2011.

[75] Michael Mitzenmacher. Digital fountains: a survey and look forward. In *IEEE ITW'04*, pages 271 – 276, San Antonio, Texas, Oct. 2004.

[76] U. Moller, L. Cottrell, P. Palfrader, and L. Sassaman. Mixmaster protocol version 2. IETF Internet Draft, July 2003.

[77] Rohan Narayana Murty, Geoffrey Mainland, Ian Rose, Atanu Roy Chowdhury, Abhimanyu Gosaint, Josh Berst, and Matt Welsh. CitySense: An urban-scale wireless sensor network and testbed. In *IEEE HST'08*, pages 583–588, Waltham, MA, May 2008.

[78] H. Mutlu, M. Alanyali, and D. Starobinski. Spot pricing of secondary spectrum usage in wireless cellular networks. In *INFOCOM'08*, pages 682 –690, April 2008.

[79] Maithili Narasimha and Gene Tsudik. Authentication of outsourced databases using signature aggregation and chaining. In *DASFAA'06*, pages 420–436, Singapore, Apr. 2006.

[80] Dong Nguyen, Tuan Tran, Thinh Nguyen, and Bella Bose. Wireless broadcast using network coding. *IEEE Transactions on Vehicular Technology,*, 58(2):914–925, Feb. 2009.

[81] Peng Ning, An Liu, and Wenliang Du. Mitigating dos attacks against broadcast authentication in wireless sensor networks. *ACM Transactions on Sensor Networks (TOSN)*, 4(1):1–31, Jan. 2008.

[82] Pascal Paillier. Public-key cryptosystems based on composite degree residuosity classes. In *EUROCRYPT'99*, pages 223–238, Prague, Czech Republic, May 1999.

[83] Hweehwa Pang and Kian-Lee Tan. Verifying completeness of relational query answers from online servers. *ACM Trans. Inf. Syst. Secur.*, 11(2):1–50, Mar. 2008.

[84] HweeHwa Pang, Jilian Zhang, and Kyriakos Mouratidis. Scalable verification for outsourced dynamic databases. *PVLDB*, 2(1):802–813, 2009.

[85] Andrew Parker, Sasank Reddy, Thomas Schmid, Kevin Chang, Ganeriwal Saurabh, Mani Srivastava, Mark Hansen, Jeff Burke, Deborah Estrin, Mark Allman, and Vern Paxson. Network system challenges in selective sharing and verification for personal, social, and urban-scale sensing applications. In *HotNets-V'06*, pages 37–42, Irvine, CA, Nov. 2006.

[86] E. Paulos and T. Jenkins. Urban Probes: encountering our emerging urban atmospheres. In *ACM CHI'05*, pages 341–350, Portland, OR, April 2005.

[87] C. Perkins, E. Belding-Royer, and S. Das. Ad hoc on-demand distance vector (AODV) routing. RFC 3561, July 2003.

[88] R. Pickholtz, D. Schilling, and L. Milstein. Theory of spread-spectrum communications–a tutorial. *IEEE Transactions on Communications*, 30(5):855–884, May 1982.

[89] Huseyin Polat and Wenliang Du. Privacy-preserving collaborative filtering using randomized perturbation techniques. In *ICDM'03*, pages 625–639, Melbourne, FL, Nov. 2003.

[90] Bartosz Przydatek, Dawn Song, and Adrian Perrig. SIA: Secure information aggregation in sensor networks. In *ACM SenSys'03*, pages 255–265, Los Angeles, CA, Nov. 2003.

[91] Shantanu Rane, Wei Sun, and Anthony Vetro. Privacy-preserving approximation of L_1 distance for multimedia applications. In *ICME'10*, pages 492–497, Singapore, July 2010.

[92] Michele Rossi, Giovanni Zanca, Luca Stabellini, Riccardo Crepaldi, Albert Harris III, and Michele Zorzi. SYNAPSE: A network reprogramming protocol for wireless sensor networks using fountain codes. In *IEEE SECON'08*, pages 188–196, San Francisco, CA, June 2008.

[93] Sankardas Roy, Mauro Conti, Sanjeev Setia, and Sushil Jajodia. Securely computing an approximate median in wireless sensor networks. In *SecureComm'08*, pages 6:1–6:10, Istanbul, Turkey, 2008.

[94] Sankardas Roy, Sanjeev Setia, and Sushil Jajodia. Attack-resilient hierarchical data aggregation in sensor networks. In *SASN'06*, pages 71–82, Alexandria, Virginia, USA, Oct. 2006.

[95] Stefan Saroiu and Alec Wolman. I am a sensor, and I approve this message. In *HotMobile'10*, pages 37–42, Annapolis, Maryland, Mar. 2010.

[96] Mischa Schwartz. *Mobile Wireless Communications*. Cambridge University Press, 2005.

[97] Mark Shaneck and Yongdae Kim. Efficient cryptographic primitives for private data mining. In *HICSS'10*, pages 1–9, Jan. 2010.

[98] N. Shankar, C. Cordeiro, and K. Challapali. Spectrum agile radios: utilization and sensing architectures. In *IEEE DySPAN'05*, Baltimore, MA, Nov. 2005.

[99] S. Shellhammer. Numerical spectrum sensing requirements. IEEE Std. 802.22-06/0088r0, Jan. 1999.

[100] S. Shellhammer, S. N, R. Tandra, and J. Tomcik. Performance of power detector sensors of DTV signals in IEEE 802.22 WRANs. In *TAPAS'06*, Boston, MA, 2006.

[101] Bo Sheng and Qun Li. Verifiable privacy-preserving range query in sensor networks. In *IEEE INFOCOM'08*, pages 46–50, Phoenix, AZ, Apr. 2008.

[102] Elaine Shi, John Bethencourt, Hubert Chan, Dawn Song, and Adrian Perrig. Multi-dimensional range query over encrypted data. In *IEEE S&P'07*, pages 350–364, Oakland, CA, May 2007.

[103] Jing Shi, Rui Zhang, and Yanchao Zhang. Secure range queries in tiered sensor networks. In *IEEE INFOCOM'09*, Rio de Janeiro, Brazil, Apr. 2009.

[104] Thanos Stathopoulos, John Heidemann, and Deborah Estrin. A remote code update mechanism for wireless sensor networks. Technical report, UCLA, Los Angeles, CA, 2003.

[105] Hailun Tan, Sanjay Jha, Diet Ostry, John Zic, and Vijay Sivaraman. Secure multi-hop network programming with multiple one-way key chains. In *WiSec'08*, pages 183–193, Mar. 2008.

[106] Hailun Tan, Diethelm Ostry, John Zic, and Sanjay Jha. A confidential and DoS-resistant multi-hop code dissemination protocol for wireless sensor networks. In *WiSec'09*, pages 245–252, Zurich, Switzerland, Mar. 2009.

[107] R. Tandra and A. Sahai. SNR walls for signal detection. *IEEE Journal of Selected Topics in Signal Processing*, 2(1):4–17, 2008.

[108] Arvind Thiagarajan, Lenin Ravindranath, Katrina LaCurts, Samuel Madden, Hari Balakrishnan, Sivan Toledo, and Jakob Eriksson. VTrack: accurate, energy-aware road traffic delay estimation using mobile phones. In *ACM SenSys'08*, pages 85–98, Berkeley, CA, Nov. 2009.

[109] V. Tuulos, J. Scheible, and H. Nyholm. Combining web, mobile phones and public displays in large-scale: Manhattan story mashup. In *the Fifth International Conference on Pervasive Computing*, pages 37–54, Toronto, Canada, May 2007.

[110] Osman Ugus, Dirk Westhoff, and Jens-Matthias Bohli. A ROM-friendly secure code update mechanism for WSNs using a stateful-verifier τ-time signature scheme. In *WiSec'09*, pages 29–40, Zurich, Switzerland, Mar. 2009.

[111] A. Wald. *Sequential Analysis*. Dover Publications, 2004.

[112] Chuang Wang, Guiling Wang, Wensheng Zhang, and Taiming Feng. Reconciling privacy preservation and intrusion detection in sensory data aggregation. In *INFOCOM'11*, pages 336–340, Shanghai, China, Apr. 2011.

[113] Haodong Wang, Bo Sheng, Chiu Tan, and Qun Li. WM-ECC: an Elliptic Curve Cryptography Suite on Sensor Motes. Technical Report WM-CS-2007-11, College of William and Mary, Computer Science, Williamsburg, VA, 2007.

[114] Dirk Westhoff, Joao Girao, and Mithun Acharya. Concealed data aggregation for reverse multicast traffic in sensor networks: Encryption, key distribution, and routing adaptation. *IEEE Transactions on Mobile Computing*, 5(10):1417–1431, Oct. 2006.

[115] Anthony Wood and John Stankovic. Online coding for reliable data transfer in lossy wireless sensor networks. In *DCOSS'09*, pages 159–172, Marina del Rey, CA, June 2009.

[116] Qiben Yan, Ming Li, Tingting Jiang, Wenjing Lou, and Y.T. Hou. Vulnerability and protection for distributed consensus-based spectrum sensing in cognitive radio networks. In *INFOCOM'12*, pages 900–908, Orlando, FL, Mar. 2012.

[117] L. Yang, Z. Zhang, B. Zhao, C. Kruegel, and H. Zheng. Enforcing dynamic spectrum access with spectrum permits. In *ACM MobiHoc'12*, Hilton Head Island, SC, June 2012.

[118] Yi Yang, Xinran Wang, Sencun Zhu, and Guohong Cao. SDAP: a secure hop-by-hop data aggregation protocol for sensor networks. In *ACM MobiHoc'06*, pages 356–367, Florence, Italy, May 2006.

[119] Yin Yang, Stavros Papadopoulos, Dimitris Papadias, and George Kollios. Spatial outsourcing for location-based services. In *IEEE ICDE*, pages 1082–1091, Cancún, México, Apr. 2008.

[120] Zhimin Yang, Boying Zhang, Jiangpeng Dai, A.C. Champion, Dong Xuan, and Du Li. E-SmallTalker: A distributed mobile system for social networking in physical proximity. In *ICDCS'10*, pages 468–477, Genoa, Italy, June 2010.

[121] Andrew Yao. Protocols for secure computations. In *SFCS'82*, pages 160–164, Washington, DC, USA, Nov 1982.

[122] Qingsong Ye, Huaxiong Wang, and Josef Pieprzyk. Distributed private matching and set operations. In *ISPEC'08*, volume 4991, pages 347–360, Sydney, Australia, Apr. 2008.

[123] M. Yiu, Yimin Lin, and Kyriakos Mouratidis. Efficient verification of shortest path search via authenticated hints. In *IEEE ICDE*, pages 237–248, Long Beach, CA, Mar. 2010.

[124] M. Yiu, Eric Lo, and Duncan Yung. Authentication of moving kNN queries. In *IEEE ICDE*, pages 565–576, Hannover, Germany, Apr. 2011.

[125] F.R. Yu, H. Tang, Minyi Huang, Zhiqiang Li, and P.C. Mason. Defense against spectrum sensing data falsification attacks in mobile ad hoc networks with cognitive radios. In *MILCOM'09*, oct. 2009.

[126] Haifeng Yu. Secure and highly-available aggregation queries in large-scale sensor networks via set sampling. In *IPSN'09*, pages 1–12, Washington, DC, Apr. 2009.

[127] Haifeng Yu, Phillip Gibbons, Michael Kaminsky, and Feng Xiao. SybilLimit: a near-optimal social network defense against sybil attacks. *IEEE/ACM Trans. Netw.*, 18:885–898, June 2010.

[128] Haifeng Yu, Michael Kaminsky, Phillip Gibbons, and Abraham Flaxman. SybilGuard: Defending against sybil attacks via social networks. *IEEE/ACM Transactions on Networking*, 16(3):576–589, June 2008.

[129] Shucheng Yu, Cong Wang, Kui Ren, and Wenjing Lou. Achieving secure, scalable, and fine-grained access control in cloud computing. In *IEEE INFOCOM'10*, San Diego, CA, Mar. 2010.

[130] Kun Zeng, P. Paweczak, and D. Cabric. Reputation-based cooperative spectrum sensing with trusted nodes assistance. *IEEE Communications Letters*, 14(3):226 –228, march 2010.

[131] L. Zhang, B. Tiwana, Z. Qian, Z. Wang, R. Dick, Z. Mao, and L. Yang. Accurate online power estimation and automatic battery behavior based power model generation for smartphones. In *CODES/ISSS'10*, pages 105–114, Scottsdale, AZ, 2010.

[132] Rui Zhang, Yunzhong Liu, Yanchao Zhang, and Xiaoxia Huang. JR-SND: jamming-resilient secure neighbor discovery in mobile ad-hoc networks. In *IEEE ICDCS'11*, Minneapolis, Minnesota, June 2011.

[133] Rui Zhang, Jing Shi, Yunzhong Liu, and Yanchao Zhang. Verifiable fine-grained top-k queries in tiered sensor networks. In *INFOCOM'10*, San Diego, CA, Mar. 2010.

[134] Rui Zhang, Jing Shi, and Yanchao Zhang. Secure multidimensional range queries in sensor networks. In *ACM MobiHoc'09*, pages 197–206, New Orleans, LA, May 2009.

[135] Rui Zhang and Yanchao Zhang. LR-Seluge: Loss-resilient and secure code dissemination in wireless sensor networks. Technical report, Arizona State University, 2011.

[136] Wensheng Zhang, Chuang Wang, and Taiming Feng. Gp^{2}s: Generic privacy-preservation solutions for approximate aggregation of sensor data (concise contribution). In *PerCom'08*, pages 179–184, Hong Kong, China, Mar. 2008.

[137] Yanchao Zhang and Yuguang Fang. ARSA: an attack-resilient security architecture for multi-hop wireless mesh networks. *IEEE J. Select. Areas Commun., Special Issue on High-Speed Network Security - Architecture, Algorithms, and Implementation*, 24(10):1916–1928, Oct. 2006.

[138] Yanchao Zhang and Yuguang Fang. A fine-grained reputation system for reliable service selection in peer-to-peer networks. *IEEE Trans. Parallel Distrib. Syst.*, 18(8):1134–1145, Aug. 2007.

[139] Yanchao Zhang, Wei Liu, Wenjing Lou, and Yuguang Fang. Securing mobile ad hoc networks with certificateless public keys. *IEEE Transactions on Dependable and Secure Computing,*, 3(4):386–399, Oct.-Dec. 2006.

[140] S. Zhu, S. Setia, and S. Jajodia. LEAP: Efficient security mechanisms for large-scale distributed sensor networks. In *ACM CCS'03*, pages 62–72, Washington, DC, Oct. 2003.

[141] Qiyue Zou, Songfeng Zheng, and A.H. Sayed. Cooperative sensing via sequential detection. *IEEE Transactions on Signal Processing*, 58(12):6266–6283, 2010.

APPENDIX A

Proof of Theorem 2.5.1

Proof. With Seluge, the sender has to keep (re)transmitting every packet until it is received by all N nodes. Let $\mathbf{T}_j$ be the number of transmissions needed for all N receivers to receive the j-th packet, $\forall 1 \leq j \leq k$. After the t-th transmission of the j-th packet, the probability that node i has received it is $1 - p_i^t$, and the probability that all N nodes have received it is thus

$$Pr(\mathbf{T}_j \leq t) = \prod_{i=1}^{N}(1 - p_i^t) \ . \tag{7.1}$$

Therefore, the probability that it takes exactly t transmissions for all N nodes to receive the j-th packet can be computed as

$$\begin{aligned} Pr(\mathbf{T}_j = t) &= Pr(\mathbf{T}_j \leq t) - Pr(\mathbf{T}_j \leq t-1) \\ &= \prod_{i=1}^{N}(1 - p_i^t) - \prod_{i=1}^{N}(1 - p_i^{t-1}) \ . \end{aligned} \tag{7.2}$$

The expected number of transmissions for the j-th packet is then $\mathsf{E}[\mathbf{T}_j] = \sum_{t=1}^{\infty} t \cdot Pr(\mathbf{T}_j = t)$. Since different packets are independent from each other, the expected number of total transmissions to transmit k packets is then $\mathsf{E}[\sum_{j=1}^{k} \mathbf{T}_j] = k\mathsf{E}[\mathbf{T}_j]$. □

APPENDIX B

Proof of Theorem 2.5.2

Proof. In LR-Seluge, the greedy round-robin scheduling at local senders makes it very difficult to directly analyze the performance of LR-Seluge. Instead, we analyze a variation of LR-Seluge whose communication cost can be viewed as the upper bound of LR-Seluge. In this variation, instead of using SNACK packets, each receiver returns an ACK packet only after receiving k' packets of the current page. Since the local sender has no information about which packets are missing at each node, it has to repeatedly transmit the n erasure-coded packets until receiving an ACK from every node. This variation is apparently less efficient than LR-Seluge, as the sender may unnecessarily transmit some packets which have reached all the receivers.

Now we analyze how many rounds are needed for the local sender to transmit all the n encoded packets such that each of the N nodes can receive at least k' packets for successfully decoding the original page. Let $\mathbf{R}$ and $\mathbf{R}_i$ be the number of rounds needed for all the N nodes and node i to receive at least k' encoded packets, respectively. After r rounds, the probability that node i has received a given packet is $1-p_i^r$, and the probability that node i has received at least k' encoding packets is given by

$$Pr(\mathbf{R}_i \leq r) = \sum_{j=k'}^{n} \binom{n}{j} (1-p_i^r)^j p_i^{r(n-j)} \ , \tag{7.3}$$

The probability that all nodes have received at least k' packets after r rounds can then be computed as

$$Pr(\mathbf{R} \leq r) = \prod_{i=1}^{N} Pr(\mathbf{R}_i \leq r) \ . \tag{7.4}$$

It follows that

$$\mathsf{E}[\mathbf{R}] = \sum_{r=1}^{\infty} r \cdot (Pr(\mathbf{R} \leq r) - Pr(\mathbf{R} \leq r-1)) \ . \tag{7.5}$$

Since each round involves n packet transmissions, the expected total number of transmissions for the above LR-Seluge variant is $n\mathsf{E}[\mathbf{R}]$, which upper-bounds that of LR-Seluge. □

APPENDIX C

Proof of Theorem 3.6.1

Proof. We first show that the user can detect any inauthentic query result containing fake POI records or indexes. Recall that the hash of every record is embedded in its index, and adjacent indexes are chained together. Therefore, any fake record or index will make the user compute an invalid leaf node for the Merkle hash tree, which can be immediately detected by the user after verifying the data collector's non-forgeable signature on the Merkle root hash.

Now we show that incorrect query results can also be detected. The key rationale is that if the LBSP returns $X_{i,j} = D'_{i,j}$ or $\phi_{i,j}$, he must also return $X_{i,1}, \cdots, X_{i,j-1}$ for the query result to pass the authenticity check. Let kPOI denote the correct top-k records with the lowest attribute rating γ and $\widetilde{\text{kPOI}}$ the incorrect top-k records with the lowest attribute rating $\tilde{\gamma} \neq \gamma$. If $\tilde{\gamma} < \gamma$, there must be at least k POI records with attribute-q ratings higher than $\tilde{\gamma}$ in the query region, which should all be returned for $\widetilde{\text{kPOI}}$ to pass the authenticity check. This apparently contradicts with the fact that $\tilde{\gamma}$ is the lowest rating in $\widetilde{\text{kPOI}}$. If $\tilde{\gamma} > \gamma$ instead, $\widetilde{\text{kPOI}}$ must contain at least one POI record outside the query region with attribute-q rating higher than γ, which can be directly detected. □

APPENDIX D

Proof of Theorem 3.6.2

Before go to the proof of Theorem 3.6.2, We first we need to prove the following lemmas.

Lemma 7.1.1. Assume that $Y_1, Y_2, \cdots, Y_n$ are i.i.d. random variables uniformly distributed in the range $[0, 1]$. Let $\tilde{Y}_1, \tilde{Y}_2, \cdots, \tilde{Y}_n$ be the random variables defined by sorting the values (realizations) of $Y_1, Y_2, \cdots, Y_n$ in decreasing order. Then $\tilde{Y}_k \sim Be(n-k+1, k)$ is a Beta random variable with p.d.f.

$$f(x, n-k+1, k) = \frac{\Gamma(n+1)}{\Gamma(n-k+1)\Gamma(k)} x^{n-k}(1-x)^{k-1} , \tag{7.6}$$

where $\Gamma(t) = (t-1)!$.

Proof. Consider an interval $[x, x+dx]$. For $\tilde{Y}_k$ to be in $[x, x+dx]$, it is necessary that exactly $k-1$ elements are larger than $x+dx$, and that at least one is between x and $x+dx$. Since the probability that more than one is in $[x, x+dx]$ is already $O(dx^2)$, the probability of $\tilde{Y}_k$ falling in the interval $[x, x+dx]$ is equal to the probability that exactly $k-1$, 1 and $n-k$ elements fall in the intervals $(x+dx, 1), (x, x+dx)$ and $(0, x)$ respectively, which is given by

$$P(\tilde{Y}_n \in (x, x+dx)) = \frac{n!}{(n-k)!(k-1)!}(1-x-dx)^{k-1}x^{n-k}dx ,$$

and the result follows. □

Lemma 7.1.2. The expected number of POIs with rating higher than γ in each zone is given by

$$\mathsf{E}[\tau] = \frac{kn}{\delta n + 1} , \tag{7.7}$$

where δ is the ratio between the areas of the query region and a single zone.

Proof. Assume that there are n POIs uniformly distributed in each zone. The expected number of POIs in the query region can be estimated as δn. According to Lemma 7.1.1, the kth highest rating within the query region is a Beta random variable, i.e., $\gamma \sim Be(\delta n - k + 1, k)$. It follows that

$$\mathsf{E}[\gamma] = \frac{\delta n - k + 1}{\delta n + 1} . \tag{7.8}$$

We therefore have

$$\mathsf{E}[\tau] = n(1 - \mathsf{E}[\gamma]) = \frac{kn}{\delta n + 1} \ . \tag{7.9}$$

□

Lemma 7.1.3. The expected number of hash computations needed to verify $\{h_{i,1}\}_{i \in \mathcal{I}}$ using Merkle hash tree is given by

$$\mathsf{E}(N_{merkle}) = \sum_{j=1}^{d-1} 2^{j-1}(1 - (1 - 2^{-(j-1)})^{|\mathcal{I}|}) \ . \tag{7.10}$$

Proof. In the Merkle hash tree, the number of nodes at level j is 2^{j-1} for all $j \in [1, d]$. To verify $h_{i,1}$, the user need to compute each internal node along the path from $h_{i,1}$ to the root of the tree.

For each node at level j, the probability that it is on the path between a randomly chosen leaf node and the root hash is $2^{-(j-1)}$. It follows that it is on at least one of the path between $\{h_{i,1}\}_{i \in \mathcal{I}}$ and the root hash, and thus appears in $\bigcup_{i \in \mathcal{I}} \mathcal{T}_i$, with probability $(1 - 2^{-(j-1)})^{|\mathcal{I}|}$. The expected number of nodes at level j in $\bigcup_{i \in \mathcal{I}} \mathcal{T}_i$ is then $2^{j-1}(1 - (1 - 2^{-(j-1)})^{|\mathcal{I}|})$. Therefore, the expected number of hash computations needed to verify $\{h_{i,1}\}_{i \in \mathcal{I}}$ is given by

$$\mathsf{E}(N_{\text{merkle}}) = \sum_{j=1}^{d-1} 2^{j-1}(1 - (1 - 2^{-(j-1)})^{|\mathcal{I}|}) \ .$$

□

Now we are ready to prove Theorem 3.6.2.

Proof. The total number of hash computations needed by Scheme 1 consists of three parts. First, the user need perform one hash computation for each returned data record to generate the corresponding index, leading to k hash computations. Second, for each zone $i \in \mathcal{I}$, the user need compute $h_{i,1}$, which incurs $\tau_i + 1$ hash computations, where τ_i can be estimated as $\mathsf{E}[\tau]$ derived in Eq. (7.7). Finally, the user need perform N_{merkle}

hash computations to verify $\{h_{i,1}\}_{i\in\mathcal{I}}$. Summing up these three part, we have

$$\mathsf{E}[N_{\mathrm{hash},1}] = k + |\mathcal{I}| \cdot (\mathsf{E}[\tau] + 1) + \sum_{j=1}^{d-1} 2^{j-1}(1 - (1 - 2^{-(j-1)})^{|\mathcal{I}|}) \ ,$$

where $\mathsf{E}[\tau]$ and N_{merkle} are given in Eqs. (7.7) and (7.10), respectively. □

APPENDIX E

Proof of Theorem 3.6.3

The following lemma is needed in proving Theorem 3.6.3.

Lemma 7.1.4. The expected number of distinct elements in $\bigcup_{i\in\mathcal{I}} \mathcal{T}_i$ is given by

$$|\bigcup_{i\in\mathcal{I}} \mathcal{T}_i| = \sum_{j=1}^{d-1} 2^j (1-(1-2^{-j})^{|\mathcal{I}|}) \tag{7.11}$$

The proof is similar to that of Lemma 7.1.3 and thus omitted.

We now prove Theorem 3.6.3.

Proof. The communication overhead between the data collector and the LBSP is given by

$$\mathsf{T}_1 = \mathsf{T}_{\mathrm{id}} + \mathsf{T}_{\mathrm{index}} + \mathsf{T}_{\mathrm{auth}} \ , \tag{7.12}$$

where $\mathsf{T}_{\mathrm{id}}, \mathsf{T}_{\mathrm{cert}}, \mathsf{T}_{\mathrm{auth}}$ are the communication overhead incurred by transmitting zone IDs, indexes and authentication information (i.e., hash images and data collector's signature), respectively.

For each zone $i \in \mathcal{I}$, the LBSP need return the corresponding zone ID to the user, so we have

$$\mathsf{T}_{\mathrm{id}} = |\mathcal{I}| \cdot d \ . \tag{7.13}$$

In addition, the LBSP need return $\tau_i + 1$ POIs for each zone $i \in \mathcal{I}$. The total number of POIs need be returned can be estimated as $\sum_{i\in\mathcal{I}}(\tau_i + 1)$. Among them, k POIs returned as data records, and rest of them are indexes. It follows that

$$\begin{aligned} \mathsf{T}_{\mathrm{index}} &= (\sum_{i\in\mathcal{I}}(\tau_i+1) - k) \cdot (L_{\mathrm{loc}} + L_{\mathrm{r}} + L_{\mathrm{h}}) \\ &= (|\mathcal{I}| \cdot (\mathsf{E}[\tau]+1) - k) \cdot (L_{\mathrm{loc}} + L_{\mathrm{r}} + L_{\mathrm{h}}) \ , \end{aligned} \tag{7.14}$$

where $\mathsf{E}[\tau]$ is given in Eq. (7.7).

Finally, the LBSP need return one hash image, i.e., h_{i,τ_i+2}, for each zone $i \in \mathcal{I}$, $\bigcup_{i\in\mathcal{I}} \mathcal{T}_i$ for $\{h_{i,1}\}_{i\in\mathcal{I}}$, and the data collector's signature on the root of the Merkle hash tree. We therefore have

$$\mathsf{T}_{\mathrm{auth}} = (|\mathcal{I}| + |\bigcup_{i\in\mathcal{I}} \mathcal{T}_i|) \cdot L_{\mathrm{h}} + L_{\mathrm{sig}} \ , \tag{7.15}$$

where $|\bigcup_{i\in\mathcal{I}} \mathcal{T}_i|$ is given by Eq. (7.11). Substituting Eqs. (7.13), (7.14) and (7.15) into Eq. (7.12), we have

$$\begin{aligned}\mathsf{E}[\mathsf{T}_1] &= (|\mathcal{I}| \cdot (\mathsf{E}[\tau] + 1) - k) \cdot (L_{\mathrm{loc}} + L_{\mathrm{r}} + L_{\mathrm{h}}) + |\mathcal{I}| \cdot d \\ &\quad + \sum_{j=1}^{d-1} 2^j (1 - (1 - 2^{-j})^{|\mathcal{I}|}) \cdot L_{\mathrm{h}} + L_{\mathrm{sig}} \ ,\end{aligned}$$

where $\mathsf{E}[\tau] = \frac{kn}{\delta n+1}$, and the theorem is proved. □

APPENDIX F

Proof of Theorem 3.6.4

Proof. As in the proof of Theorem 3.6.1, the user can easily detect any inauthentic query result containing fake POI records or indexes. The proof is omitted here for brevity.

Now assume that the LBSP returns an authentic but incorrect query result, from which the user derives incorrect top-k POI records $\widetilde{\text{kPOI}}$ with the lowest attribute rating $\tilde{\gamma}$. Against, let γ denote the correct lowest top-k attribute rating. If $\tilde{\gamma} > \gamma$, we can apply the same argument in proving Theorem 3.6.1 to show that the user can discover that at least one POI in $\widetilde{\text{kPOI}}$ is outside the query region. If $\tilde{\gamma} < \gamma$, the LBSP should have deleted at least one POI record in the query region with an attribute rating higher than $\tilde{\gamma}$. Suppose that the LBSP did not return $D'_{i,j}$ with $A'_{i,j,q} > \tilde{\gamma}$ in the macro zone e. There are two cases.

- If the LBSP returned nothing from zone i, it must have returned at least one index with a rating $< \tilde{\gamma}$ in the macro zone e, say ϕ_{i_1,j_1}. It follows that $A'_{i,1,q} > A'_{i,j,q} > \tilde{\gamma} > A'_{i_1,j_1,q}$ and $\langle j, A'_{i,1,q} \rangle \in \bigcup_{x=1}^{j_1} \mathcal{I}_{i_1,x}$, from which the user knows that the LBSP omitted valid information from zone i.

- If the LBSP returned some POI records or indexes for zone i, it must have returned $X_{i,1}, \ldots, X_{i,\tilde{\tau}_i+1}$ to pass the authenticity check. Since $A'_{i,j,q} > \tilde{\gamma}$, we have $j < \tilde{\tau}_i$, and $D'_{i,j}$ or $\phi_{i,j}$ must have been returned, leading to a contradiction.

Therefore, the user can detect any incorrect query result. □

APPENDIX G

Proof of Theorem 3.6.5

Proof. Similar to Scheme 1, the number of hash computations needed by Scheme 2 consists of three parts.

First, the user need perform k hash computations to obtain the index for the returned data records. Second, for each zone i with $\tau_i > 0$, the user need perform $\tau_i + 1$ hash computations to obtain $h_{i,1}$.

Let $\mathcal{I}' \in \mathcal{I}$ be the set of zones in the query region that has POI with rating higher than γ, i.e., $\tau_i > 0$. We now estimate $|\mathcal{I}'|$. The probability that zone $i \in \mathcal{I}$ has no POI with rating higher than γ is γ^n. The probability that zone $i \in \mathcal{I}$ has at least one POI with rating higher than γ is then $1 - \gamma^n$. It follow that the expected number of zones in the query region that has at least one POI with rating higher than γ is given by

$$|\mathcal{I}|' = |\mathcal{I}| \cdot (1 - \gamma^n) \ , \tag{7.16}$$

where $\gamma = (\delta n - k + 1)/(\delta n + 1)$.

For each zone $i \in \mathcal{I}'$, the conditional probability that $\tau_i = x$ given that $\tau_i > 0$ can be computed as

$$\begin{aligned} P(\tau_i = x | \tau_i > 0) &= \frac{P(\tau_i = x, \tau_i > 0)}{P(\tau_i > 0)} \\ &= \begin{cases} \frac{\binom{n}{x}(1-\gamma)^x \gamma^{n-x}}{1-\gamma^n} & \text{if } 1 \leq x \leq n, \\ 0 & \text{otherwise} \ . \end{cases} \end{aligned} \tag{7.17}$$

It follows that

$$\mathsf{E}[\tau_i | \tau_i > 0] = n(1-\gamma)/(1-\gamma^n) \ . \tag{7.18}$$

The expected number of hash computations required to obtain $\{h_{i,1} | i \in \mathcal{I}'\}$ is then given by

$$\begin{aligned} \mathsf{E}[N_2] &= \sum_{i \in \mathcal{I}'} (\tau_i + 1) \\ &= |\mathcal{I}| \cdot (1 - \gamma^n) \cdot (\frac{n(1-\gamma)}{1-\gamma^n} + 1) \\ &= |\mathcal{I}|(n - n\gamma + 1 - \gamma^n) \ . \end{aligned} \tag{7.19}$$

Finally, the user also need to verify $\{h_{i,1}|i \in \mathcal{I}'\}$ using Merkle hash tree. The number of hash computation required is given by

$$\mathsf{E}[N_{\text{merkle}}] = \sum_{j=1}^{d-1} 2^{j-1}(1-(1-2^{-(j-1)})^{|\mathcal{I}|'}) \ . \tag{7.20}$$

We therefore have

$$\begin{aligned}\mathsf{E}[N_{\text{hash},2}] &= k + \mathsf{E}[N_2] + \mathsf{E}[N_{\text{merkle}}] \\ &= k + |\mathcal{I}|(n - n\gamma + 1 - \gamma^n) \\ &\quad + \sum_{j=1}^{d-1} 2^{j-1}(1-(1-2^{-(j-1)})^{|\mathcal{I}'|}) \ ,\end{aligned}$$

where $\gamma = (\delta n - k + 1)/(\delta n + 1)$ and $|\mathcal{I}'|$ is given in Eq. (7.16). □

APPENDIX H

Proof of Theorem 3.6.6

Proof. Similar to that of Scheme 1, the communication overhead between the LBSP and user is given by

$$\mathsf{T}_1 = \mathsf{T}_{\mathrm{id}} + \mathsf{T}_{\mathrm{embed}} + \mathsf{T}_{\mathrm{index}} + \mathsf{T}_{\mathrm{auth}} \ , \tag{7.21}$$

where $\mathsf{T}_{\mathrm{embed}}$ is the communication overhead incurred by transmitting embedded $\langle j, A'_{j,1,q}\rangle$.

We first estimate $\mathsf{T}_{\mathrm{index}}$. Recall that the expected number of zones $\tau_i > 0$ is $\check{m}(1-\gamma^n)$. For each of them, the LBSP need return $\tau_i + 1$ POIs. It follows that

$$\begin{aligned}\mathsf{T}_{\mathrm{index}} &= \check{m}(1-\gamma^n)(\frac{n(1-\gamma)}{1-\gamma^n}+1)\cdot(L_{\mathrm{loc}}+L_{\mathrm{r}}+L_{\mathrm{h}}) \\ &= \check{m}(n-n\gamma+1-\gamma^n)\cdot(L_{\mathrm{loc}}+L_{\mathrm{r}}+L_{\mathrm{h}}) \ .\end{aligned} \tag{7.22}$$

Now we estimate $\mathsf{T}_{\mathrm{auth}}$. Similar to that in Scheme 1, we have

$$\mathsf{T}_{\mathrm{auth}} = (\check{m}(1-\gamma^n) + \sum_{j=1}^{d-1} 2^j(1-(1-2^{-j})^{\check{m}(1-\gamma^n)}))\cdot L_{\mathrm{h}} + L_{\mathrm{sig}} \ . \tag{7.23}$$

For T_{id}, we have

$$\mathsf{T}_{\mathrm{id}} = \check{m}(1-\gamma^n)\cdot d \ , \tag{7.24}$$

For $\mathsf{T}_{\mathrm{embed}}$, we further divide it into two parts: $\mathsf{T}_{\mathrm{embed},1}$, the $\langle j, A_{j,1}\rangle$s from the $\check{m}$ zones in the query region, and $\mathsf{T}_{\mathrm{embed},2}$, the $\langle j, A_{j,1}\rangle$s from the $m-\check{m}$ zones outside the query region.

For $\mathsf{T}_{\mathrm{embed},1}$, there are two cases. First, when $k < \check{m}$, there are at most k zones with $\tau_i > 0$. Each of these k zones at most embeds the information from other $k-1$ zones in $\mathcal{I}$. Second, when $k \geq \check{m}$, there are at most $\check{m}$ zones with $\tau_i > 0$. Each of the $\check{m}$ at most embed the information from other $\check{m}-1$ zones in $\mathcal{I}$. We therefore have

$$\mathsf{T}_{\mathrm{embed},1} \leq t(t-1)\cdot(d+L_{\mathrm{r}}) \ , \tag{7.25}$$

where $t = \min(k, \check{m})$.

Now we estimate $\mathsf{T}_{\mathrm{embed},2}$. Take zone $i \in \mathcal{I}'$ as an example. Let $\mathcal{I}'_{i,1}, \mathcal{I}'_{i,2}, \cdots, \mathcal{I}'_{i,n_i+1}$ denote the sets of zone IDs outside the query region whose highest POI with attribute-q rating in the range $(\overline{\chi}, A_{i,1}), (A_{i,1}, A_{i,2}), \cdots, (A_{i,n_i}, \chi)$, respectively. We then have $\sum_{j=1}^{n_i+1} |\mathcal{I}'_{i,j}| = m - \check{m}$.

The c.d.f. of $A'_{i,1,q}$ is given by

$$\begin{aligned} P(A'_{i,1,q} \leq a) &= P(A_{i,1,q} \leq a, A_{i,2,q} \leq a, \cdots, A_{i,n,q} \leq a) \\ &= \prod_{i=1}^{n} P(A_{i,1,q} \leq q) \\ &= a^n \ . \end{aligned}$$

According to Lemma 7.1.1, we have $E(A'_{i,j,q}) = (n-j+1)/(n+1)$. For each zone $i' \in \mathcal{M}_e \setminus \mathcal{I}$, the probability that $i' \in \bigcup_{x=1}^{j} \mathcal{I}'_{i,x}$ is given by

$$\begin{aligned} P(i' \in \bigcup_{x=1}^{j} \mathcal{I}'_{i,x}) &= P(A'_{i',1,q} > \frac{n-j+1}{n+1}) \\ &= 1 - (\frac{n-j+1}{n+1})^n \ . \end{aligned} \tag{7.26}$$

Therefore, the expected number of $\langle j, A_{j,1} \rangle$ in $\bigcup_{x=1}^{j} \mathcal{I}'_{i,x}$ can be estimated as $(m - \check{m})(1 - ((n-j+1)/(n+1))^n)$, we thus have $\sum_{x=1}^{j} |\mathcal{I}'_{i,x}| = (m - \check{m})(1 - ((n-j+1)/(n+1))^n)$. It follows that

$$|\mathcal{I}'_{i,j}| = (m - \check{m})((\frac{n-j+2}{n+1})^n - (\frac{n-j+1}{n+1})^n) \tag{7.27}$$

The total number of $\langle j, A_{j,1,q} \rangle$ appear in these $\tau_i + 1$ POIs is given by

$$\begin{aligned} \sum_{j=1}^{\tau_i+1} |\mathcal{I}'_{i,j}| &= \sum_{x=1}^{\tau_i+1} (m - \check{m})((\frac{n-j+2}{n+1})^n - (\frac{n-j+1}{n+1})^n) \\ &= (m - \check{m})(1 - (\frac{n-\tau_i}{n+1})^n); . \end{aligned} \tag{7.28}$$

We therefore have

$$\begin{aligned} \mathsf{T}_{\text{embed},2} &= \sum_{i \in \mathcal{I}'} |\mathcal{I}'_{i,j}| \\ &= |\mathcal{I}'|(m - \check{m})(1 - (\frac{n - \mathsf{E}[\tau_i | \tau_i > 0]}{n+1})^n) \\ &= \check{m}(1 - \gamma^n)(m - \check{m})(1 - (\frac{n - \mathsf{E}[\tau]}{n+1})^n) \cdot (d + L_{\text{r}}) \ . \end{aligned} \tag{7.29}$$

Substituting Eqs. (7.22), (7.23), (7.24), (7.25) and (7.29) into Eq. (7.21), we have

$$
\begin{aligned}
\mathsf{T}_2 \leq\ & \check{m}(1-\mu^n)d + \check{m}(n - n\mu + 1 - \mu^n)(L_{\mathrm{loc}} + L_{\mathrm{r}} + L_{\mathrm{h}}) \\
& + (\check{m}(1-\mu^n) + \sum_{j=1}^{d-1} 2^j (1 - (1-2^{-j})^{\check{m}(1-\mu^n)}))L_{\mathrm{h}} + L_{\mathrm{sig}} \\
& + \check{m}(1-\mu^n)(m-\check{m})(1 - (\frac{n-\nu}{n+1})^n)(d + L_{\mathrm{r}}) \\
& + t(t-1)(d + L_{\mathrm{r}})\ .
\end{aligned}
$$

where $\mu = \mathsf{E}[\gamma] = (\check{m}n - k + 1)/(\check{m}n + 1), \nu = \mathsf{E}[\tau] = n(1-\mu)/(1-\mu^n)$, and $t = \min(k, \check{m})$. The theorem is proved. □

APPENDIX I

Proof of Theorem 3.6.7

Proof. Suppose that the user has issued a sequence of snapshot top-k queries $Q_1, \ldots, Q_w$ to realize a moving top-k query under Scheme 3. For each received complete query result, the user can easily verify if it contains the authentic and correct top-k POIs according to Scheme 1. The only option left for the LBSP is to purposefully omit some complete query results by returning ACKs for some snapshot top-k queries, for which we will show that it will be detected as well.

Assume that the LBSP has omitted a complete query result including kPOI$_a$ and returns an ACK instead in response to query Q_a. According to the query-processing process, the LBSP need return a complete query result in response to both query Q_1 and Q_w; otherwise, it would be easily detected. So there exist $1 \leq x < a < y \leq w$, such that the user has received complete query results for queries Q_x and Q_y as well as ACKs for queries $Q_{x+1}, \ldots, Q_{y-1}$. We thus have kPOI$_a \neq$ kPOI$_x$ under the assumption about the LBSP's misbehavior.

If kPOI$_a \neq$ kPOI$_x$, there are two possible cases: at least one POI in kPOI$_x$ is not in the ath query region $\mathbf{R}_a$, or all the POIs in kPOI$_x$ are all in $\mathbf{R}_a$, but there is at least one POI in $\mathbf{R}_a$ but not in $\mathbf{R}_x$ with attribute-q rating higher than the lowest attribute-q rating γ_x in $\mathbf{R}_a$. In the former case, the user knows that the LBSP should have returned a complete query result instead of an ACK in response to Q_a, so the LBSP can be detected immediately. In the later case, there must exist at least one POI with rating higher than γ_x in $\mathbf{R}_a$, say POI$_{i,j}$. According to the query-processing process, the complete query result for query Q_y need to include the index $\phi_{i,j}$ for POI$_{i,j}$ According to the query-result verification process, the user will detect that $\phi_{i,j}$ contains attribute-q rating larger than γ_x and a POI location in the suspicion range $\mathbf{S}_{x \to y} = \bigcup_{j=x}^{y-2} \mathbf{P}_j$, thus detecting the misbehavior of the LBSP. □

APPENDIX J

Proof of Theorem 4.4.1

Proof. Assume that the AS receives $\{H(e_i) : i \in \mathcal{U}\}$ during the commitment submission phase, whereby it derives $H(\sum_{i\in\mathcal{U}} e_i) = g^{\sum_{i\in\mathcal{U}}} \mod p$. Suppose that some malicious nodes injected false data during the data-aggregation phase so that the AS receives $e' \neq \sum_{i\in\mathcal{U}} e_i$. The AS cannot detect the false-data injection attack if and only if

$$g^{e'} = g^{\sum_{i\in\mathcal{U}} e_i} \mod p \,.$$

However, for any $y \in \mathbb{Z}_p^*$, there is a unique $x \in [0, p-2]$ such that $g^x = y \mod p$. Since $p > 2^{l+2\lfloor \log_2 n \rfloor + \phi} > \sum_{i\in\mathcal{U}} e_i$, there is no $e' \neq \sum_{i\in\mathcal{U}} e_i$ can satisfy the above equation. It is thus impossible for the adversary to inject false data without being detected under VPA$^+$. □

APPENDIX K

Proof of Theorem 4.4.2

Proof. Consider node i as an example. Under DP, node i's data e_i is exposed if the AS is curious and colludes with all the neighbors of node i in the aggregation tree, which are denoted by $\mathcal{T}_i$. The probability that the AS being curious is M_c/M. Assume that $|\mathcal{T}_i| = N_{\text{tree}}$ and that there are n_c curious nodes. The probability of all nodes in $\mathcal{T}_i$ being curious is $\binom{n_c}{N_{\text{tree}}}\binom{n-n_c}{n_c-N_{\text{tree}}}/\binom{n}{n_c}$. We therefore have

$$P_{\text{exp}} = \frac{M_c}{M} \cdot \frac{\binom{n-n_c}{n_c-N_{\text{tree}}}}{\binom{n}{n_c}} \ . \tag{7.30}$$

□

APPENDIX L

Proof of Theorem 4.4.3

Proof. Under RCS and μCS, node i's data e_i is exposed if all the nodes in $\mathcal{C}_i$, $\mathcal{S}_i$ and $\mathcal{T}_i$ are curious, where $\mathcal{S}_i$ is the set of nodes that choose node i as a cover. For RCS, we have $|\mathcal{C}_i| = t$ and for μCS, we have $|\mathcal{C}_i| = \sum_{x=1}^{\mu} N_x$, where N_x is the number of the x-hop neighbors of node i. Since $\max(|\mathcal{C}_i|, |\mathcal{S}_i|, |\mathcal{T}_i|) \leq |\mathcal{C}_i \bigcup \mathcal{S}_i \bigcup \mathcal{T}_i|$, we have

$$P_{\text{exp}} \leq \frac{\binom{n-n_c}{n_c-w}}{\binom{n}{n_c}}$$

where

$$w = \begin{cases} \max(N_{\text{tree}}, t) & \text{for RCS,} \\ \max(N_{\text{tree}}, \sum_{x=1}^{\mu} N_x) & \text{for } \mu\text{CS .} \end{cases}$$

□

APPENDIX M

Proof of Theorem 4.4.4

Proof. The communication overhead incurred by VPA$^+$ consists of three parts: $\mathsf{T}_{\mathrm{tree}}$, the overhead incurred by forming aggregation tree, $\mathsf{T}_{\mathrm{commit}}$, the overhead incurred by submitting commitment to $\mathcal{A}$, and $\mathsf{T}_{\mathrm{agg}}$, the overhead incurred by privacy-preserving aggregation.

We first estimate $\mathsf{T}_{\mathrm{tree}}$. During the aggregation-tree formation process, each node broadcasts the tree formation request once. Therefore we have

$$\mathsf{T}_{\mathrm{tree}} = nl_{\mathrm{tree}} \ ,$$

where l_{tree} is the length of the tree formation request in bits.

Now we estimate $\mathsf{T}_{\mathrm{commit}}$. Suppose that $H(e)$ and $h(\cdot)$ are of l_{hmac} and l_{h} bits, respectively. The length of each commitment message is $l_{\mathrm{hmac}} + l_{\mathrm{h}}$. Recall that each node ID is of λ bits. We then have

$$\mathsf{T}_{\mathrm{commit}} = n(\lambda + l_{\mathrm{hmac}} + l_{\mathrm{h}}) \ .$$

Finally, we estimate $\mathsf{T}_{\mathrm{agg}}$ for DP, RCS and μCS.

- DP: during the aggregation phase, each node need transmit the intermediate aggregation result to its parent node. We therefore have

  $$\mathsf{T}_{\mathrm{agg}} = nl_{\mathrm{data}} \ ,$$

  where $l_{\mathrm{data}} = l + 2\lfloor \log_2 n \rfloor + \phi$.

- RCS: before in-network aggregation, each node need send one slice to each of the t chosen cover nodes, which involves one route discovery to the chosen cover node. Each route discovery incurs a communication overhead of $nl_{\mathrm{req}} + Ll_{\mathrm{rsp}}$, where l_{req} and l_{rsp} denote the length of a AODV route discovery request and a response, respectively. In addition, transmitting one slice incurs a communication overhead of $L(l_{\mathrm{data}} + \lambda)$. Therefore we have

  $$\mathsf{T}_{\mathrm{agg}} = nt(nl_{\mathrm{req}} + L(l_{\mathrm{rsp}} + l_{\mathrm{data}} + \lambda)) + nl_{\mathrm{data}} \ .$$

- μCS: For μCS, during slicing and mixing, each node i and all the nodes within its $\mu - 1$-hop neighborhood need broadcast a random seed along with ID_i once, which incurs communication overhead of $(1 + \sum_{x=1}^{\mu-1}) N_x(\lambda + l_{\text{seed}})$. In addition, each nodes in its $\mu - 1$-hop neighborhood need return its node ID to node i, which leads to communication overhead $\sum_{x=1}^{\mu} N_x \lambda$. We therefore have

$$\mathsf{T}_{\text{agg}} = n((1 + \sum_{x=1}^{\mu-1} N_x)(\lambda + l_{\text{seed}}) + \sum_{x=1}^{\mu} N_x x \lambda) + n l_{\text{data}} \ .$$

□

APPENDIX N

Proof of Theorem 4.5.1

Before we prove Theorem 4.5.1, we first have the following lemmas.

Lemma 7.1.5. Assume that $d_{\max} = 2^l - 1$. Let $\{\chi_{i,j}\}_{j=1}^{l}$ be the sequence of answers from a node i, where $\chi_{i,j} \in \{0,1\}$ for all $j \in [1,l]$. If $\chi_{i,j} = 1$, then $\chi_{i,j'} = 1$ for all $j' < j$; likewise, if $\chi_{i,j} = 0$, then $\chi_{i,j'} = 0$ for all $j' > j$.

Proof. Assuming that $d_{\max} = 2^l - 1$, we have $Q_j = [d \geq \sum_{i=1}^{j} 2^{l-i}]$ for all $j \in [1,l]$. It follows that $Q_{j+1} \subset Q_j$ for all $j \in [1, l-1]$. Therefore, if $\chi_{i,x} = 1$, then $d_i \in Q_j$. It follows that $d_i \in Q_{j'}$ and $\chi_{i,j'} = 1$ for all $j' < j$. The second part can be proved similarly and is thus omitted. □

Lemma 7.1.6. Assume that $d_{\max} = 2^l - 1$. Let χ_{i,y_e} and χ_{i,n_e} be the first exposed yes answer and last exposed no answer, where $0 \leq y_e < n_e \leq l+1$, $y_e = 0$ and $n_e = l+1$ denote the case that no yes answer is disclosed and the case that no no answer is exposed, respectively. The suspicion ratio after l count queries is given by

$$\rho[y_e, n_e] = \begin{cases} 2^{-y_e} - 2^{-n_e} & \text{if } y_e + 1 \leq n_e \leq l, \\ 2^{-y_e} & \text{if } n_e = l+1. \end{cases} \tag{7.31}$$

Proof. Suppose $Q_{y_e} = [d \geq \sum_{j=1}^{y_e} 2^{l-j}]$ and $\chi_{i,y_e} = 1$. The disclosure of $\chi_{i,y_e} = 1$ let the adversary know that $d_i \geq \sum_{j=1}^{y_e} 2^{l-j}$. Similarly, suppose that n_e exists, i.e., at least one no answer is exposed. The disclosure of $\chi_{i,n_e} = 0$ let the adversary know that $d_i < \sum_{j=1}^{n_e} 2^{l-j}$. It follows that

$$\sum_{j=1}^{y_e} 2^{l-j} \leq d_i < \sum_{j=1}^{n_e} 2^{l-j} \ .$$

Therefore, we have

$$\begin{aligned} \rho[y_e, n_e] &= 2^{-l}\Big(\sum_{j=1}^{n_e} 2^{l-j} - \sum_{j=1}^{y_e} 2^{l-j}\Big) \\ &= \sum_{j=1}^{n_e} 2^{-j} - \sum_{j=1}^{y_e} 2^{-j} \\ &= \sum_{j=y_e+1}^{n_e} 2^{-j} \\ &= 2^{-y_e} - 2^{-n_e} \ . \end{aligned}$$

Note that the disclosure of other answers does not give the adversary additional information, since $Q_{j+1} \subset Q_j$ for all $j \in [1, l-1]$.

In addition, if no "no" answer is disclosed, we have

$$\rho = 2^{-y_e} \ .$$

□

Now we prove Theorem 4.5.1.

Proof. We first consider DP and μCS in which each node i directly interacts with the same set nodes during the whole sequence of l count queries. This means that if $\chi_{i,1}$ is exposed, which happens with probability P_{exp}, then all the subsequent answers $\{\chi_{i,1}\}_{j=2}^{l}$ are also exposed. On the other hand, if $\chi_{i,1}$ is kept secret, which happens with probability $1 - P_{\text{exp}}$, so are $\{\chi_{i,1}\}_{j=2}^{l}$.

We further partition the data range $[0, 2^l - 1]$ into $l+1$ *equivalent classes*, denoted by $\{C_x\}_{x=0}^{l}$, where $C_x = [2^l - 2^{l-x}, 2^l - 2^{l-x-1} - 1]$ for $x \in [0, l-1]$, and $C_l = [2^l - 1, 2^l - 1]$. It follows if $d_i \in C_x$, then $\chi_{i,j} = 1$ for all $j \in [1, x]$, and $\chi_{i,j} = 0$ for all $j \in [x+1, l]$. In other words, different nodes have the same sequence of answers if their data are in the same equivalent classes.

Assume that $d_i \in C_x$. If $\{\chi_{i,1}\}_{j=1}^{l}$ are exposed, which happens with probability P_{exp}, then we have $y_e = x$ and $n_e = x + 1$. According to Eq. (4.23), we have

$$\rho = \begin{cases} 2^{-x-1} & \text{if } 0 \leq x \leq l-1, \\ 2^{-l} & \text{if } x = l, \end{cases}$$

in this case. If $\{\chi_{i,1}\}_{j=1}^{l}$ are kept secret, which happens with probability $1 - P_{\text{exp}}$, then we have $\rho = 1$.

Combining the above two cases, we can compute the expected suspicion ratio as

$$
\begin{aligned}
\rho &= \sum_{x=0}^{l-1} Pr(d_i \in C_x)((1 - P_{\text{exp}}) + 2^{-x-1} P_{\text{exp}}) \\
&\quad + Pr(d_i \in C_l)((1 - P_{\text{exp}}) + 2^{-l} P_{\text{exp}}) \\
&= \sum_{x=0}^{l-1} 2^{-x-1}((1 - P_{\text{exp}}) + 2^{-x-1} P_{\text{exp}}) \\
&\quad + 2^{-l}((1 - P_{\text{exp}}) + 2^{-l} P_{\text{exp}}) \\
&= 1 - P_{\text{exp}} + (2^{-2l} + \sum_{x=1}^{l} 2^{-2x}) P_{\text{exp}} \ .
\end{aligned}
\tag{7.32}
$$

□

APPENDIX O

Proof of Theorem 4.5.2

Proof. In RCS, each node i chooses the cover nodes independently for each query. This means that each $\chi_{i,j}$ is exposed independently with probability P_{exp}. Suppose $d_i \in C_x$. So node i returns x yes answers and $l - x$ no answers. The p.d.f. of y_e is then given by

$$Pr(y_e = k) = \begin{cases} (1 - P_{\text{exp}})^x & \text{if } k = 0, \\ P_{\text{exp}}(1 - P_{\text{exp}})^{k-1} & \text{if } 1 \leq k \leq x, \\ 0 & \text{otherwise.} \end{cases} \tag{7.33}$$

Similarly, p.d.f. of n_e can be computed as

$$Pr(n_e = k) = \begin{cases} P_{\text{exp}}(1 - P_{\text{exp}})^{k-x-1} & \text{if } x + 1 \leq k \leq l, \\ (1 - P_{\text{exp}})^{l-x} & \text{if } k = l + 1, \\ 0 & \text{otherwise.} \end{cases} \tag{7.34}$$

For all data in $d_i \in C_x$, the expected suspicion ratio can be computed as

$$\mathsf{E}[\rho_x] = \sum_{k_1=0}^{x} Pr(y_e = k_1) \sum_{k_2=x+1}^{l+1} Pr(n_e = k_2)\rho[y_e, n_e] \ . \tag{7.35}$$

Finally, the expected suspicion ratio can be computed as

$$\begin{aligned} \mathsf{E}[\rho] &= \sum_{x=0}^{l} Pr(d_i \in C_x)\mathsf{E}[\rho_x] \\ &= \sum_{x=0}^{l-1} 2^{-x-1}\mathsf{E}[\rho_x] + 2^{-l}\mathsf{E}[\rho_l] \ . \end{aligned} \tag{7.36}$$

□