Security and Privacy in Mobile Computing: Challenges and Solutions

by

Jingchao Sun

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

Approved March 2017 by the
Graduate Supervisory Committee:

Yanchao Zhang, Chair
Junshan Zhang
Lei Ying
Gail-Joon Ahn

ARIZONA STATE UNIVERSITY

May 2017

ABSTRACT

Mobile devices are penetrating everyday life. According to a recent Cisco report [10], the number of mobile connected devices such as smartphones, tablets, laptops, eReaders, and Machine-to-Machine (M2M) modules will hit 11.6 billion by 2021, exceeding the world's projected population at that time (7.8 billion). The rapid development of mobile devices has brought a number of emerging security and privacy issues in mobile computing. This dissertation aims to address a number of challenging security and privacy issues in mobile computing.

This dissertation makes fivefold contributions. The first and second parts study the security and privacy issues in Device-to-Device communications. Specifically, the first part develops a novel scheme to enable a new way of trust relationship called spatiotemporal matching in a privacy-preserving and efficient fashion. To enhance the secure communication among mobile users, the second part proposes a game-theoretical framework to stimulate the cooperative shared secret key generation among mobile users. The third and fourth parts investigate the security and privacy issues in mobile crowdsourcing. In particular, the third part presents a secure and privacy-preserving mobile crowdsourcing system which strikes a good balance among object security, user privacy, and system efficiency. The fourth part demonstrates a differentially private distributed stream monitoring system via mobile crowdsourcing. Finally, the fifth part proposes VISIBLE, a novel video-assisted keystroke inference framework that allows an attacker to infer a tablet user's typed inputs on the touchscreen by recording and analyzing the video of the tablet backside during the user's input process. Besides, some potential countermeasures to this attack are also discussed. This dissertation sheds the light on the state-of-the-art security and privacy issues in mobile computing.

To My Family.

ACKNOWLEDGMENTS

During my Ph.D. study, I'm really lucky to meet so many terrific people who made my journey of learning and exploring as a valuable and memorable experience in my life.

First and foremost I want to thank my supervisor Dr. Yanchao Zhang. It has been a great honor to be his Ph.D. student. He is a tremendous mentor for me and teaches me, both consciously and unconsciously, how to discover a research problem, find a solution, and write a research paper. I would also like to thank him for encouraging me on research and for allowing me to grow as an independent researcher. I really appreciate his contributions of effort, time, supervision, and funding to make my Ph.D. study both productive and enjoyable. His help and advices on both research as well as on my life are very constructive and priceless.

Second, I would like to express my sincere appreciation and thanks to Dr. Junshan Zhang, Dr. Lei Ying, and Dr. Gail-Joon Ahn for serving on my supervisory committee. They give me many constructive and brilliant comments and suggestions starting from the first year of my graduate study.

Working in an active and helpful research group, ASU Cyber and Network Security Group (CNSG), is also an unforgettable and awesome experience. Dr. Rui Zhang, who is currently a professor in the University of Delaware, helped me a lot both in and out of research. He gave me many inspiring suggestions and detailed coaching in research and writing. He himself is a very good model of a smart, active, and enthusiastic researcher. It is really my great honor to collaborate with such a wonderful person who made my research experience interesting and fruitful. I would also like to thank other members, Jinxue Zhang, Xiaocong Jin, Yimin Chen, Tao Li, Xin Yao, Dianqi Han, and Junwei Zhang, in CNSG for their constructive and inspiring discussions in various research topics and helpful support in my daily life.

Last but not least, I would like to specifically thank my parents and wife. Words can hardly express how grateful I am to them. They sacrificed a lot to support my Ph.D. study. I cannot finish this dissertation without their understanding, support, and encouragement.

TABLE OF CONTENTS

CHAPTER Page

CHAPTER Page

CHAPTER Page

CHAPTER Page

LIST OF TABLES

LIST OF FIGURES

Figure Page

Figure Page

Chapter 1

INTRODUCTION

Mobile devices are penetrating everyday life. According to a recent Cisco report [10], the number of mobile-connected devices such as smartphones, tablets, laptops, eReaders, and Machine-to-Machine (M2M) modules will hit 11.6 billion by 2021, exceeding the world's projected population at that time (7.8 billion). People have been using mobile devices for social activities, entertainment, crowdsourcing, personal health, and many other private/public contexts.

The rapid development of mobile devices has brought a number of emerging security and privacy issues in mobile computing. This dissertation aims to address a number of challenging security and privacy issues in mobile computing and propose corresponding defenses. The rest of this dissertation is as follows.

Chapter 2 studies the problem of establishing trust in Device-to-Device (D2D) communication. D2D communications are emerging due to the explosive growth of mobile devices such as smartphones and tablets. Given the possible presence of attackers, a fundamental challenge in secure D2D communications is to develop sound mobile authentication techniques whereby mobile users can select the most trustworthy D2D communication partners from possibly many candidates. This chapter tackles this open challenge and proposes spatiotemporal matching as a promising enabler for secure D2D communications. Spatiotemporal matching is built upon the location-aware capability of D2D devices. In particular, a mobile user could very easily maintain his spatiotemporal profile recording his continuous whereabouts in time, and the level of his spatiotemporal profile matching that of the other user can be translated into the level of trust they two can have in each other. Since spa-

tiotemporal profiles contain very sensitive personal information, privacy-preserving spatiotemporal matching is needed to ensure that as little information as possible about the spatiotemporal profile of either matching participant is disclosed beyond the matching result. Towards this end, this chapter proposes two novel privacy-preserving spatiotemporal matching protocols, which are thoroughly analyzed and evaluated through detailed simulation studies driven by experimental data.

Chapter 3 investigates the problem of secret key establishment between two adjacent mobile devices, which is crucial for securing emerging D2D communications. As a promising method, cooperative key generation allows two mobile devices to select some common neighbors as relays and directly extract a secret key from the wireless channels among them. A challenging issue that has been overlooked is that mobile devices are often self-interested and reluctant to act as relays without adequate reward in return. In this chapter, we propose SYNERGY, a game-theoretical approach for stimulating cooperative key generation. The underlying idea of SYNERGY is to partition a group of mobile devices into disjoint coalitions such that the mobile devices in each coalition fully collaborate on cooperative key generation. We formulate the group partitioning as a coalitional game and design centralized and also distributed protocols for obtaining the core solution to the game. The performance of SYNERGY is evaluated by extensive simulations.

Chapter 4 considers the problem of secure and privacy-preserving object finding via mobile crowdsourcing. The plummeting cost of Bluetooth tags and the ubiquity of mobile devices are revolutionizing the traditional lost-and-found service. This chapter presents SecureFind, a secure and privacy-preserving object-finding system via mobile crowdsourcing. In SecureFind, a unique Bluetooth tag is attached to every valuable object, and the owner of a lost object submits an object-finding request to many mobile users via the SecureFind service provider. Each mobile user involved searches

his vicinity for the lost object on behalf of the object owner who can infer the location of his lost object based on the responses from mobile users. SecureFind is designed to ensure strong object security such that only the object owner can discover the location of his lost object as well as offering location privacy to mobile users involved. The high efficacy and efficiency of SecureFind are confirmed by extensive simulations.

Chapter 5 presents a novel privacy-preserving and communication-efficient stream monitoring scheme of thresholded percentile statistics. Distributed stream monitoring has numerous potential applications in future smart cities. Communication efficiency and data privacy are two main challenges for distributed stream monitoring services. In this chapter, we propose PriStream, the first communication-efficient and privacy-preserving distributed stream monitoring system for thresholded PERCENTILE aggregates. PriStream allows the monitoring service provider to evaluate an arbitrary function over a desired percentile of distributed data reports and monitor when the output exceeds a predetermined system threshold. Detailed theoretical analysis and evaluations show that PriStream has high accuracy and communication efficiency, and differential privacy guarantees under a strong adversary model.

Chapter 6 proposes VISIBLE, a novel video-assisted keystroke inference framework to infer a tablet user's typed inputs from surreptitious video recordings of tablet backside motion. VISIBLE is built upon the observation that the keystrokes on different positions of the tablet's soft keyboard cause its backside to exhibit different motion patterns. VISIBLE uses complex steerable pyramid decomposition to detect and quantify the subtle motion patterns of the tablet backside induced by a user's keystrokes, differentiates different motion patterns using a multi-class Support Vector Machine, and refines the inference results using a dictionary and linguistic relationship. Extensive experiments demonstrate the high efficacy of VISIBLE for inferring single keys, words, and sentences. In contrast to previous keystroke inference attacks,

VISIBLE does not require the attacker to visually see the tablet user's input process or install any malware on the tablet.

Chapter 7 summarizes our work and presents several possible future work.

Chapter 2

PRIVACY-PRESERVING SPATIOTEMPORAL MATCHING FOR SECURE DEVICE-TO-DEVICE COMMUNICATIONS

2.1 Introduction

Device-to-Device (D2D) communications are emerging due to the explosive growth of smartphones and tablets. In a typical D2D communication session, two physically proximate mobile devices can directly communicate without involving the base station. D2D communications are widely expected to enhance spectrum efficiency and system throughput, enable efficient cellular traffic offloading, improve energy efficiency and network coverage, and stimulate excitingly new services [96, 116].

Sound mobile authentication techniques are needed for secure and effective D2D communications. In particular, a mobile user interested in initiating a D2D communication session in crowded places may have many candidate D2D partners to choose from, consisting of normal users and possibly attackers. It is thus crucial for the initiating user to select the most trustworthy candidate(s) to ensure effective and secure D2D communications. For example, if an attacker is chosen by mistake, the attacker can obtain sensitive information from the initiating user and also refuse to collaborate in the way he initially agreed to. Such pitfalls can be largely avoided if the initiating user only considers the candidate D2D partners who can be reliably authenticated.

Traditional mobile authentication techniques are insufficient for D2D communications. Specifically, one may think about letting the initiating user seek help from the trusted base station to select trustworthy D2D partners. This approach would place too much burden on base stations and largely offset the benefits of conducting

D2D communications. Another plausible approach is to equip every D2D user with a public-key certificate and let the initiating user choose the neighbors with valid public-key certificates. This approach, however, does not permit the initiating user to further distinguish potentially many candidates having a valid certificate.

We propose *spatiotemporal matching* as a promising enabler for secure D2D communications. This technique is motivated by the fact that almost all target D2D devices are location-aware through cellular, WiFi, or GPS technology. A mobile user thus can conveniently maintain his *spatiotemporal profile* recording his continuous whereabouts in time, and the level of his spatiotemporal profile matching that of another mobile user can be translated into the level of trust they two can have in each other. For example, if Alice and Bob discover via spatiotemporal matching that they often go to the same coffee shop or take the same train in the same period, it is natural for Alice to trust Bob over another person whom she only met once before. Spatiotemporal matching is naturally well suited for D2D communications. In particular, if two mobile users have very similar spatiotemporal profiles, it is much more likely that they will stay in each other's communication range for longer time, leading to a longer-live D2D communication session.

There are two critical requirements for releasing the full potential of spatiotemporal matching. In particular, spatiotemporal profiles contain very sensitive personal information, and incautiously disclosing them to the public may cause severe consequences. For example, if an employer surreptitiously discovers an employee's frequent patronage of night clubs, the employee may get unfair treatment at the workplace; if a thief knows the routine of a target victim, he could break in when the victim will be away for a long time. It is thus crucial to have *privacy-preserving* spatiotemporal matching, which ensures that as little information as possible about the spatiotemporal profile of either participant is disclosed beyond the matching result. In addition,

spatiotemporal matching is directly performed on mobile devices and thus needs to be very *efficient* in both communication and computation.

We make three main contributions in this chapter. First, we coin privacy-preserving spatiotemporal matching as a fundamental primitive for secure D2D communications. Second, we present two solutions towards efficient privacy-preserving spatiotemporal matching. The first solution is a passive approach, in which every mobile user periodically records his locations, and a user's spatiotemporal profile is defined as a set of (time, location) pairs. The second solution is an active approach, where every mobile user continuously broadcasts cryptographic tokens and also records every token he overhears. The tokens a user broadcasts and receives form his spatiotemporal profile. Third, we propose two protocols for the privacy-preserving comparison of two arbitrary active/passive spatiotemporal profiles. The first protocol is based on a novel use of the Bloom filter [31] to enable either user to estimate with tunable accuracy the number of common elements in their spatiotemporal profiles without disclosing too much private information to each other. The second protocol generalizes the first protocol and enables weighted spatiotemporal matching by allowing each user to assign different weights to different elements in his/her profile to obtain the weighted matching result. In addition, we thoroughly analyze both protocols and also evaluate them via detailed simulations driven by experimental data.

The rest of this chapter is organized as follows. Section 2.2 presents the problem formulation. Section 2.3 introduces two approaches for creating spatiotemporal profiles. Section 2.4 presents two protocols for privacy-preserving spatiotemporal matching. Section 2.5 theoretically analyzes the proposed protocols. Section 2.6 evaluates the proposed protocols by detailed numerical and experimental results. Section 2.7 surveys the related work. Section 2.8 summarizes this chapter.

2.2 Problem Formulation

2.2.1 Problem Statement

We consider a large geographic region such as the NYC metropolitan area with system users as either permanent residents or temporary visitors. Each user carries at least one mobile device which has a WiFi/Bluetooth interface and can acquire his realtime position via on-device positioning software. Such assumptions on device capabilities are fairly justifiable on most current and future mobile devices for D2D communications. Besides, unlike traditional communications between mobile users and the service provider [166], mobile users want to performance secure D2D communications via the WiFi/Bluetooth interfaces on their mobile devices. In addition, time is divided into equal-length *epoch*s, each represented by a globally unique epoch index of l_{epoch} bits. We also postulate that each mobile device, which may traverse different time zones, can always convert its local time into the corresponding epoch index.

Each user u's *spatiotemporal profile* is defined as a set of 2-tuples $(i, loc_{u,i})$, where i and $loc_{u,i}$ denote the epoch index and the corresponding location index, respectively. In our protocol, $loc_{u,i}$ comprises some physical locations closely approximating the user's whereabouts in epoch i. The detailed construction of spatiotemporal profiles is postponed to Section 2.4.

We use Alice and Bob as two exemplary mobile users throughout the chapter. Let $\mathcal{P}_A = \{(i, loc_{A,i})\}_{i>0}^{\infty}$ and $\mathcal{P}_B = \{(i, loc_{B,i})\}_{i>0}^{\infty}$ denote the spatiotemporal profiles of Alice and Bob, respectively. We also let $\mathcal{P}_{A,\alpha\to\beta}$ and $\mathcal{P}_{B,\alpha\to\beta}$ denote their respective spatiotemporal profiles from epochs α to β. Assume that Alice is the initiator of a D2D communication session and that Bob is one of the candidate D2D partners in Alice's proximity. Alice wants to select a trustworthy D2D partner and needs to

conduct spatiotemporal matching with every candidate partner. Consider Bob as an example. Alice and Bob need to compare their spatiotemporal profiles from epochs α to β, where α and β are chosen by Alice herself. A complete matching process involves each of them initiating an independent protocol instance. The number of encounters with Bob in Alice's eye in any epoch $i \in [\alpha, \beta]$ equals the number of common locations in their location indexes in epoch i, and the number of encounters with Bob from epochs α to β in her eye equals the sum of total encounters in every epoch from α to β. In the similar fashion, we can define the total number of encounters with Alice from Bob's viewpoint from epochs α to β. We proceed to introduce the following definition.

Definition 2.2.1. *(**Spatiotemporal Match**) After protocol execution, a* spatiotemporal match *between Alice and Bob from epochs α to β is said to occur if the total number of encounters with Bob exceeds τ_A from Alice's viewpoint, and the total number of encounters with Alice exceeds τ_B from Bob's viewpoint, where τ_A and τ_B are personal thresholds independently chosen by Alice and Bob, respectively.*

We assume that Alice and Bob both desire strong spatiotemporal privacy and collaborate only when a spatiotemporal match occurs between them. Our focus is to devise an efficient protocol ensuring that as little information as possible about the spatiotemporal profile of either Alice or Bob is disclosed beyond the matching result. One may think about letting them directly exchange and compare their spatiotemporal profiles under pseudonyms instead of real names so that a known spatiotemporal profile cannot be directly linked to a real identity. Unfortunately, the knowledge of a pseudo-identity's spatiotemporal profile may be disastrous enough, e.g., leading to physical chasing to unveil the corresponding real identity. We thus need a sound solution regardless of pseudonyms.

2.2.2 *Adversary Model*

We assume a honest-but-curious adversary model commonly adopted to study privacy-preserving profile matching [21, 84, 167] or proximity test [93, 114, 137]. With Alice and Bob as an example, they both honestly follow the spatiotemporal matching protocol while having great curiosity about the other's spatiotemporal profile.

We do not consider *continuous fake-profile* attacks and *Denial-of-Service* (DoS) attacks in this chapter. In the former, either matching participant keeps using fake spatiotemporal profiles possibly under different pseudonyms in order to accumulate more information about the other party's spatiotemporal profile as time goes by, while in the latter, an attacker aims at depleting the resources of the other party in the same way. The only feasible countermeasure against both attacks in our opinion is for every party to rate-limit the total number of matching requests he/she will accept. Further investigation on these attacks is beyond the scope of this chapter.

There might also be external eavesdroppers or physical chasers. The former overhear the messages incurred by a spatiotemporal matching instance and can be easily thwarted by letting the matching participants encrypt the protocol messages. The latter tail a victim user and thus can always have a spatiotemporal profile resembling that of the victim user. There is no sound technical solution to such chasing attacks.

2.3 Spatiotemporal Profile Construction

In this section, we introduce two approaches for constructing spatiotemporal profile, including a *passive* approach and an *active* approach. In the passive approach, each user records his own spatiotemporal information periodically whereby to construct his spatiotemporal profile. In the active approach, each mobile user continuously broadcasts epoch-specific cryptographic token at an adaptive frequency and also

records every token he overhears via WiFi/Bluetooth interface. The spatiotemporal profile of each mobile user is then constructed from the sent and received tokens.

2.3.1 A Passive Approach

The passive approach explores the prevalent capability of mobile devices obtaining their physical locations via hybrid GPS, WiFi, and cellular positioning techniques. Assume that each epoch is evenly divided into λ intervals, where $\lambda \geq 1$ is a global parameter. In general, each user passively records his location in the middle of each interval to tolerate synchronization errors among mobile devices. Recall that any user u's spatiotemporal profile is defined in Section 2.2.1 as a set of 2-tuples like $(i, loc_{u,i})$. We have $loc_{u,i} = \{p_{u,i}[j]\}_{j=1}^{\lambda}$, where $p_{u,i}[j]$ denotes user u's jth location in epoch i. Consider the exemplary users Alice and Bob with profiles $\mathcal{P}_A = \{i, \{p_{A,i}[j]\}_{j=1}^{\lambda}\}_{i=1}^{\infty}$ and $\mathcal{P}_B = \{i, \{p_{B,i}[j]\}_{j=1}^{\lambda}\}_{i=1}^{\infty}$, respectively. Now they attempt to compare their profiles from epochs α to β, i.e., $\{i, \{p_{A,i}[j]\}_{j=1}^{\lambda}\}_{i=\alpha}^{\beta}$ and $\{i, \{p_{B,i}[j]\}_{j=1}^{\lambda}\}_{i=\alpha}^{\beta}$, equivalent to the comparison of $\lambda(\beta - \alpha + 1)$ location pairs.

We further assume that each physical region of interest (like a metropolitan area) can be approximated by a square called a *level-1* cell. Then we divide the level-1 cell into four equally-sized squares called *level-2* cells, each of which is further divided into four equally-sized squares named as level-3 cells. This process continues until reaching level-θ cells, each having a side length no larger than a desired threshold, and how to determine the cell-division threshold will be discussed later. Note that there are totally 4^{j-1} level-j cells for $\forall j \in [1, \theta]$. Then we assign a unique cell index to the cell(s) on every level. In particular, the index of the level-1 cell is 0, and the indexes of the upper-left, lower-left, upper-right, and lower-right level-2 cells are 00, 01, 02, and 03, respectively. The same indexing rule can be applied to the cells on all levels. The region-division rules are public information and can be downloaded as

needed. In practice, each user just needs to have the rules related to the regions he commonly stays in or travel to, so the related storage overhead is negligible.

To facilitate customized spatiotemporal matching, we propose an *adaptive quantization* technique which works by letting each user convert his locations into cell indexes. In particular, assume that Alice and Bob negotiate a common region of interest on which to conduct spatiotemporal matching. Since each region corresponds to a large geographic area, disclosing the regions of interest to each other may not be a serious concern in practice; otherwise, Alice and Bob can apply Private Set Intersection (PSI) [46] to negotiate the common region, which will be very efficient given the limited possible regions. In addition, they agree on a cell level $\xi \in [1, \theta]$ on which the quantization takes place, and the impact of ξ will be discussed shortly. Then Alice converts $\{i, \{p_{A,i}[j]\}_{j=1}^{\lambda}\}_{i=\alpha}^{\beta}$ into $\overline{\mathcal{P}}_{A,\alpha\to\beta} = \{\{\langle i, j, \bar{p}_{A,i}[j]\rangle\}_{j=1}^{\lambda}\}_{i=\alpha}^{\beta}$, where $\bar{p}_{A,i}$ denotes the index of the level-ξ cell that contains $p_{A,i}$. If a certain location is not in the negotiated region, the corresponding cell index is set to some randomly chosen unlikely cell index indicating this abnormality. Similarly, Bob can convert his profile $\{i, \{p_{B,i}[j]\}_{j=1}^{\lambda}\}_{i=\alpha}^{\beta}$ into $\overline{\mathcal{P}}_{B,\alpha\to\beta} = \{\{\langle i, j, \bar{p}_{B,i}[j]\rangle\}_{j=1}^{\lambda}\}_{i=\alpha}^{\beta}$. With adaptive quantization in place, the number of encounters between Alice and Bob equals the number of level-ξ cells they both came across in the same epoch interval, or equivalently the intersection cardinality $|\overline{\mathcal{P}}_{A,\alpha\to\beta} \bigcap \overline{\mathcal{P}}_{B,\alpha\to\beta}|$.

2.3.2 An Active Approach

In the active approach, each mobile user continuously broadcasts an epoch-specific cryptographic token at an adaptive frequency and also records every token he overhears via WiFi-direct, Bluetooth, Frequency Hopping, or other available Device-to-Device (D2D) technologies widely used in many applications [135, 137, 156, 165]. For example, the tokens can be exchanged via WiFi/Bluetooth interfaces without

requiring the involved parties to explicitly establish any WiFi/Bluetooth connection [156].

Assume that every user u has a unique identifier ID_u and also a secret key k_u. Let $H(\cdot)$ denote any good cryptographic hash function. The token he broadcasts in epoch i is computed as $t_{u,i} = H(k_u, i, ID_u)$ truncated to a given length. User u needs to broadcast $\text{token}_{u,i}$ at a personally-chosen frequency to make sure that it can be overheard by sufficient users he encounters, and how to determine this token frequency will be discussed shortly. In addition, user u should use a different pseudonym in every epoch for broadcasting tokens; otherwise, a powerful adversary would be able to associate the tokens he sends in different epochs with him, thus breaching his location privacy.

User u also receives tokens from other users through his WiFi and/or Bluetooth interfaces and only records any token once that he may receive multiple times. Let $\mathcal{R}_{u,i} = \{r_{u,i,j}\}_{j=1}^{n_{u,i}}$ denote the set of $n_{u,i}$ tokens user u receives from others he encounters in epoch i. Any token in $\mathcal{R}_{u,i}$ can serve as the proof that user u was in the WiFi or Bluetooth transmission range of the token sender. User u's whereabouts in epoch i can thus be implicitly determined by his physical proximity to other mobile users from which he has received tokens.

We define two types of spatiotemporal profiles for the active approach, including *initiator profile* and *receiver profile*. Recall that user u's spatiotemporal profile is defined in Section 2.2.1 as a set of 2-tuples $(i, loc_{u,i})$. The initiator and receiver profiles of user u are defined as $\mathcal{I}_u = \{(i, t_{u,i})\}_{i=1}^{\infty}$ and $\mathcal{R}_u = \{\{(i, r_{u,i,j})\}_{j=1}^{n_{u,i}}\}_{i=1}^{\infty}$.

Continue the example of Alice and Bob. An encounter with Bob (or Alice) occurs in epoch i from Alice's (or Bob's) viewpoint if $t_{A,i} \in \mathcal{R}_{B,i}$ (or $t_{B,i} \in \mathcal{R}_{A,i}$). Suppose they attempt to compare their profiles from epochs α to β to determine the number of their encounters. Let $m_{A,\alpha\to\beta}$ and $m_{B,\alpha\to\beta}$ denote the number of encounters with

Bob in Alice's view and with Alice in Bob's view, respectively. We have $m_{A,\alpha\to\beta} = |\mathcal{I}_{A,\alpha\to\beta} \bigcap \mathcal{R}_{B,\alpha\to\beta}|$ and $m_{B,\alpha\to\beta} = |\mathcal{I}_{B,\alpha\to\beta} \bigcap \mathcal{R}_{A,\alpha\to\beta}|$, where $\mathcal{I}_{u,\alpha\to\beta} = \{(i, t_{u,i})\}_{i=\alpha}^{\beta}$ and $\mathcal{R}_{u,\alpha\to\beta} = \{\{(i, r_{u,i,j})\}_{j=1}^{n_{u,i}}\}_{i=\alpha}^{\beta}$ for $u = A$ or B.

2.3.3 Discussion

We now discuss some factors that may affect the spatiotemporal profile construction and thus the spatiotemporal matching result. In particular, the passive approach may be affected by the following three factors.

- *Recording frequency*: Each user records his location in the middle of each interval in each epoch of fixed length. The fewer intervals in each epoch, the lower the recording frequency, and the more likely for *false negatives* to occur, in which case a protocol initiator considers the responder a mismatch who actually encountered him multiple times and just did not record the encounter locations due to the low recording frequency. In contrast, the higher the recording frequency, the less likely for false negatives to occur, and the longer every location index in every epoch which will lead to larger computation and communication overhead.

- *Quantization granularity*: The granularity of spatiotemporal matching can be controlled by choosing a proper quantization level $\xi \in [1, \theta]$. A larger ξ can lead to finer-grained matching at the sacrifice of spatiotemporal privacy and matching efficiency, while a smaller ξ can lead to better spatiotemporal privacy at the cost of coarser-grained matching and longer spatiotemporal matching time.

- *Imperfect quantization*: Our quantization process may cause some ambiguity. For example, if the recorded locations of Alice and Bob in the same interval are

near the upper-left and lower-right corners of the same level-ξ cell, they will be quantized to the same level-ξ index and thus translated into one encounter. In contrast, if the two locations are in adjacent level-ξ cells and close to each other along the cell boundary, they, however, will be quantized to different level-ξ indexes and translated into a non-encounter.

Similarly, the active approach may be affected by the following two factors.

- *Token broadcasting frequency*: The more frequently a user broadcasts an epoch-specific token, the more users he encounters can receive the token, and the less likely for *false negative*s to occur, in which case a protocol initiator deems the responder a mismatch who actually encountered him many times and just did not receive sufficient tokens from him in the matching epochs due to channel errors, missing the time points for token transmissions, etc. In contrast, the less frequently a token is broadcasted in one epoch, the less energy the user consumes at the cost of higher false-negative rates. The user can adopt an adaptive method by letting a user dynamically adjust his broadcasting frequency proportional to his moving speed which can be readily inferred based on the accelerometer increasingly available on mobile devices. The intuition is that the users encountered by a high-speed (or low-speed) user may quickly (slowly) move out of his WiFi/Bluetooth transmission range, so he can increase (or decrease) the token frequency accordingly.

- *Uniqueness of each user's broadcasted tokens*: The correctness of our protocols depends on $\{t_{A,i}\}_{i=\alpha}^{\beta}$ (or $\{t_{B,i}\}_{i=\alpha}^{\beta}$) being all unique in our previous example. Recall that the token any user u (i.e., $t_{u,i}$) broadcasts in epoch i equals $H(k_u, i, ID_u)$ truncated to a given length. Due to the randomness of the hash output, it is likely that the tokens user u sent in adjacent epochs might be the

same. A simple remedy is to let user u keep a FIFO queue of size equal to the longest matching epoch-interval he may be interested in. The queue records all the recently used tokens. Consider epoch i as an example. If the truncated $H(k_u, i, ID_u)$ is in the queue, user u tries $H(k_u, i, ID_u, 1)$, $H(k_u, i, ID_u, 2)$, ..., until finding a token not in the queue, which will be used as $t_{u,i}$ and inserted into the queue.

2.4 Privacy-Preserving Spatiotemporal Matching

In this section, we present two novel privacy-preserving spatiotemporal matching protocols.

From the discussion of Section 2.3, we can see that the problem of privacy-preserving spatiotemporal matching boils down to the problem of enabling two users (e.g., Alice and Bob) to learn the cardinality of the intersection of their spatiotemporal profiles represented by two sets Ψ_A and Ψ_B, respectively, while disclosing as little additional information as possible beyond the matching result. In particular, if the passive approach is adopted to construct the spatiotemporal profile, we have $\Psi_A = \overline{\mathcal{P}}_{A,\alpha\to\beta}$ and $\Psi_B = \overline{\mathcal{P}}_{B,\alpha\to\beta}$. Similarly, under the active approach, we have $\Psi_A = \mathcal{I}_{A,\alpha\to\beta}$ and $\Psi_B = \mathcal{R}_{B,\alpha\to\beta}$ if Alice's point of view is considered, and $\Psi_B = \mathcal{I}_{B,\alpha\to\beta}$ and $\Psi_A = \mathcal{R}_{A,\alpha\to\beta}$ if Bob's point of view is considered.

2.4.1 A Bloom-filter-based Privacy-Preserving Spatiotemporal Matching Protocol

Our first spatiotemporal matching protocol is motivated by the observation that an accurate estimation of the number of encounters may suffice in practice and involves a novel use of the Bloom filter [31].

A Bloom filter [31] is a space-efficient probabilistic data structure [33, 137, 169] for set-membership testing. Assume that a w-bit Bloom filter is used for a data set

$\{s_i\}_{i=1}^{d}$, which has every bit initialized to 0. Let $\{h_a(\cdot)\}_{a=1}^{k}$ denote k different hash functions, each with output in $[1, w]$. Every element s_i is added into the Bloom filter by setting all bits at positions $\{h_a(s_i)\}_{a=1}^{k}$ to 1. To check the membership of an arbitrary element e in the given data set, we can simply verify whether all the bits at positions $\{h_a(e)\}_{a=1}^{k}$ have been set. If not, e is certainly not in the data set; otherwise, it is in the data set with some probability jointly determined by d, w, and k.

Our protocol involves Alice and Bob each using a different set of hash functions to construct a Bloom filter based on his/her spatiotemporal profile. In particular, let $\mathcal{H}$ denote a large and public pool of hash functions with each indexed by a unique identifier. Assume that Alice and Bob are to find out $|\Psi_A \cap \Psi_B|$. Without loss of generality, let $\Psi_A = \{a_1, \ldots, a_{n_A}\}$ and $\Psi_B = \{b_1, \ldots, b_{n_B}\}$, where $n_A = n_B$ if the passive approach is adopted and $n_A \neq n_B$ otherwise. The following operations are done in sequence for Alice to obtain an estimated $\hat{m}_A$ about $m_A = |\Psi_A \cap \Psi_B|$, where m_A represents the number of encounters with Bob in Alice's view.

1. Alice sends a spatiotemporal matching request with n_A to Bob.

2. If $n_A > n_B$, Bob adds $n_A - n_B$ dummy elements that are definitely not in Ψ_A to obtain his new spatiotemporal profile Ψ'_B. Bob then randomly chooses k hash functions from $\mathcal{H}$ with indexes denoted by $\mathcal{H}_B$ and then inserts each element in his profile Ψ'_B into a w-bit Bloom filter (denoted by BF_B) with different $l < k$ functions randomly selected from $\mathcal{H}_B$ and $k - l$ random hash functions outside $\mathcal{H}$. Finally, Bob returns n_B, $\mathcal{H}_B$, and BF_B to Alice.

3. If $n_B > n_A$, Alice adds $n_B - n_A$ dummy elements that are definitely not in Ψ_B to obtain his new spatiotemporal profile Ψ'_A.

4. Alice constructs a w-bit Bloom filter (denoted by BF_A) based on the hash functions specified in $\mathcal{H}_B$ and her profile Ψ'_A. Then she counts the number of

common bit-0 positions in BF_A and BF_B (denoted by n_0) whereby to compute

$$\hat{m}_A = \frac{2kn - w(\ln w - \ln n_0)}{l} \,, \tag{2.1}$$

where $n = \max(n_A, n_B)$. The correctness and accuracy of this estimation will be analyzed in Section 2.5.2.

Likewise, Bob can initiate a spatiotemporal matching process to estimate the number of encounters with Alice $\hat{m}_B$ from his point of view. Finally, they can jointly determine whether there is a successful spatiotemporal matching after independently comparing $\hat{m}_A$ (or $\hat{m}_B$) with the personal threshold τ_A (or τ_B).

We have some important remarks to make. First, since Alice and Bob use some common hash functions in $\mathcal{H}_B$ to construct their respective Bloom filter, the same elements in their spatiotemporal profiles (if any) are likely to set the same bit positions. So we can estimate the number of common elements via the number of common bit-0 and/or bit-1 positions. Second, the reason for Bob using $k - l$ random hash functions unknown to Alice for each element is to prevent Alice from estimating Bob's spatiotemporal presence by simple Bloom set-membership tests. In particular, if Bob uses the same k hash functions in $\mathcal{H}_B$ to generate BF_B, Alice can easily test whether some possible element is in BF_B, which is equivalent to breaching Bob's spatiotemporal privacy. This set-membership test is less critical to the active approach because the adversary does not know the user's secret keys and can only randomly guess the broadcasted tokens. However, it is critical to the passive approach in which all the possible pairs of epoch and cell indexes are known to the adversary as well. The choice of k and l will be detailed in Section 2.5.2. Finally, the construction of many different hash functions for implementing the Bloom filter is also very important. One common method is to seed a cryptographic hash function such as SHA-2 with the

indexes of hash functions we want. There are also some more efficient realizations of many hash functions specifically for the Bloom filter [48, 74].

2.4.2 A Weighted Privacy-Preserving Spatiotemporal Matching Protocol

We now generalize the above protocol to support weighted privacy-preserving spatiotemporal matching, which is defined as follows.

Definition 2.4.1. *(**Weighted Spatiotemporal Match**) Assume that Alice and Bob each assign different weights for encounter at different locations and times. A weighted spatiotemporal match between Alice and Bob is said to occur if the weighted sum of encounters with Bob exceeds τ_A from Alice's viewpoint, and the weighted sum of encounters with Alice exceeds τ_B from Bob's viewpoint, where τ_A and τ_B are personal thresholds independently chosen by Alice and Bob, respectively.*

More specifically, consider Alice and Bob with spatiotemporal profiles $\Psi_A = \{a_1, \ldots, a_{n_A}\}$ and $\Psi_B = \{b_1, \ldots, b_{n_B}\}$, respectively. Assume that Alice assigns a weight $w_{A,i}$ for possible encounter corresponding to element a_i in Ψ_A for each $i \in [1, n_A]$, and that Bob assigns a weight $w_{B,j}$ for possible encounter corresponding to element b_j in Ψ_B for each $j \in [1, n_B]$. The weighted count of encounters with Bob from Alice's viewpoint is computed as

$$m_A = \sum_{i=1}^{n_A} c_i \tag{2.2}$$

where

$$c_i = \begin{cases} w_{A,i} & \text{if } a_i \in \Psi_B, \\ 0 & \text{otherwise,} \end{cases}$$

and the weighted count of encounters with Alice from Bob's viewpoint can be computed accordingly.

We observe that weighted spatiotemporal matching can be converted into spatiotemporal matching between two spatiotemporal profiles constructed from weight sets. Specifically, assume that that $w_{A,i} \in \{1, \ldots, \mathsf{w}\}$ for all $i \in [1, n_A]$, where w is a publicly known parameter. Alice can construct a new spatiotemporal profile Ψ'_A from her original profiles Ψ_A and weight set $\mathcal{W}_A = \{w_{a,i}\}_{i=1}^{n_A}$ as follow. For each element $a_i \in \Psi_A$ with weight assignment $w_{A,i}$, Alice converts a_i into $w_{A,i}$ different elements $a_i||1$, $a_i||2$, ..., $a_i||w_{A,i}$. As a result, Alice obtains her new spatiotemporal profile $\Psi'_A = \{\{a_i||j\}_{j=1}^{w_{A,i}}\}_{i=1}^{n_A}$. On the other hand, Bob can construct a new spatiotemporal profile Ψ_B in a different way. For each element $b_i \in \Psi_B$ with weight $w_{B,i} \in \mathcal{W}_B$, Bob converts b_i into w different elements $b_i||1$, $b_i||2$, ..., $b_i||\mathsf{w}$ to obtain a new spatiotemporal profile $\Psi'_B = \{\{b_i||j\}_{j=1}^{\mathsf{w}}\}_{i=1}^{n_B}$. It follows that

$$m_A = |\Psi'_A \cap \Psi'_B| \ .$$

Assume Alice and Bob have their respective spatiotemporal profiles Ψ_A and Ψ_B via either the passive or active approaches. The following operations are done in sequence to allow Alice to obtain an estimated $\hat{m}_A$ about $m_A = |\Psi'_A \cap \Psi'_B|$.

1. Alice creates a new spatiotemporal profile $\Psi'_A = \{\{a_i||j\}_{j=1}^{w_{A,i}}\}_{i=1}^{n_A}$.

2. Alice sends a weighted spatiotemporal matching request with $w_A = \sum_{i=1}^{n_A} w_{A,i}$ to Bob.

3. Bob creates a new spatiotemporal profile $\Psi'_B = \{\{b_i||j\}_{j=1}^{\mathsf{w}}\}_{i=1}^{n_B}$, and calculates $w_B = n_B\mathsf{w}$.

4. If $w_A > w_B$, Bob adds $w_A - w_B$ dummy elements that are definitely not in Ψ'_A to Ψ'_B to obtain his new spatiotemporal profile Ψ''_B. Bob then randomly chooses k hash functions from $\mathcal{H}$ with indexes denoted by $\mathcal{H}_B$ and then inserts each

element in his profile Ψ''_B into a w-bit Bloom filter BF_B with different $l < k$ functions randomly selected from $\mathcal{H}_B$ and $k - l$ random hash functions outside $\mathcal{H}$. Finally, Bob returns w_B, $\mathcal{H}_B$, and BF_B to Alice.

5. If $w_B > w_A$, Alice adds $w_B - w_A$ dummy elements that are definitely not in Ψ'_B to Ψ'_A to obtain her new spatiotemporal profile Ψ''_A.

6. Alice constructs a w-bit Bloom filter BF_A using the hash functions specified in $\mathcal{H}_B$ and her profile Ψ''_A. Then she counts the number of common bit-0 positions in BF_A and BF_B (denoted by n_0) whereby to compute

$$\hat{m}_A = \frac{2kn - w(\ln w - \ln n_0)}{l} , \tag{2.3}$$

where $n = \max(w_A, w_B)$.

2.5 Performance Analysis

In this section, we analyze the performance of the proposed protocols.

2.5.1 Performance Metrics

We use the following metrics to evaluate our protocols.

Accuracy

The following standard (ϵ, δ) guarantee is used to measure the accuracy of the protocol output,

$$Pr[(1 - \epsilon)m \leq \hat{m} \leq (1 + \epsilon)m] > 1 - \delta, \tag{2.4}$$

where m is the actual number of common elements (or encounters) in , and $\hat{m}$ is the estimation of m output by a spatiotemporal matching protocol.

Privacy

We quantify spatiotemporal privacy by the Shannon entropy, a commonly used measure of uncertainty.

We take Bob as an example to analyze the his spatiotemporal privacy under the passive approach. Recall that Bob's quantized spatiotemporal profile from epochs α to β is $\Psi_B = \overline{\mathcal{P}}_{B,\alpha\to\beta} = \{\{i, j, \bar{p}_{B,i}[j]\}_{j=1}^{\lambda}\}_{i=\alpha}^{\beta}$, where $\bar{p}_{B,i}$ denotes a level-ξ cell index. The only information Alice knows about $\overline{\mathcal{P}}_{B,\alpha\to\beta}$ before protocol execution includes the parameters α, β, and λ. Since there are total $N = 4^{\xi-1}$ level-ξ cell indexes, each of them is equally likely to be $\bar{p}_{B,i}[j]$ from Alice's viewpoint. There are thus total $N^{\lambda(\beta-\alpha+1)}$ candidate quantized profiles for $\overline{\mathcal{P}}_{B,\alpha\to\beta}$ with equal probability from Alice's viewpoint. So the maximum spatiotemporal privacy of Bob with regard to Alice (i.e., the maximum uncertainty of his spatiotemporal profile to Alice) in bits can be computed as

$$\mathbf{E}^* = \log_2 N^{\lambda(\beta-\alpha+1)} = 2\lambda(\beta-\alpha+1)(\xi-1). \tag{2.5}$$

To make the analysis of the spatiotemporal privacy of Bob under the active approach tractable and comparable with the passive approach, we make the following assumptions. We assume that during each epoch, Alice and Bob each wander in one level-ξ cell as in the passive approach and that Alice keeps broadcasting a unique token at sufficiently high frequency such that Bob always receives Alice's token if they are in the same cell. In addition, we ignore the case in which Bob receives Alice's token while they are in two different cells, e.g., they are close two the boundary of two adjacent cells. Similar to the analysis of the passive approach, since there are total $N = 4^{\xi-1}$ level-ξ cells, each of them is equally likely to be the cell Bob resides from Alice's viewpoint. There are total N^{n_B} candidate quantized profiles with equal probability from Alice's viewpoint. So the maximum spatiotemporal privacy of Bob

with regard to Alice in bits (i.e., the maximum uncertainty of his spatiotemporal profile to Alice) can be computed as

$$\mathbf{E}^* = \log_2 N^{n_B} = 2n_B(\xi - 1). \tag{2.6}$$

After the execution of either protocol, Alice can know more information about the probability of each candidate profile being Bob's profile whereby to reduce the entropy or uncertainty, which we will analyze shortly. The maximum spatiotemporal privacy of Alice with regard to Bob can be analyzed in a similar fashion and thus omitted here.

Overhead

We will measure the communication and computation overhead of the spatiotemporal matching protocol using the number of hash computations and the number of bits transferred between two users during protocol execution, respectively.

2.5.2 Analysis of the Spatiotemporal Matching Protocol

Accuracy Analysis

We have the following theorem regarding the accuracy of the privacy-preserving spatiotemporal matching protocol.

Theorem 2.5.1. *Given the number of common bit-0 positions n_0 in the w-bit Bloom filters BF_A and BF_B constructed in the spatiotemporal matching protocol, Alice can estimate $|\Psi_A \bigcap \Psi_B|$ as*

$$\hat{m} = \frac{2nk - w(\ln w - \ln n_0)}{l}\ , \tag{2.7}$$

where $n = \max(n_A, n_B)$. Assuming that $\epsilon m \geq 1$, $\hat{m}$ is an (ϵ, δ) estimation of m if

$$\delta \geq \frac{w(e^{\frac{2nk}{w}} - (1 + \frac{2nk}{w}))}{l^2\epsilon^2 m^2}. \tag{2.8}$$

The proof of Theorem 2.5.1 is given as follows.

Proof. For each bit position of either Bloom filter, the probability that it is set to bit-1 by a common element with l common hash functions is given by

$$p = 1 - (1 - \frac{1}{w})^{ml} \approx 1 - e^{-\frac{ml}{w}}. \tag{2.9}$$

The probability that it is set to bit-1 in all the other cases is given by

$$q = 1 - (1 - \frac{1}{w})^{nk-ml} \approx 1 - e^{-\frac{nk-ml}{w}}. \tag{2.10}$$

Therefore, the probability that a position is bit-0 in both BF_A and BF_B (i.e., common bit-0 position) is given by

$$P_0 = (1-p)(1-q)^2 = e^{-\frac{ml}{w}} e^{-\frac{2(nk-ml)}{w}}. \tag{2.11}$$

Since Alice can count the number of common bit-0 positions n_0 in BF_A and BF_B, the following equation can be established

$$P_0 = e^{-\frac{ml}{w}} e^{-\frac{2(nk-ml)}{w}} = \frac{n_0}{w}\ . \tag{2.12}$$

Solving this equation, we have

$$\hat{m} = \frac{2nk - w(\ln w - \ln n_0)}{l}\ . \tag{2.13}$$

Next, we derive the variance. We cast the problem into RFID tag estimation and refer to the results in [77]. The RFID system with t tags divides a time period into f slots and let each RFID tag randomly select one of f slots to respond. One slot may be responded by zero, one, or multiple tags. The expected number of zero-response slots is nearly $fe^{-t/f}$. Knowing the number of zero-response slots, the system administrator can estimate the number of present RFID tags. Our estimation method based on the Bloom filter is similar to RFID tag estimation if we consider common bit-1 positions

and common bit-0 positions as multiple-response and zero-response slots in the RFID system, respectively. The expected number of common bit-0 positions of BF_A and BF_B is nearly $we^{-(2nk-ml)/w}$. Knowing the number of common bit-0 positions, we can estimate the intersection size m.

Let $\rho = \frac{2nk-ml}{w}$. According to Theorem 1 in [77], we have $n_0 \sim \mathcal{N}(\mu, \sigma^2)$, where

$$\mu = w(1 - \frac{1}{w})^{2nk-ml} = we^{-\rho} , \tag{2.14}$$

$$\sigma^2 = we^{-\rho}(1 - (1+\rho)e^{-\rho}) . \tag{2.15}$$

We can view μ as a function of the true number of common elements, denoted by $\mu(m)$. Since $\mu(m)$ is monotonic continuous functions of m, it has a unique inverse, denoted by $g()$, i.e., $g(\mu(m)) = m$. Let $2nk-ml \to \infty$ and $w \to \infty$, while maintaining $\frac{2nk-ml}{w} = \rho$. Since $g(\mu(m)) = m$, differentiating this equation with respect to m, we get $g'(\mu(m))\mu'(m) = 1$. it follows that $g'(\mu(m)) = \frac{1}{\mu'(m)}$. According to Theorem 6 in [77], the variance of common bit-0 estimation of m is given by

$$\delta_0 = \sigma^2(m)[g'(\mu(m))]^2 = \frac{\sigma^2(m)}{[\mu'(m)]^2} . \tag{2.16}$$

Since $\mu = we^{-\frac{2nk-ml}{w}}$ and $\sigma^2 = we^{-\rho}(1-(1+\rho)e^{-\rho})$. Differentiating $\mu(m)$ with respect to m, we can obtain $\frac{d\mu(m)}{dm} = le^{-\rho}$. Therefore we have

$$\delta_0 = \frac{we^{-\rho}(1-(1+\rho)e^{-\rho})}{l^2e^{-2\rho}} = \frac{w(e^{\rho}-(1+\rho))}{l^2}. \tag{2.17}$$

In addition, since $\frac{d\delta_0}{d\rho} = \frac{w}{l^2}(e^{\rho}-1) > 0$, we know that δ_0 is monotonic increasing with ρ. Since $0 \le m \le n$, we have $\frac{n(2k-l)}{w} \le \rho \le \frac{2nk}{w}$. Therefore when $\rho = \frac{2nk}{w}$, we have

$$\delta_{0\max} = \frac{w(e^{\frac{2nk}{w}} - (1+\frac{2nk}{w}))}{l^2} . \tag{2.18}$$

We thus have $\hat{m} \sim \mathcal{N}(m, \delta_0)$. According to the Chebyshev's inequality, we have

$$Pr(|\hat{m} - m| \le \epsilon m) \ge 1 - \frac{\delta_0}{\epsilon^2 m^2} \ge 1 - \delta . \tag{2.19}$$

Therefore, $\hat{m}$ is an (ϵ, δ) estimation of m if

$$\delta \geq \frac{\delta_{0\max}}{\epsilon^2 m^2} = \frac{w(e^{\frac{2nk}{w}} - (1 + \frac{2nk}{w}))}{l^2 \epsilon^2 m^2}. \tag{2.20}$$

□

Privacy Analysis

For the passive approach, the privacy analysis of the spatiotemporal matching protocol is given by the following theorem.

Theorem 2.5.2. *Assuming that Bob constructs a w-bit Bloom filter BF_B from his level-ξ quantized profile $\Psi_B = \{\{i, j, \bar{p}_{B,i}[j]\}_{j=1}^{\lambda}\}_{i=\alpha}^{\beta}$ using l functions from $\mathcal{H}_B$ and $k - l$ functions unknown to Alice. After transmitting BF_B and $\mathcal{H}_B$ to Alice, his remaining privacy of Ψ_B against Alice is given by*

$$\mathbf{E} = \lambda(\alpha + \beta - 1)\mathbf{E}[i, j] \ , \tag{2.21}$$

where

$$\begin{aligned} \mathbf{E}[i, j] &= \sum_{x=1}^{N} \binom{N}{x} P^x (1 - P)^{N-x} \log_2 x, \\ P &= \sum_{i=l}^{k} \binom{k}{i} p^i (1 - p)^{k-i} \ , \\ p &= 1 - e^{-\frac{\lambda(\alpha+\beta-1)k}{w}} \ . \end{aligned} \tag{2.22}$$

The proof of Theorem 2.5.2 is given as follows.

Proof. In the passive approach, since Alice and Bob's spatiotemporal profiles have the same size, we have $\Psi'_A = \Psi_A$ and $\Psi'_B = \Psi_B$. Bob's privacy disclosure is caused by transmitting BF_B and the indexes $\mathcal{H}_B$ of k hash functions to Alice. In particular, Alice can exploit BF_B and the knowledge that Bob inserts every element in $\overline{\mathcal{P}}_{B,\alpha\to\beta}$ using l random hash functions from $\mathcal{H}_B$ and $k - l$ unknown hash functions to deduce

some information about $\overline{\mathcal{P}}_{B,\alpha\to\beta}$. Consider an arbitrary element $\langle i, j, \bar{p}_{B,i}[j]\rangle$ as an example. For each of the N possible cell indexes, say cID, Alice can test whether it is a viable candidate for the unknown $\bar{p}_{B,i}[j]$ by using all the k hash functions in $\mathcal{H}_B$ to compute the k corresponding positions for the resulting element $\langle i, j, cID\rangle$. If there are at least l out of k corresponding positions set to bit-1 in BF_B, we have $cID = \bar{p}_{B,i}[j]$ with probability P; otherwise, we must have $cID \neq \bar{p}_{B,i}[j]$.

We now estimate P. After inserting all the $\lambda(\alpha+\beta-1)$ elements in $\overline{\mathcal{P}}_{B,\alpha\to\beta}$ into BF_B, the expected number of bit-1 positions is $w(1-(1-\frac{1}{w})^{\lambda(\alpha+\beta-1)k})$. For a random hash function applied to cID, the probability of the corresponding bit position having been set to bit-1 is

$$p = 1-(1-\frac{1}{w})^{\lambda(\alpha+\beta-1)k} \approx 1-e^{-\frac{\lambda(\alpha+\beta-1)k}{w}} \ . \tag{2.23}$$

The probability that at least l corresponding bit positions corresponding to cID have been set to bit-1 is then given by

$$P = \sum_{i=l}^{k} \binom{k}{i} p^i (1-p)^{k-i} \ . \tag{2.24}$$

Let $X_{i,j}$ denote the number of valid candidate cell indexes for $\bar{p}_{B,i}[j]$. The remaining entropy for interval i in epoch j is then $\log_2 X_{i,j}$. Since $X_{i,j}$ is randomly distributed in $[1, N]$ ($N = 4^{\xi-1}$), we have the mean remaining entropy for interval i in epoch j as

$$\begin{aligned} \mathbf{E}[i,j] &= \sum_{x=1}^{N} Pr(X_{i,j}=x)\log_2 x \\ &= \sum_{x=1}^{N} \binom{N}{x} P^x (1-P)^{N-x} \log_2 x \ . \end{aligned} \tag{2.25}$$

Assuming that the $\lambda(\beta-\alpha+1)$ intervals are independent from each other, the total remaining entropy is given by

$$\mathbf{E} = \sum_{i=\alpha}^{\beta}\sum_{j=1}^{\lambda} \mathbf{E}[i,j] = \lambda(\alpha+\beta-1)\mathbf{E}[i,j] \ . \tag{2.26}$$

□

The following theorem is about the privacy of the spatiotemporal matching protocol under the active approach.

Theorem 2.5.3. *Assuming that Bob constructs a w-bit Bloom filter BF_B from Ψ'_B using l functions from $\mathcal{H}_B$ and $k-l$ functions unknown to Alice. After transmitting BF_B and $\mathcal{H}_B$ to Alice, his remaining privacy of Ψ_B against Alice is*

$$\mathbf{E} = \sum_{x=0}^{n_A} \binom{n_A}{x} P^x (1-P)^{n_A - x} \log_2 N^{n_B - x} \ , \tag{2.27}$$

where Ψ_A and Ψ_B are the spatiotemporal profiles of Alice and Bob, respectively,

$$\begin{aligned} P &= \sum_{i=l}^{k} \binom{k}{i} p^i (1-p)^{k-i} \ , \\ p &= 1 - e^{-\frac{nk}{w}} \ . \end{aligned} \tag{2.28}$$

The proof of Theorem 2.5.3 is given as follows.

Proof. Assume that Alice and Bob conduct spatiotemporal profile matching with profiles $\Psi_A = \mathcal{I}_{A,\alpha\to\beta}$ and $\Psi_B = \mathcal{R}_{B,\alpha\to\beta}$, respectively. For every element in Alice's spatiotemporal profile, Alice can test whether it is a viable candidate in Bob's spatiotemporal profile by using all the k hash functions in $\mathcal{H}_B$ to compute the k corresponding positions for the resulting element. If there are at least l out of k corresponding positions set to bit-1 in BF_B, we have the conclusion that Bob and Alice were at the the same location at the same time with probability P.

Let $n = \max(n_A, n_B)$, where $n_A = |\Psi_A|$ and $n_B = |\Psi_B|$. Similar to Theorem 2.5.2, the probability that at least l corresponding bit positions have been set to bit-1 is then given by

$$P = \sum_{i=l}^{k} \binom{k}{i} p^i (1-p)^{k-i} \ , \tag{2.29}$$

where

$$p = 1 - (1 - \frac{1}{w})^{nk} \approx 1 - e^{-\frac{nk}{w}} \ . \tag{2.30}$$

Let X denote the number of tokens which might be in Bob's spatiotemporal profile. The remaining entropy for Bob's spatiotemporal profile is given by

$$\begin{aligned}\mathbf{E} &= \sum_{x=0}^{n_A} Pr(X=x)\log_2 N^{n_B-x} \\ &= \sum_{x=0}^{n_A} \binom{n_A}{x} P^x(1-P)^{n_A-x}\log_2 N^{n_B-x}\ .\end{aligned} \tag{2.31}$$

□

Overhead Analysis

The spatiotemporal matching protocol involves Alice and Bob each performing kn hash operations, where $n = \max(n_A, n_B)$, which is very efficient. The communication overhead mainly comes from the transmission of one Bloom filter and is of w bits.

2.5.3 Analysis of Weighted Spatiotemporal Matching Protocol

Accuracy Analysis

The accuracy of the weighted spatiotemporal matching protocol is guaranteed by the following theorem.

Theorem 2.5.4. *Given the number of common bit-0 positions n_0 in the w-bit Bloom filters BF_A and BF_B constructed from Ψ''_A and Ψ''_B, respectively, in the weighted spatiotemporal matching protocol, Alice can estimate the result of the weighted spatiotemporal matching as*

$$\hat{m} = \frac{2kn - w(\ln w - \ln n_0)}{l}\ , \tag{2.32}$$

where $n = \max(w_A, w_B)$. Assuming that $\epsilon m \geq 1$, $\hat{m}$ is an (ϵ, δ) estimation of m if

$$\delta \geq \frac{w(e^{\frac{2kn}{w}} - (1 + \frac{2kn}{w}))}{l^2\epsilon^2 m^2}. \tag{2.33}$$

The proof of Theorem 2.5.4 is similar to that of Theorem 2.5.1 and is thus omitted here.

Privacy Analysis

The privacy guarantee of weighted spatiotemporal matching protocol under the passive approach is given as follows.

Theorem 2.5.5. *Let* BF_B *denote a* w*-bit Bloom filter Bob constructs on his converted spatiotemporal profile* Ψ''_B *from epoch* α *to* β *using* l *functions from* $\mathcal{H}_B$ *and* $k-l$ *functions unknown to Alice. After transmitting* BF_B *and* $\mathcal{H}_B$ *to Alice, his remaining privacy of* Ψ''_B *against Alice is*

$$\mathbf{E} = \lambda(\alpha+\beta-1)\mathbf{E}[i,j] \ , \tag{2.34}$$

where

$$\begin{aligned} \mathbf{E}[i,j] &= \sum_{x=1}^{N} \binom{N}{x} P^{x\mathsf{w}}(1-P^{\mathsf{w}})^{N-x} \log_2 x, \\ P &= \sum_{i=l}^{k} \binom{k}{i} p^i (1-p)^{k-i} \ , \\ p &= 1 - e^{-\frac{nk}{w}} \ , \\ n &= \max(w_A, w_B) \ . \end{aligned} \tag{2.35}$$

The proof of Theorem 2.5.5 is given as follows.

Proof. Recall that Bob converts each of the elements in his profile to w new elements. For each of the N possible cell indexes, say cID, Alice wants to test whether its converted w elements $cID||1, cID||2, \ldots, cID||\mathsf{w}$ are in Bob's new profile Ψ''_B. Let $n = \max(w_A, w_B)$. For each element $cID||i, 1 \le i \le \mathsf{w}$, if there are at least l out of k corresponding positions set to bit-1 in BF_B, $cID||i$ is considered in Bob's new profile Ψ''_B with probability P, where

$$P = \sum_{i=l}^{k} \binom{k}{i} p^i (1-p)^{k-i} \ , \tag{2.36}$$

$$p = 1 - (1 - \frac{1}{w})^{nk} \approx 1 - e^{-\frac{nk}{w}} \ . \tag{2.37}$$

For any cID, it is considered in Bob's unconverted profile Ψ_B only if each of the w elements has at least l corresponding bit-1 positions, and the probability is P^{w}.

Let $X_{i,j}$ denote the number of candidate cell indexes. The remaining entropy for interval i in epoch j is then $\log_2 X_{i,j}$. Since $X_{i,j}$ is randomly distributed in $[1, N]$ ($N = 4^{\xi-1}$), we have the mean remaining entropy for interval i in epoch j as

$$\begin{aligned} \mathbf{E}[i,j] &= \sum_{x=1}^{N} Pr(X_{i,j} = x) \log_2 x \\ &= \sum_{x=1}^{N} \binom{N}{x} P^{x\mathsf{w}} (1 - P^{\mathsf{w}})^{N-x} \log_2 x \ . \end{aligned} \tag{2.38}$$

Assuming that the $\lambda(\beta - \alpha + 1)$ intervals are independent from each other, the total remaining entropy is given by

$$\mathbf{E} = \sum_{i=\alpha}^{\beta} \sum_{j=1}^{\lambda} \mathbf{E}[i,j] = \lambda(\alpha + \beta - 1)\mathbf{E}[i,j] \ . \tag{2.39}$$

□

The privacy guarantee of weighted spatiotemporal matching protocol under the active approach is given by the following theorem.

Theorem 2.5.6. *Let* BF_B *denote a* w*-bit Bloom filter Bob constructs on his converted spatiotemporal profile* Ψ''_B *using* l *functions from* $\mathcal{H}_B$ *and* $k - l$ *functions unknown to Alice. Assume we adopt level-*ξ *quantized After transmitting* BF_B *and* $\mathcal{H}_B$ *to Alice, his remaining privacy of* Ψ''_B *against Alice is*

$$\mathbf{E} = \sum_{x=0}^{n_A} \binom{n_A}{x} P^{x\mathsf{w}} (1 - P^{\mathsf{w}})^{n_A - x} \log_2 N^{n_B - x} \ , \tag{2.40}$$

where n_A and n_B are the sizes of spatiotemporal profiles of Alice and Bob before conversion, respectively,

$$\begin{aligned} P &= \sum_{i=l}^{k} \binom{k}{i} p^i (1-p)^{k-i} \ , \\ p &= 1 - e^{-\frac{nk}{w}} \ , \\ n &= \max(w_A, w_B) \ . \end{aligned} \tag{2.41}$$

The proof of Theorem 2.5.6 is given as follows.

Proof. Consider an arbitrary element in Alice's profile Ψ_A as an example. Alice can convert it to w elements as what Bob does. For each of the elements in Alice's profile Ψ_A, Alice wants to know whether it is a viable candidate in Bob's profile Ψ_B by testing whether each of its w converted elements results in at least l corresponding bit-1 positions. Let $n = \max(w_A, w_B)$. Similar to Theorem 2.5.5, the probability that each of the w elements has at least l corresponding bit-1 positions is P^{w}, where P is the probability that at least l corresponding bit positions have been set to bit-1 and is then given by

$$P = \sum_{i=l}^{k} \binom{k}{i} p^i (1-p)^{k-i} \ , \tag{2.42}$$

$$p = 1 - (1 - \frac{1}{w})^{nk} \approx 1 - e^{-\frac{nk}{w}} \ . \tag{2.43}$$

Let X denote the number of candidate elements in Bob's profile Ψ_B. The mean remaining entropy of Bob's profile is

$$\begin{aligned} \mathbf{E} &= \sum_{x=0}^{n_A} Pr(X = x) \log_2 N^{n_B - x} \\ &= \sum_{x=0}^{n_A} \binom{n_A}{x} P^{x\mathsf{w}} (1 - P^{\mathsf{w}})^{n_A - x} \log_2 N^{n_B - x} \ . \end{aligned} \tag{2.44}$$

□

Overhead Analysis

Similar to the spatiotemporal matching protocol, the weighted spatiotemporal matching protocol involves Alice and Bob each performing kn hash operations, where $n = \max(w_A, w_B)$. The communication overhead mainly comes from the transmission of one Bloom filter and is of w bits.

2.6 Performance Evaluation

In this section, we evaluate the two proposed protocols using simulations.

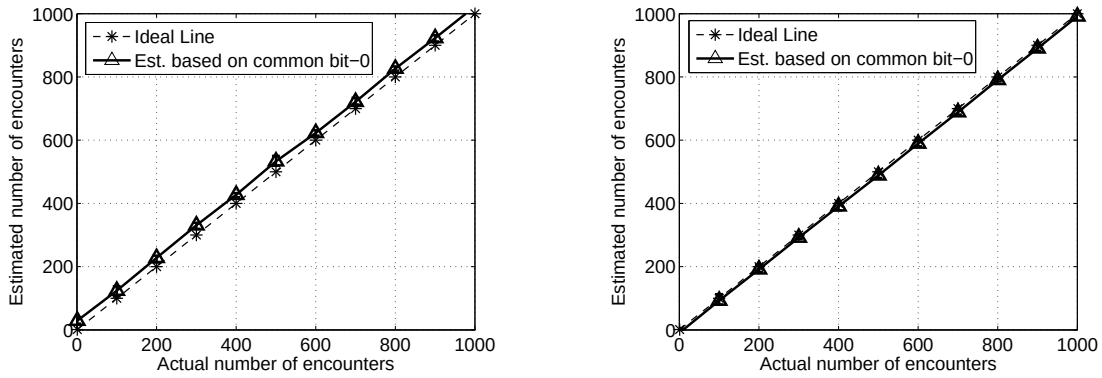
2.6.1 Simulation Settings

In the preliminary version of this chapter [139], we have shown that our protocol incurs significantly lower computation and communication overhead than traditional PSI-CA protocols [46, 56] based on computationally expensive public-key operations. Our simulation studies here will focus on the impact of various parameters on the accuracy and privacy of the spatiotemporal matching protocols.

We assume that the quantization is done on the level $\xi = 6$, i.e., $N = 4^{\xi-1} = 1024$. In addition, our experiments are on a Dell desktop with 2.67 GHz CPU, 9 GB RAM, and Windows 7 64-bit Professional, the evaluation program is written in Java, and every data point represents the average of 1000 runs. As discussed, a complete spatiotemporal matching involves Alice and Bob each initiating one protocol execution, but we only show the results for one protocol execution for simplicity. In addition, we set δ to 0.02, and ϵ is the relative error.

2.6.2 Simulation Results

Fig. 2.1a compares the estimated number of encounters $\hat{m}$ with the actual number of encounters m, when $k = 20$, $l = 16$, $n = 1000$, and $w = 40000$. We can see that



(a) Effect of n

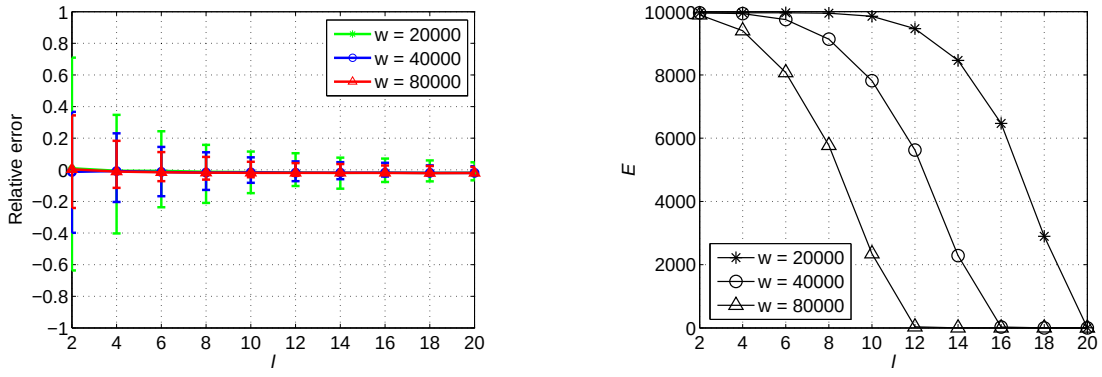
(b) Effect of $\hat{n}$

Figure 2.1: The Estimation Accuracy of the Advanced Protocol.

the estimator in Eq. (2.1) is always biased. The reason is that traditional analysis about the w-bit Bloom filter assumes that every bit position is set to bit-1 for any of n elements with equal probability $1/w$. In practice, however, the probability that one position is set to bit-1 is not independent of other positions: when one position is set to bit-1, it slightly reduces the probability that other positions are set to bit-1 [32, 33, 43]. Therefore, the actual number of bit-1 positions n_1 in the Bloom filter is a little smaller than that obtained via theoretical analysis, and the actual number of bit-0 positions n_0 in the Bloom filter is a little larger than that obtained via theoretical analysis. Since $\hat{m} = \frac{2kn - w(\ln w - \ln n_0)}{l}$, we can expect $\hat{m}$ to be larger than the true value m.

We resolve the biased estimation by letting $\hat{m} = \frac{2k\hat{n} - w(\ln w - \ln n_0)}{l}$, where $\hat{n} = \frac{\ln(n_{A0}/w)}{k\ln(1-1/w)}$, n_{A0} is the number of bit-0 positions in BF_A. Fig. 2.1b shows that this new estimator is almost unbiased and matches well with m. The reason is that using estimated number of elements $\hat{n}$ instead of the real number of elements $n = \lambda(\beta - \alpha + 1)$ takes into account the above difference between observed and theoretical numbers of

bit-0 and bit-1 positions. So we will use this modified estimator hereafter whose effectiveness will be further evidenced.



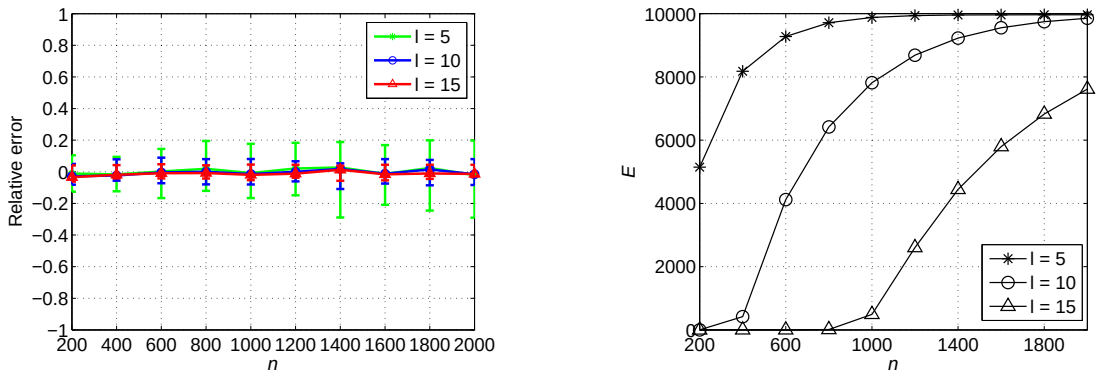
(a) Relative Error

(b) Privacy

Figure 2.2: The Impact of l, the Number of Common Hash Functions.

Fig. 2.2 shows the impact of l (the number of common hash functions Bob chooses to insert each of his elements) on the performance of advanced protocol, when $n = 1000$, $m = 500$, and $k = 20$. We can see from Fig. 2.2a that the more common hash functions (i.e., larger l), the smaller the variance of the relative error $|\hat{m}_A - m|/m$ (i.e., the more accurate the estimation). The reason is that the more common hash functions, the more common bit-0 positions in BF_A and BF_B, leading to fewer possible Bloom filters for Alice and Bob, and the smaller estimation error variance, because the estimation error mainly comes from the uncertainty of BF_A and BF_B.

In addition, the more common hash functions Alice and Bob share, the lower the probability that a random location index having corresponding bits set to bit-1 by at least l out of k hash functions, and thus the lower remaining entropy left for Bob's location profile after Alice testing all possible location indexes. It is thus of no surprise to see that Bob's remaining privacy against Alice decreases with both l and w.



(a) Relative Error

(b) Privacy

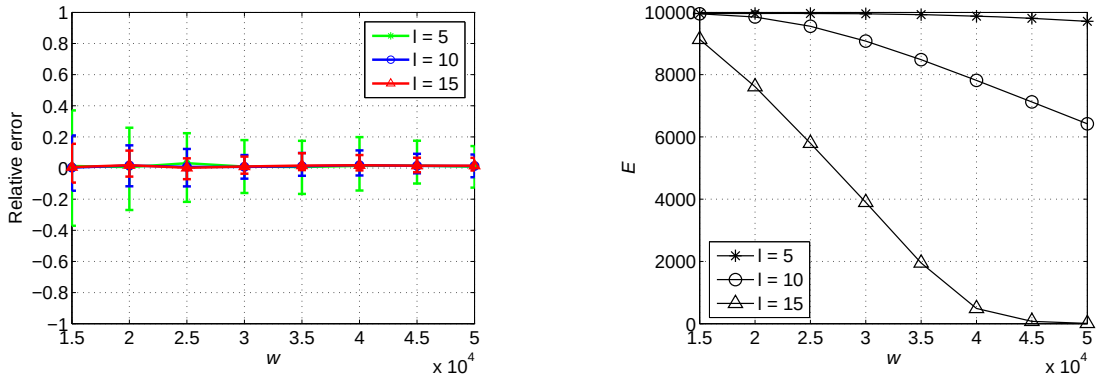
Figure 2.3: The Impact of n, the Cardinality of Profiles.

Fig. 2.3 shows the impact of n (the number of location indexes of each user) on the performance of advanced protocol, when $k = 20$, $w = 40000$, and $m = n/2$. We can see that as n increases, the relative error becomes larger. The reason is that when the Bloom-filter length w is fixed, the more elements inserted, the fewer common bit-0 positions in BF_A and BF_B, the more possible Bloom filters for Alice and Bob, which leads to higher estimation variance. In contrast, Bob's remaining privacy increases as n increases because the fewer bits-0 positions in BF_A, the higher the probability of a random location index having corresponding bits set to bit-1 by at least l out of k known hash functions, and the higher remaining entropy for Bob's location profile from Alice's point of view after testing all possible location indexes.

Fig. 2.4 shows the impact of w (the Bloom-filter length) on the performance of advanced protocol, when $k = 20$, $n = 1000$, and $m = 500$. We can see that the relative error decreases as w increases. This is because when the number of elements n is fixed, increase in w leads to more common bit-0 positions. The more common bit-0 positions, the fewer possible Bloom filters for Alice and Bob, and thus the smaller estimation error variance. In addition, Bob's remaining privacy against Alice

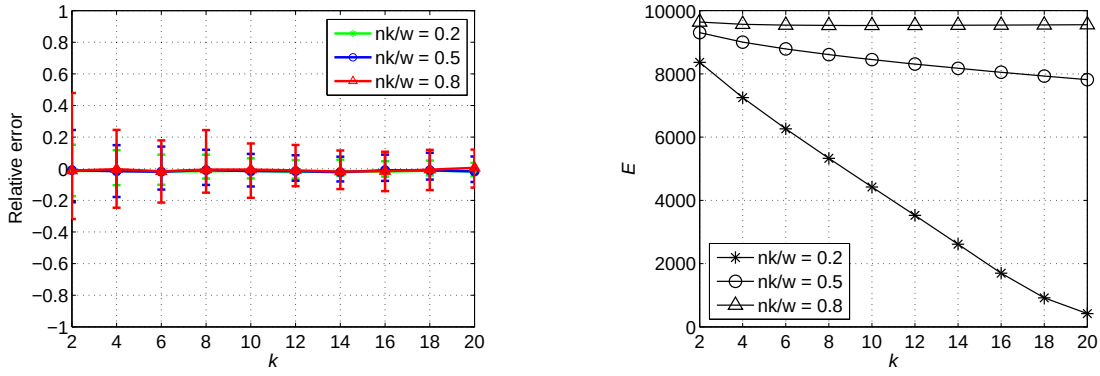
decreases as w increases. The reason is that the longer the Bloom filter, the lower the probability that a random location index having corresponding bits set to bit-1 by at least l out of k known hash functions, and thus the lower remaining entropy left for Bob's location profile after Alice testing all possible location indexes.

Fig. 2.5 shows the impact of k (the total number of hash functions for Bloom filter construction) on the performance of advanced protocol, when $n = 1000$, $m = 500$, and



(a) Relative Error (b) Privacy

Figure 2.4: The Impact of w, the Length of the Bloom Filter.



(a) Relative Error (b) Privacy

Figure 2.5: The Impact of k, the Total Number of Hash Functions.

the ratios l/k and nk/w are both fixed. It is obvious that the relative error decreases as k increases. The reason is that when k increases, l and w also increase proportionally with fixed l/k and nk/w. Recall that the variance of the $\hat{m}$ is inversely proportional to w/l^2 for fixed ρ (cf. Eq. (2.17)). As l increases, the variance of estimation error decreases. In addition, Bob's remaining privacy against Alice decreases as k increases. The reason is that the probability that at least l bit positions have been set decreases as k increases, which leads to lower remaining entropy.

From the above figures, a general conclusion we can draw is that there is an inherent tradeoff between matching accuracy and spatiotemporal privacy: the more accuracy Alice wants, the lower spatiotemporal privacy Bob can enjoy, and vice versa.

2.7 Related Work

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

There is some work on encounter-based matching [104, 105]. Manweiler *et al.* [104] discussed the privacy concerns for some missed-connection sites, which allows anonymous users to rediscover strangers that they ever encountered. In their follow-on work [105], they proposed to let mobile users exchange spatiotemporal credentials when encountering each other and later attempt to discover each other via a third-party server which acts as a rendezvous point for the users. In contrast, our protocols focus on a more general problem and are completely distributed without requiring mobile users to interact with a third-party server in most scenarios.

Existing proposals for private matching can be generally classified into two categories. The first category such as [21, 84, 98, 98, 112, 117, 146, 161] assumes that each participant's personal profile consists of multiple attributes chosen from a public set of attributes [161], which can be various interests [84], disease symptoms [98], or

friends [21] in different contexts. Private matching is then converted into Private Set Intersection (PSI) [75, 158], Private Set Intersection Cardinality (PSI-CA) [46, 56], or their variations, whereby two mutually mistrusting parties, each holding a private data set, jointly compute some function over the two sets without leaking any additional information to either party. The second category such as [49, 91, 92, 167, 173] assumes that user profile can be modeled as a multi-dimensional vector, where each element is an integer indicating the priority level, knowledge level [91], or interest level [92] of users on the corresponding attribute. Private matching is then converted into the secure computation of various functions over two vectors. Our work belongs to the first category but does not rely on computationally expensive PSI-CA.

Private proximity testing aims at testing the physical proximity of two users at some discrete time points in a privacy-preserving fashion. In [114], private proximity test is reduced to private equality test based on some location tags often sent by third parties, and the sketches of GSM location tags [93] are for efficient private proximity test. In contrast, our protocols evaluate the proximity of two users for any desired continuous time period. Moreover, our most efficient protocol does not involve expensive cryptographic operations unlike [93, 114].

2.8 Summary

In this chapter, we motivate and formulate privacy-preserving spatiotemporal matching as a fundamental primitive for supporting secure D2D communications. We present a novel privacy-preserving spatiotemporal matching protocol and a novel weighted privacy-preserving spatiotemporal matching protocol based on a novel use of the Bloom filter. Detailed performance analysis and evaluation confirm the high efficacy and efficiency of our solutions.

Chapter 3

SYNERGY: A GAME-THEORETICAL APPROACH FOR COOPERATIVE KEY GENERATION IN WIRELESS NETWORKS

3.1 Introduction

Device-to-Device (D2D) communication is quickly emerging due to the ever-growing popularity of powerful mobile devices and also the rapid advance in D2D technologies [82]. In a typical D2D session, adjacent mobile devices can directly communicate without involving a base station. The competing technologies for establishing D2D connections include Bluetooth and WiFi-direct over the unlicensed band as well as LTE-A over the licensed band. D2D communication is promising to enhance spectrum efficiency and system throughput, enable efficient traffic offloading, improve energy efficiency and network coverage, and stimulate excitingly new services.

A key challenge for advancing D2D communication is to secure a D2D connection given the ease of malicious eavesdropping on wireless transmissions. One may think about encrypting and authenticating the content sent over a D2D connection based on a secret key shared between two mobile nodes. A conventional way for establishing secret keys depends on each node owning a public-key certificate, but it is unlikely to have a public-key certificate on every mobile node in the near future. Another traditional method is through a trusted third party which may not exist in most scenarios; even if there were one, heavily involving it may largely offset the benefits from autonomous D2D communication.

It is more promising to generate a secret key directly from the wireless channel between two mobile nodes. Specifically, according to the channel reciprocity theory,

the channel responses between two wireless devices share some common randomness which is unavailable and also unpredictable to any eavesdropper more than one-half wavelength away from both devices. There have been some efforts (e.g., [23, 68, 107, 160]) whereby two mobile nodes can extract a secret key from such common channel randomness. The resulting key is information-theoretically secure, and it can be generated on demand and updated dynamically in line with time-varying and location-dependent wireless channels [79]. In addition, there is no requirement for a trusted third party or prior trust relationship between two mobile nodes. This PHY (short for physical layer) approach is thus very suitable for secure D2D communication. The rate at which secret bits are generated from the wireless channel heavily depends on how fast the channel changes. In slowly changing wireless environments, the key generation rate can be very low. This practical limitation is widely reported [61, 68, 79] and may jeopardize the potential of PHY-based secret key generation in D2D scenarios with high security demand.

Cooperative key generation [78, 79] is to improve the key generation rate of the PHY approach by incorporating additional randomness. The main idea is to explore some relay nodes in the vicinity of two target nodes and use the random channels associated with these relay nodes as additional random sources for secret key generation between the two target nodes. The efficacy of cooperative key generation is well analyzed and confirmed in various scenarios [78, 79]. The feasibility of this technique is, however, still questionable, as mobile nodes are self-interested in nature and typically reluctant to participate if they cannot get adequate benefit from the cooperation.

We propose SYNERGY, a game-theoretical approach for stimulating cooperative key generation in wireless networks. SYNERGY targets a multi-hop D2D communication scenario in which every node wishes to establish a secret key with at least one

neighbor via the PHY approach. The underlying idea of SYNERGY is to partition all the nodes involved into multiple disjoint coalitions. Every node in a coalition is strongly motivated to help other nodes in the same coalition establish secret keys to get help in return.

Our contributions can be summarized as follows. First, we are the first to study incentive-aware cooperation key generation in wireless networks to the best of our knowledge. Second, we formulate it as a coalitional game and devise an algorithm to find the core solution. Third, we propose centralized and distributed implementations for the core discovery algorithm. Finally, we show that SYNERGY is highly efficient and effective through extensive simulations.

In what follows, Section 3.2 outlines the background for cooperative key generation in wireless networks. Section 3.3 gives the system and adversary models. Section 3.4 presents a coalitional game formulation for incentive-aware cooperative key generation and the algorithm for obtaining the core solution. Section 3.5 introduces centralized and distributed implementations of SYNERGY and analyzes their performance. Section 3.6 evaluates SYNERGY using simulations. Section 3.7 briefs the related work. Section 3.8 summarizes this chapter.

3.2 Background

This section outlines the basics of PHY-based noncooperative and cooperative secret key generation. Such background information is necessary for understanding our work.

3.2.1 PHY-based Noncooperative Key Generation

Assume that two nodes Alice (A) and Bob (B) want to establish a shared secret key via the wireless channel between them in the presence of an eavesdropper Eve (E).

Both Alice and Bob can transmit, while Eve only passively eavesdrops on wireless transmissions to avoid being detected.

Key generation starts by Alice sending a signal X_A. Then Bob and Eve will receive $Y_B = h_{AB}X_A + n_B$ and $Y_E = h_{AE}X_A + n_E$, respectively. Next, Bob transmits a signal X_B, and Alice and Eve will receive $Y_A = h_{BA}X_B + n_A$ and $Y_E = h_{BE}X_B + n_E$, respectively. Here h_{AB}, h_{AE}, h_{BA}, and h_{BE} denote the channel gains from Alice to Bob, from Alice to Eve, from Bob to Alice, and from Bob to Eve, respectively; n_A, n_B, and n_E are all commonly assumed to be zero-mean additive Gaussian noise with variance σ^2.

The wireless channel between Alice and Bob is assumed to be reciprocal, which means that $h_{AB} \cong h_{BA}$. In addition, assuming that Eve is more than one-half wavelength away from Alice and Bob, h_{AB} and h_{AE} are thus uncorrelated, so are h_{BA} and h_{BE}. Also assume that the channel response is a Gaussian random variable with zero mean and variance σ_1^2. According to [18], the optimal key generation rate is

$$R_{A,B} = \frac{1}{T} I(\tilde{h}_{AB}; \tilde{h}_{BA}) = \frac{1}{2T} \log_2 \left(1 + \frac{\sigma_1^4 P^2 T^2}{4(\sigma^4 + \sigma^2 \sigma_1^2 PT)}\right), \tag{3.1}$$

where $\tilde{h}_{AB}$ denotes Bob's estimate of h_{AB}, $\tilde{h}_{BA}$ denotes Alice's estimation of h_{BA}, $I(\tilde{h}_{AB}; \tilde{h}_{BA})$ denotes the mutual information [45] of $\tilde{h}_{AB}$ and $\tilde{h}_{BA}$, T is the number of symbols during which the channel gains are fixed (i.e., coherence time), and P is the transmission power of each node.

3.2.2 PHY-based Cooperative Key Generation

Cooperative key generation [78, 79] via relay nodes is proposed to improve the key generation rate of the above noncooperative approach. The underlying idea is to explore the additional randomness brought by other nodes in the vicinity of Alice and Bob. Consider the example in Fig. 3.1, where Charlie (C) and Dave (D) are two

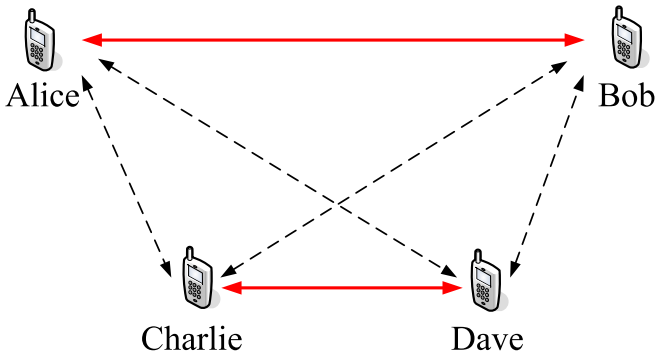


Figure 3.1: PHY-based Cooperative Key Generation. Dashed and Solid Lines Both Denote Neighboring Relationships, and a Solid Line Additionally Means That the Two Line Ends (i.e., Two Peer Nodes) Want to Establish a Secret Key.

common neighbors of Alice and Bob and thus can both serve as a relay. Cooperative key generation involving one relay, say Charlie, consists of two steps: channel estimation and key generation.

Channel Estimation

1. Alice sends a known sequence $\mathbf{S}_A$, from which Bob and Charlie estimate the channel gains h_{AB} and h_{AC} as $\tilde{h}_{AB}$ and $\tilde{h}_{AC}$, respectively.

2. Bob sends a known sequence $\mathbf{S}_B$, from which Alice and Charlie estimate the channel gains h_{BA} and h_{BC} as $\tilde{h}_{BA}$ and $\tilde{h}_{BC}$, respectively.

3. Charlie sends a known sequence $\mathbf{S}_C$, from which Alice and Bob estimate the channel gains h_{CA} and h_{CB} as $\tilde{h}_{CA}$ and $\tilde{h}_{CB}$, respectively.

Key Agreement

1. Alice and Bob establish a secret key K_{AB} based on $\tilde{h}_{AB}$ and $\tilde{h}_{BA}$. In addition, Alice and Charlie establish on a secret key K_{AC} based on $\tilde{h}_{AC}$ and $\tilde{h}_{CA}$. Finally, Bob and Charlie establish a secret key K_{BC} based on $\tilde{h}_{BC}$ and $\tilde{h}_{CB}$.

2. Charlie broadcasts $K_{AC} \oplus K_{BC}$, from which Alice and Bob each know both K_{BC} and K_{AC}. If K_{AC} is shorter than K_{BC}, Alice and Bob set the final secret key as $K_{AB} \parallel K_{AC}$ and $K_{AB} \parallel K_{BC}$ otherwise.

According to the result of [79], the optimal key generation rate of this cooperative approach is

$$R_{A,B}^{(C)} = \frac{1}{T}\Big\{ \min\{I(\tilde{h}_{CA}; \tilde{h}_{AC}), I(\tilde{h}_{CB}; \tilde{h}_{BC})\} + I(\tilde{h}_{AB}; \tilde{h}_{BA})\Big\}. \tag{3.2}$$

Similarly, if both Charlie and Dave act as relays, the optimal key generation rate is given by [79]

$$\begin{aligned} R_{A,B}^{(C,D)} =& \frac{1}{T}\Big\{ I(\tilde{h}_{AB}; \tilde{h}_{BA}) + \min\{I(\tilde{h}_{CA}; \tilde{h}_{AC}), I(\tilde{h}_{CB}; \tilde{h}_{BC})\} \\ &+ \min\{I(\tilde{h}_{DA}; \tilde{h}_{AD}), I(\tilde{h}_{DB}; \tilde{h}_{BD})\}\Big\}. \end{aligned} \tag{3.3}$$

We have two important remarks to make. First, the optimal key generation rates given above are only in information-theoretical sense. In practice, any two nodes involved have to generate a secret key from their channel estimates by following a few steps in sequel, including quantization, information reconciliation, and privacy amplification, as in [23, 68, 107, 160]. So the real key generation rates are usually smaller. Second, it can be seen from our illustration above that the relay node(s) know partial information about the eventual secret key. If this is a concern, a more advanced and also complicated technique in [79] can be applied instead. Our proposed SYNERGY can work with both techniques, but we focus on the basic technique above to facilitate the presentation.

3.3 System and Adversary Models

We consider a multi-hop D2D scenario, in which every mobile node has at least one mobile device ready for D2D communication via Bluetooth, WiFi-direct, LTE-A,

or other available D2D technologies. To enable analytical tractability, we assume that every node has the same transmission power and range. Mobile nodes are assumed to be *selfish* and *rational*. By selfish, we mean that every node will not act as a relay to help other nodes establish a secret key without a sound incentive. Our goal is to divide the nodes into disjoint coalitions, in which every node assists others establishing a secret key and also gets help from others in return. By rational, we mean that every node in a coalition faithfully follows the protocol operations and collaborates with others on key generation.

We consider the following adversary model commonly adopted for PHY-based key generation [23, 68, 78, 79, 107, 160]. Specifically, the adversary only passively eavesdrops on the wireless channel without actively jamming the channel. It is more than one-half wave length away from any two neighboring nodes trying to establish a secret key. Therefore, the adversary can only obtain the noisy versions of the wireless transmissions between two target nodes, so it cannot directly construct the secret key between them. For example, for wireless transmissions in the 2.4 GHz band, we only require the adversary to be more than 6.25 cm away from target nodes. This assumption is thus easily justifiable in practice. As in [78, 79], we assume that the nodes serving as relays for secret key generation do not cooperate with the adversary or other relays to obtain useful information about secret keys. There may be multiple eavesdroppers, which are assumed to be independent from each other. How to deal with collaborative eavesdroppers is still an open challenge.

3.4 SYNERGY: Cooperative Key Generation based on Social Reciprocity

As said, the key challenge for adopting cooperative key generation [78, 79] for D2D communication is the natural self-interest of mobile nodes: nobody wants to spend scarce system resources as a relay without getting adequate reward. We propose

SYNERGY, the first known solution to this open challenge based on the powerful theory of social reciprocity. The essential idea of SYNERGY is that a mobile node can be strongly motivated to act as a relay for other nodes if it could also get help in return to generate a secret key for itself. More specifically, SYNERGY partitions a set of mobile nodes into disjoint coalitions, each comprising some nodes acting as relays for others in cooperative key generation in order to improve their respective key generation rate. The main design challenge for SYNERGY lies in the partitioning rule for a given set of mobile nodes. In this section, we formulate this challenging issue as a coalitional game and describe an algorithm to find the core solution to the game.

3.4.1 Notation and Terms

We consider N mobile nodes denoted by $\mathcal{N} = \{1, \ldots, N\}$ and define the following terms to ease the illustration.

- **Peer**: Two nodes are said to be peers of each other iff they are physical neighbors and want to establish a secret key. The two nodes are called a *peer pair*.
- **Relay**: A relay of a peer pair refers to a node which is their common neighbor and helps them establish a secret key through cooperative key generation.
- **Contributor**: If either or both of two peers serves as a relay for another peer pair, we say that the first peer pair is a contributor to the second pair.

We assume that each node has one and only one peer in any SYNERGY session, which means that N is even. Due to the limited transmission range of mobile nodes, we can view $\mathcal{N}$ as the vertex set of an undirected graph, where every edge corresponds to two neighboring nodes. Assume that every peer pair has at least one common neighbor as a candidate relay. Otherwise, the peer pair can only establish a secret key

using noncooperative key generation and does not need to participate in SYNERGY operations. Let $\mathcal{C}_{i,j} \neq \phi$ denote the common neighbors of peers i and j. We further assume that i and j can use no more than two relays (if any) from $\mathcal{C}_{i,j} \neq \phi$, and the two-relay case can only occur when the two relays themselves compose a peer pair. The extension of SYNERGY to more general cases is very challenging and left as future work. For simplicity, we also assume that every node acts as a relay at most once in a SYNERGY session.

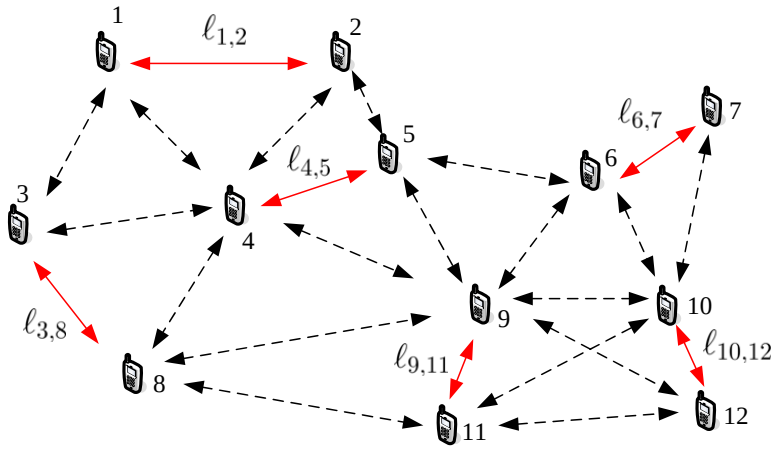


Figure 3.2: An Exemplary Multi-hop D2D Scenario, Where Dashed and Solid Lines Both Denote Neighboring Relationships, and a Solid Line Additionally Means That the Two Line Ends (i.e., Two Peer Nodes) Want to Establish a Secret Key.

As an example, we have $\mathcal{N} = \{1, \ldots, 12\}$, $\mathcal{C}_{1,2} = \{4\}$, $\mathcal{C}_{4,5} = \{2, 9\}$, $\mathcal{C}_{9,11} = \{8, 10, 12\}$in Fig. 3.2. Peers 9 and 11 can potentially use nodes 10 and 12 as two relays because nodes 10 and 12 also form a peer pair. In contrast, peers 4 and 5 can have at most one relay, either node 2 or 9.

3.4.2 Coalitional Game Formulation

In game theory, a coalitional game refers to a game where a competition is between coalitions of players instead of between individual players [110]. It is thus a very natural tool for incentive-aware cooperative key generation.

Our coalitional game formulation relies on a special trick. We introduce a **virtual node** (denoted by $\ell_{i,j}$) for every peer pair i and j, as shown in Fig. 3.2. Note that we have $\ell_{i,j} = \ell_{j,i}$. Now consider any other virtual node $\ell_{s,d}$ ($i \neq j \neq s \neq d$). If either or both of s and d act as a relay for peers i and j, we say that $\ell_{s,d}$ **contributes** to $\ell_{i,j}$. Since every peer pair can be assumed to have a common interest in improving their key generation rate, we can use the $N/2$ virtual nodes as the game players rather than the N real nodes.

What is the most preferred contributor of every virtual node $\ell_{i,j}$, or equivalently the most preferred relay of peers i and j? Recall that the optimal key generation rates with one relay and two relays are given in Eq. (3.2) and Eq. (3.3), respectively. To answer the preceding question, let $\mathcal{L}_{i,j}$ denote the set of potential contributors to $\ell_{i,j}$. For example, we have $\mathcal{L}_{4,5} = \{\ell_{1,2}, \ell_{9,11}\}$ in Fig. 3.2. Consider any virtual node in $\mathcal{L}_{i,j}$, say $\ell_{s,d}$. It contributes one potential relay to $\mathcal{L}_{i,j}$ if only one of s and d is a common neighbor of i and j and two potential relays if both s and d are a common neighbor of i and j. Accordingly, we define the key-rate function of $\ell_{i,j}$ with regard to any potential contributor $\ell_{s,d} \in \mathcal{L}_{i,j}$ as

$$\hat{R}_{i,j}^{s,d} = \begin{cases} R_{i,j}^{(s)} & \forall s \in \mathcal{C}_{i,j}, d \notin \mathcal{C}_{i,j} \\ R_{i,j}^{(d)} & \forall s \notin \mathcal{C}_{i,j}, d \in \mathcal{C}_{i,j} \\ R_{i,j}^{(s,d)} & \forall s, d \in \mathcal{C}_{i,j} \ . \end{cases} \tag{3.4}$$

The most preferred virtual node or contributor of $\ell_{i,j}$ can then be defined as $r_{i,j}^* = \underset{\ell_{s,d} \in \mathcal{L}_{i,j}}{\operatorname{argmax}} \hat{R}_{i,j}^{s,d}$.

Based on the concepts above, we now formulate incentive-aware cooperative key generation as a coalitional game $\Omega = \langle \mathcal{L}, \mathcal{X}_{\mathcal{L}}, \Theta, (\succ_{i,j})_{\ell_{i,j} \in \mathcal{L}} \rangle$ as follows.

- **Players**: $\mathcal{L}$ denotes the set of game players consisting of all the $N/2$ virtual nodes.

- **Strategies**: $\mathcal{X}_{\mathcal{L}}$ denotes the set of feasible cooperation strategies (i.e., contributor selections) for all the players. We denote the contributor chosen by any player $\ell_{i,j} \in \mathcal{L}$ by $r_{i,j} \in \mathcal{L}$. It follows that $\mathcal{X}_{\mathcal{L}} = \{r_{i,j} | r_{i,j} \in \mathcal{L}, \forall \ell_{i,j} \in \mathcal{L}\}$.

- **Characteristic function**: Every virtual node in every coalition $\mathcal{S} \subseteq \mathcal{L}$ selects one and only one other virtual node in $\mathcal{S}$ as a contributor. In addition, every virtual node outside $\mathcal{S}$ cannot get an contributor from $\mathcal{L}$. Therefore, the characteristic function for every coalition $\mathcal{S} \subseteq \mathcal{L}$ can be denoted as $\Theta(\mathcal{S}) = \big\{\{r_{i,j}\}_{\ell_{i,j} \in \mathcal{S}} = \{\ell_{i,j}\}_{\ell_{i,j} \in \mathcal{S}}, \{r_{i,j} = \ell_{i,j}\}_{\ell_{i,j} \in \mathcal{L} \setminus \mathcal{S}}\big\}$.

- **Preference order**: If a virtual node $\ell_{i,j} \in \mathcal{L}$ chooses to compare the performance of any two virtual nodes in $\mathcal{L}_{i,j}$, say ℓ_{s_1,d_1} and ℓ_{s_2,d_2}, its preference order is defined as $\ell_{s_1,d_1} \succ_{i,j} \ell_{s_2,d_2}$ if ℓ_{s_1,d_1} is determined to have better performance. Here the performance refers to the key generation rate defined in Eq. (3.2) for one relay or in Eq. (3.3) for two relays.

Similar to Nash equilibrium in a non-cooperative game, the *core* plays an essential role in a coalitional game. Generally speaking, the core refers to a set of cooperation strategies such that no coalition can deviate and improve for all its members by cooperation within the coalition [110]. The core of our game Ω is a set of contributor selection strategies $r_{i,j} \in \Theta(\mathcal{L})$ where there are no coalition $\mathcal{S}$ and $\tilde{r}_{i,j} \in \Theta(\mathcal{S})$ such that $\tilde{r}_{i,j} \succ_{i,j} r_{i,j}$ for all $\ell_{i,j} \in \mathcal{S}$. It means that no improvement on the key generation rate can be made by cooperation within the coalition $\mathcal{S}$. We can prove the existence

of a core solution to game Ω. Due to space limitations, we omit the proof here and refer interested readers to [136] for details.

3.4.3 Core Discovery Algorithm

This section introduces our core discovery algorithm. For this purpose, we first introduce two concepts as follows.

Definition 3.4.1. ***(Coalitional Subgame)*** *Given a coalitional game* $\Omega = \langle \mathcal{L}, \mathcal{X}_{\mathcal{L}}, \Theta, (\succ_{i,j})_{\ell_{i,j} \in \mathcal{L}} \rangle$*, we call a coalitional game* $\Psi = \langle \mathcal{M}, \mathcal{X}_{\mathcal{M}}, \Theta, (\succ_{i,j})_{\ell_{i,j} \in \mathcal{M}} \rangle$ *a coalitional subgame of* Ω *iff* $\mathcal{M} \subseteq \mathcal{L}$ *and* $\mathcal{M} \neq \emptyset$*.*

Definition 3.4.2. ***(Contributor Cycle)*** *Given a coalitional subgame* $\Psi = \langle \mathcal{M}, \mathcal{X}_{\mathcal{M}}, \Theta, (\succ_{i,j})_{\ell_{i,j} \in \mathcal{M}} \rangle$*, a sequence of virtual nodes,* $(\ell_{i_1,j_1}, \ldots, \ell_{i_H,j_H})$*, is called a contributor cycle of length* H *if and only if* $r_{i_x,j_x} = \ell_{i_{x+1},j_{x+1}}$ *for* $\forall x \in [1, H-1]$ *and* $r_{i_H,j_H} = \ell_{r_1,j_1}$*.*

A contributor cycle of length one clearly contains a single virtual node, which means that the two mobile nodes forming this virtual node cannot find a relay and thus should directly generate a secret key using noncooperative key generation in Section 3.2.1. In contrast, a contributor cycle of length $H \geq 2$ means that every virtual node in the contributor cycle has its most preferred contributor as the next virtual node of the same cycle in the circular fashion. Every contributor cycle thus corresponds to a coalition, in which the mobile nodes involved are reciprocal for key generation.

Our core discovery algorithm is to iteratively identify all the contributor cycles which form a core solution. We achieve this by first constructing a directed graph $\mathbf{G} = (\mathcal{L}, \mathcal{E})$, where a directed edge from any vertex $\ell_{i,j}$ to another vertex $\ell_{s,d}$ exists if and only if $\ell_{s,d}$ is the most preferred contributor of $\ell_{i,j}$, i.e., $r^*_{i,j} = \ell_{s,d}$. Recall that every virtual node can choose at most one contributor, which corresponds to

at most two relays. The outdegree of every vertex $\mathbf{G}$ is thus one if it has at least one candidate contributor and zero otherwise. The problem of discovering all the contributor cycles or the core solution can then be translated into simple-cycle search in $\mathbf{G}$. In particular, a path in a graph refers to a sequence of edges which connect a sequence of vertices, a cycle is a path with the same start and end vertices, and a cycle with no repeated vertices or edges except the start and end vertices is called a simple cycle. Since every vertex in $\mathbf{G}$ corresponds to a virtual node, a simple cycle is equivalently a contributor cycle. So we can denote a simple path of H vertices by a contributor cycle of H virtual nodes as $(\ell_{i_1,j_1}, \ldots, \ell_{i_H,j_H})$. If there is a simple path of $H = |\mathcal{L}| = N/2$ vertices, all the $N/2$ virtual nodes or N mobile nodes are in a single coalition. In contrast, any simple path of one vertex (i.e., a self contributor cycle) means that the two mobile nodes related to the vertex do not use any relay for secret key generation. The following proposition underlies our simple-cycle (contributor-cycle) search algorithm.

Proposition 3.4.1. *A simple path beginning from any vertex in the directed graph* $\mathbf{G}$ *results in one and only one simple cycle.*

Proof. It is easy to prove that a simple path beginning from any vertex in $\mathbf{G}$ must lead to a simple cycle, as otherwise there must be infinite vertices in $\mathbf{G}$. Now we prove the uniqueness of the resulting simple cycle. If multiple simple cycles exist, there must be at least one vertex whose outdegree is larger than one. This contradicts with the property of $\mathbf{G}$ that the outdegree of every vertex is no more than one. □

Another way to interpret Proposition 3.4.1 is that every vertex (virtual node) in $\mathbf{G}$ is on one and only one contributor cycle, possibly a self cycle involving itself only. Then we can discover all the contributor cycles and thus implement the core solution to game Ω as follows. Initially, all the vertices in $\mathbf{G}$ are marked unvisited. We can

start a walk from any unvisited vertex, say $\ell_{i,j}$, until when the walk either hits an visited vertex or returns to $\ell_{i,j}$. In the former case, $\ell_{i,j}$ is marked visited and forms a self contributor cycle. If the later case occurs, a new contributor cycle is found, and all the vertices on the cycle are marked visited. This process continues until either when all the vertices in $\mathbf{G}$ are marked visited or when none of the remaining unvisited vertices can be the start of a walk towards unvisited vertices. In the later case, we mark all the remaining vertices visited and terminate the algorithm.

3.5 Implementations

In this section, we present C-SYNERGY and D-SYNERGY, two protocols to implement SYNERGY in centralized and distributed fashions, respectively. We also analyze the security, computational overhead, and communication overhead of C-SYNERGY and D-SYNERGY.

3.5.1 C-SYNERGY: A Centralized Implementation

In C-SYNERGY, every peer pair reports its own preference order to a single server which computes all the contributor cycles and returns each to the corresponding nodes. The server can be a base station if available or a mobile node elected from the mobile nodes themselves. It is worth emphasizing that the server merely does the computation and does not participate in cooperative key generation in the server's role. So it is blind to the final secret key of any peer pair.

Detailed Operations

Recall that $\mathcal{N} = \{1, \ldots, N\}$ denote the N nodes involved. C-SYNERGY works as follows.

1. Every node $i \in \mathcal{N}$ locally broadcasts a hello message including its ID and also records the IDs in received hello messages. Let $\mathcal{C}_i$ denote the neighbor IDs of node i.

2. Every two neighboring nodes estimate the channel response between them by exchanging probe messages. The estimated variance is needed for deriving the optimal key rates according to Eq. (3.2) or Eq. (3.3).

3. Every two peers, say $i, j \in \mathcal{N}$, exchange $\mathcal{C}_i$ and $\mathcal{C}_j$ to identify their common neighbors as $\mathcal{C}_{i,j} = \mathcal{C}_i \cap \mathcal{C}_j$.

4. Every node $i \in \mathcal{N}$ locally broadcasts its peer ID and also records the peer ID of every neighbor. The peer IDs allow i and its peer j to learn their local topology and the associated peer pairs, based on which to construct the list of candidate contributors, i.e., $\mathcal{L}_{i,j}$.

5. Every two peers, say $i, j \in \mathcal{N}$, compute the optimal key rate for each candidate contributor in $\mathcal{L}_{i,j}$ based on Eq. (3.4). Then they determine the most preferred contributor which is reported by either of them to the server.

6. The server applies the core discovery algorithm in Section 3.4.3 on the received information to compute all the contributor cycles and finally returns every contributor cycle to each node in that cycle.

7. The nodes in every contributor cycle work together to derive their respective secret key as in Section 3.2.2.

An Example

We shed more light on C-SYNERGY using the example in Fig. 3.2, where every two peers i, j are represented with a solid line and annotated by the corresponding virtual

node $\ell_{i,j}$. Every two peers jointly determine the preference order for their candidate contributors. We assume that the preference orders are as given in Table 3.1. For instance, $\ell_{1,2}$ has only node 4 as a candidate relay, so its most preferred relay (or contributor) is simply node 4 (or virtual node $\ell_{4,5}$). In addition, $\ell_{9,11}$ has a preference order $8 \succ (12, 10)$, which means that it prefers node 8 as a relay and equivalently virtual node $\ell_{3,8}$ as a contributor.

Table 3.1: A Preference-order Table, Where (i, j) Means Both i and j Serve as a Relay for the Corresponding Virtual Node.

Virtual Node	**Preferred Real Node**	**Preferred Virtual Node**
$\ell_{1,2}$	4	$\ell_{4,5}$
$\ell_{3,8}$	4	$\ell_{4,5}$
$\ell_{4,5}$	2$\succ$9	$\ell_{1,2} \succ \ell_{9,11}$
$\ell_{9,11}$	8$\succ$(12,10)	$\ell_{3,8} \succ \ell_{10,12}$
$\ell_{10,12}$	(9,11)	$\ell_{9,11}$
$\ell_{6,7}$	10	$\ell_{10,12}$

Based on the received preference orders, the server applies the core discovery algorithm in Section 3.4.3 to derive the contributor cycles. Specifically, the server first constructs a directed graph with six vertices $\{\ell_{1,2}, \ell_{3,8}, \ell_{4,5}, \ell_{9,11}, \ell_{10,12}, \ell_{6,7}\}$. Since $\ell_{4,5}$ is reported as the most preferred contributor of $\ell_{1,2}$, the server adds an edge from $\ell_{1,2}$ to $\ell_{4,5}$ in $\mathbf{G}$. Other edges of $\mathbf{G}$ are added similarly. The resulting graph $\mathbf{G}$ is shown in Fig. 3.3(a). In iteration 1, the server identifies one contributor cycle $(\ell_{1,2}, \ell_{4,5})$ and removes it from $\mathbf{G}$. The modified graph is shown in Fig. 3.3(b). In iteration 2, the server identifies a self cycle consisting of $\ell_{3,5}$ only and also removes it. Subsequently, a cycle $(\ell_{9,11}, \ell_{10,12})$ is identified in iteration 3, and a self cycle containing $\ell_{6,7}$ only is

identified in iteration 4. Therefore, there are four contributor cycles or coalitions in total, including $(\ell_{1,2}, \ell_{4,5})$, $(\ell_{3,5})$, $(\ell_{9,11}, \ell_{10,12})$, and $(\ell_{6,7})$. Finally, the server returns every contributor cycle to every node involved in that cycle. The mobile nodes can then determine which nodes they can use as a relay and whom they should act as a relay for. For example, nodes 9 and 11 use both nodes 10 and 12 as a relay, and nodes 10 and 12 use both nodes 9 and 11 as a relay. They can all assure that collaborating with each other is the best strategy to improve their respective key generation rate.

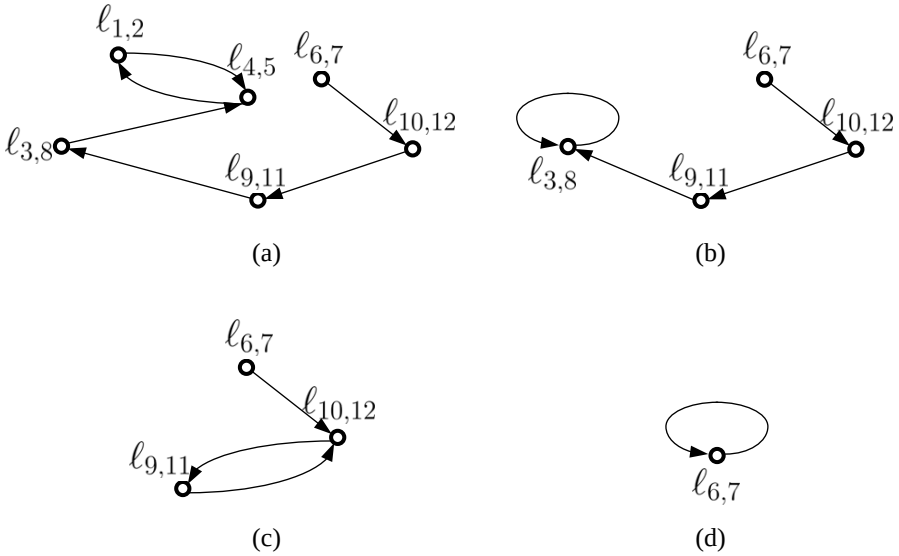


Figure 3.3: Illustration of Contributor Cycle Discovery.

3.5.2 D-SYNERGY: A Distributed Implementation

D-SYNERGY enables the mobile nodes to discover the core solution (or contributor cycles) in a purely distributed fashion. To emulate centralized core discovery in C-SYNERGY, D-SYNERGY also works in iterations. Every iteration is initiated by a mobile node not in any identified contributor cycle, and a new contributor cycle is identified in every iteration. D-SYNERGY terminates when every node in $\mathcal{N}$ is included in a contributor cycle.

Detailed Operations

We introduce two binary flags f_i and v_i for each node $i \in \mathcal{N}$. Referred to as an *inclusion flag*, f_i is initially zero and permanently set to one after a contributor cycle including i is discovered. In contrast, v_i is called a *visit flag* and equals zero if node i has not been included in any contributor cycle at the beginning of an iteration. It is set to and remains one when node i receives a core-discovery message in an iteration. In addition, v_i remains one after node i is included in any contributor cycle. D-SYNERGY works as follows.

1. All the mobile nodes in $\mathcal{N}$ act according to the first five steps of C-SYNERGY.

2. An iteration starts when any node $i \in \mathcal{N}$ with $f_i = 0$ broadcasts a BUSY message. The BUSY message reaches other nodes in $\mathcal{N}$ in a hop-by-hop fashion. Every node resets its visit flag to zero after seeing the BUSY message unless its inclusion flag is one. Multiple nodes may try to initiate an iteration simultaneously, in which case the one with a smaller ID always wins.

3. Node i then sends a REQ_i message to its peer, say node j, and also the most preferred relay. Then i and j sets $v_i = 1$ and $v_j = 1$, respectively.

4. Any node $s \neq i$ may receive REQ_i from its peer or a node considering it the most preferred relay. In the former case, node s does nothing other than recording REQ_i and setting $v_s = 1$, as its peer has taken care of REQ_i. This operation is designed because s and its peer form a single virtual node in the directed graph $\mathbf{G}$ used in C-SYNERGY. So we let node s and its peer have synchronized internal states to emulate a single virtual node in $\mathbf{G}$. In the latter case, node s does the following operations in sequel.

- If $f_s = 1$, node s sends a REJ message to the node who sent it REQ_i. The receiver of a REJ message then sends REQ_i to its next most preferred relay. If all its candidate relays respond with a REJ message, it returns a REJ message to its own neighbor which sent it REQ_i. If i gets REJ from all its candidate relays, it sets $f_i = 1$, notifies its peer j to set $f_j = 1$, and broadcasts an EXIT message to terminate this iteration. In this case, i and j have to establish a secret key without using any relay, which corresponds to the case that the virtual node $\ell_{i,j}$ in $\mathbf{G}$ belongs to a self contributor cycle containing $\ell_{i,j}$ only.
- If $f_s = 0$ and $v_s = 1$, node s checks whether it has seen REQ_i before. If so, a new contributor cycle is discovered. Node s then notifies every node in this contributor cycle which can be obtained from REQ_i. Subsequently, all the nodes in the contributor cycle set their inclusion flag to one. Finally, node s broadcasts an EXIT message to terminate this iteration.
- If $f_s = 0$ and $v_s = 0$, node s appends s and its peer ID d to REQ_i, sets $v_s = 1$, and then sends the modified REQ_i to node d and also most preferred relay.

D-SYNERGY terminates when all the nodes in $\mathcal{N}$ have their inclusion flags set to one.

An Example

We still use the example in Fig. 3.2 and the preference orders in Table 3.1 to clarify D-SYNERGY.

Assume that node 9 starts the first iteration, in which the inclusion flags $\{f_i\}_{i=1}^{12}$ and visit flags $\{v_i\}_{i=1}^{12}$ are all zero initially. Node 9 sends REQ_9 to its peer (node 11)

and its most preferred relay (node 8). Then nodes 9 and 11 set $f_9 = 1$ and $f_{11} = 1$, respectively. After receiving REQ_9, node 8 finds that $f_8 = 0$ and $v_8 = 0$. So node 8 sends $\langle \text{REQ}_9 \parallel (3, 8) \rangle$ to its peer (node 3) and most preferred relay (node 4). Next, nodes 8 and 3 set $v_8 = 1$ and $v_3 = 1$, respectively.

After receiving $\text{REQ}_9 \parallel (3, 8)$, node 4 finds that $f_4 = 0$ and $v_4 = 0$. So node 4 sends $\langle \text{REQ}_9 \parallel (3, 8) \parallel (4, 5) \rangle$ to its peer (node 5) and most preferred relay (node 2). Next, nodes 4 and 5 set $v_4 = 1$ and $v_5 = 1$, respectively. Similarly, node 2 sends $\langle \text{REQ}_9 \parallel (3, 8) \parallel (4, 5) \parallel (1, 2) \rangle$ to its peer (node 1) and most preferred relay (node 4). Also, nodes 2 and 1 set $v_2 = 1$ and $v_1 = 1$, respectively.

After receiving $\langle \text{REQ}_9 \parallel (3, 8) \parallel (4, 5) \parallel (1, 2) \rangle$, node 4 finds that $f_4 = 0$ and $v_4 = 1$. In addition, it has seen REQ_9 before, so there is a contributor cycle including peer pairs $(4, 5)$ and $(1, 2)$, which can also be represented by virtual nodes $(\ell_{1,2}, \ell_{4,5})$. Then node 4 broadcasts the contributor cycle and an EXIT message to all the other nodes. Subsequently, the nodes 1, 2, 4, and 5 all have their inclusion flag set to one. Finally, all the remaining nodes, i.e., $\{3, 8, 9, 11, 6, 7, 10, 12\}$, set their visit flag to zero and enter the next iteration. This process continues until finding three other contributor cycles as $(\ell_{3,5})$, $(\ell_{9,11}, \ell_{10,12})$, and $(\ell_{6,7})$.

3.5.3 Performance Analysis

In this section, we analyze the security, computational overhead, and communication overhead of SYNERGY (C-SYNERGY and D-SYNERGY).

Security Analysis: The security of the generated secret key is first guaranteed by the key generation process introduced in Section 3.2.2. The generated secret key is provably secure from any eavesdropper who experiences an independent wireless channel from the legitimate nodes [79]. In addition, neither C-SYNERGY nor D-SYNERGY discloses any secret-key information to eavesdroppers. Although eaves-

droppers might overhear the candidate relay nodes and the preference order of each peer pair who want to establish a secret key, they cannot be in the proximity of any legitimate node and thus still cannot extract any useful information from the wireless channel. Furthermore, although a relay node knows partial information about a secret key, it is blind to the rest information tied to the wireless channel between the peer nodes it assists. As note that SYNERGY can be easily adapted to work with the most advanced cooperative key generation technique in [79] such that the relay nodes know nothing about the final secret key. We finally want to point out that C-SYNERGY and D-SYNERGY are both vulnerable to active attacks on modifying the information exchange to and from every mobile node. Such active attacks can be mitigated, e.g., by authenticating the exchanged information using a temporal group key chosen by any involved node. Same as all previous work on PHY-based secret key generation, we focus on passive eavesdropping attacks in this chapter. A detailed treatment of active attacks is beyond the scope of this chapter.

Computational Overhead: SYNERGY's computational overhead is mainly incurred in the process of discovering the contributor cycles or the core solution. In particular, according to the description of the core discovery algorithm in Section 3.4.3, we can easily see that the computation complexity of SYNERGY is $\mathcal{O}(|\mathcal{L}_j|)$ at jth iteration, where $|\mathcal{L}_j|$ denotes the number of virtual nodes involved in the jth iteration. Therefore, the overall computation complexity for all J iterations is $\mathcal{O}(\sum_{j=1}^{J}|\mathcal{L}_j|)$. Although $|\mathcal{L}_j|$ and J depend on many factors and cannot be precisely determined, we can estimate the lower and upper bounds for the computational complexity. Specifically, the lower bound is $\mathcal{O}(N)$ which is achieved when all the $N/2$ virtual nodes form a single contributor cycle in the first iteration. In contrast, the upper bound is attained if every virtual node is fond to form a self contributor cycle, leading to $N/2$ iterations in total. This corresponds to an upper bound $\mathcal{O}(\sum_{j=1}^{J}|\mathcal{L}_j|) = \mathcal{O}(\sum_{i=1}^{N/2} i) = \mathcal{O}(N^2)$.

The computations are performed at a single server in C-SYNERGY and distributed over mobile nodes in D-SYNERGY.

Communication Overhead: The communication overhead of two SYNERGY implementations is different. Specifically, the communication overhead of C-SYNERGY is mainly incurred in channel estimation, neighbor discovery, and communication between mobile nodes and the server. It can be estimated by $\mathcal{O}(\tilde{N}^2)$, where $\tilde{N}$ is the average number of neighbors every node has. In addition to the above overhead, D-SYNERGY incurs some message overhead in the distributed core-discovery phase. Its communication overhead can be lower-bounded by $\mathcal{O}(N)$, which occurs when all the virtual nodes form a contributor cycle in one iteration, and upper-bounded by $\mathcal{O}(N^2)$, which is incurred when each iteration produces a self cycle containing a unique virtual node. Therefore, the overall communication overhead of D-SYNERGY is larger than that of C-SYNERGY. So we can prefer C-SYNERGY to D-SYNERGY unless a base station does not exist, and no mobile node can be elected as a server.

3.6 Performance Evaluation

In this section, we evaluate the performance of C-SYNERGY and D-SYNERGY via Matlab simulations. The simulation strategy and settings are as follows. We consider a square region with a side length of D meters. We randomly deploy N nodes in the square region and assume that the transmission range of each node is a circle of radius A meters. We set coherent time $T = 20$ symbols, and the Gaussian noise variance $\sigma^2 = 1$. Then the channel variances between every two neighboring nodes are set to be a random variable with uniform distribution. In addition, we randomly select two nodes within each other's transmission range as a peer pair to establish a secret key. Each point in the following figures represents the average value

of 1000 runs. Since the results for C-SYNERGY and D-SYNERGY are the same in Figs. 3.4∼3.7, we do not differentiate them there.

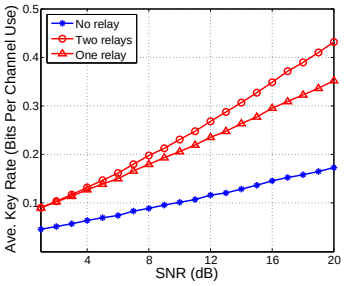


Figure 3.4: Average Key Rate for 20 Users.

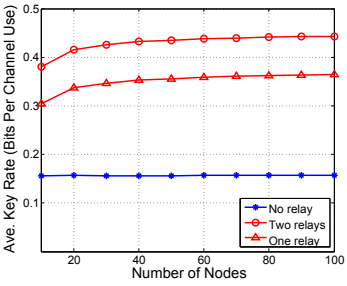


Figure 3.5: Average Key Rate for $D = 200$ m.

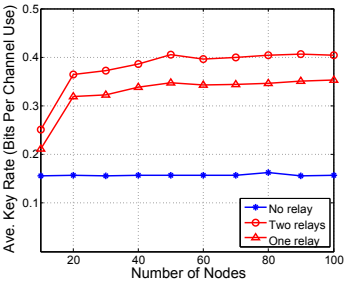


Figure 3.6: Average Key Rate for $D = 300$ m.

Fig. 3.4 illustrates the impact of SNR on the optimal key rates with no relay, with one relay, and with two relays. The first case corresponds to non-cooperative key generation, and the later two cases correspond to SYNERGY (cooperative key generation). In this set of simulations, we fix the region side length $D = 200$ m and set the transmission range of each node large enough to cover the whole square region. Besides, we fix the number of users to $N = 20$ and increase SNR from 1 to 20 dB. From Fig. 3.4, we can clearly see that as SNR increases, the optimal key rates of the three cases all increase. This is anticipated because the key generation rate increases with the transmission power according to Eqs. (3.1)∼(3.3). Moreover, the optimal key rate of SYNERGY always outperforms non-cooperative key generation, and it is always better to use two relays (if any) than using one relay. This is also as expected because the more relays, the more common channel randomness available for secret key generation.

Fig. 3.5 demonstrates the impact of the average number of nodes on the optimal key rate. In this set of simulations, we fix SNR = 20 dB, the transmission range of each node to 200 m, and the region side length to $D = 200$ m. We also vary the

number of nodes from $N = 10$ to 100. We can observe that the optimal key rate of SYNERGY is much higher than that of non-cooperative key generation. In addition, the optimal key rate of non-cooperative key generation is almost stable along with the increase of users, as it only depends on the channel condition between two peer nodes who want to generate a secret key and does not rely on any other node. In contrast, the more nodes in a fixed region, the more candidate relay nodes available for two peer nodes. So we can observe that the optimal key rate of SYNERGY increases with the number of users.

Fig. 3.6 shows the impact of the average number of neighbors on the optimal key rate when the region side length is $D = 300$ m. Other simulation settings are the same for generating Fig. 3.5, so the entire region is larger than every node's transmission range. We have almost the same observations as in Fig. 3.6 due to the same reason. In addition, the optimal key rate of SYNERGY in Fig. 3.6 is always lower than that in Fig. 3.5 for the same N, as the larger the region, the less likely that two peer nodes can find a common neighbor as a relay node. Another observation is that the gap between the optimal key rates of the one-relay and two-relay cases becomes smaller in contrast to Fig. 3.5. The reason is that a larger region makes it more difficult for two peer nodes to find two common neighbors as two relay nodes who themselves need to be two peer nodes as well according to our requirement in SYNERGY. Moreover, when there are fewer than 20 users, the optimal key rate of SYNERGY is only slightly better than non-cooperative key generation, as most peer pairs cannot find a relay in their common transmission range. Finally, it is still better to use two relays than using one relay in SYNERGY.

Fig. 3.7 shows the impact of the number of nodes on the average number of operations for discovering all contributor cycles. In this set of simulations, we fix SNR = 20 dB and each node's transmission range to 200 m. As we can see, the

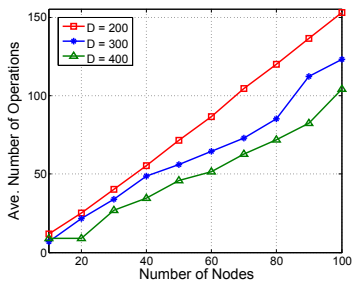


Figure 3.7: Comp. Overhead (SYNERGY).

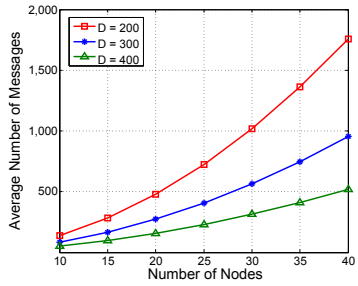


Figure 3.8: Comm. Overhead (C-SYNERGY).

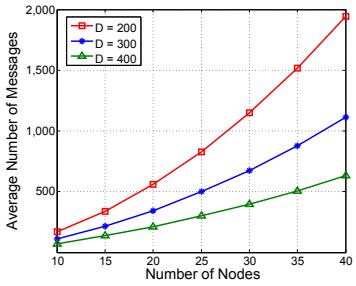


Figure 3.9: Comm. Overhead (D-SYNERGY).

average number of operations needed increases almost linearly with number of nodes. Since the overall computational overhead of SYNERGY is dominated by contributor-cycle discovery, this result confirms the high computational efficiency of SYNERGY. As said, the computational overhead of SYNERGY is incurred at a single server in C-SYNERGY but distributed over the N mobile nodes in D-SYNERGY. In addition, we compare the average number of operations needed when the region side length $D = 200$ m, 300 m, and 400 m. For a fixed number of nodes, the larger the region, the fewer common neighbors and thus candidate relay nodes every two peer nodes have, the fewer edges in the graph $\mathbf{G}$ composed of virtual nodes, and the fewer operations needed for contributor-cycle discovery. This conjecture is confirmed in Fig. 3.7.

Fig. 3.8 demonstrates the impact of the number of nodes on the communication overhead of C-SYNERGY. In this set of simulation, we fix SNR = 20 dB and each node's transmission range to 200 m. The communication overhead lies in the messages for neighbor discovery, channel estimation, and communicating with the server. The total number of messages and thus the communication overhead obviously would increase with the number of nodes, as shown in Fig. 3.8. For a given number of nodes, the larger the region, the fewer neighbors each node has, and the fewer messages needed for neighbor discovery and channel estimation. So we can see that

the communication overhead of C-SYNERGY decreases as the region side length D increases.

Fig. 3.9 illustrates the impact of the number of nodes on the communication overhead of D-SYNERGY under the same simulation settings for Fig. 3.8. The simulation results show the similar trend in Fig. 3.8 due to the similar reason. One point we want to point out is that a larger region for a given number of nodes can decrease the likelihood that two peer nodes find a relay node in their common communication range, leading to possibly fewer edges between virtual nodes in the directed graph $\mathbf{G}$. As such, the number of messages incurred in channel estimation and distributed relay-cycle discovery is likely to be reduced. This factor also contributes to the reduced communication overhead in Fig. 3.9 as the region side length D increases. Furthermore, D-SYNERGY has higher communication overhead than C-SYNERGY due to distributed contributor-cycle discovery, but it does not need a base station or an elected node as a server doing centralized computation.

As a summary of the above simulation results, the more neighbors each node has, the higher the optimal key generation rate, and the higher the computational and communication overhead of SYNERGY. There is thus an inherent tradeoff between the key generation rate and the associated computational/communication overhead.

3.7 Related Work

In this section, we briefly discuss some work most germane to SYNERGY, which is divided into two categories.

Secret Key Generation from Wireless Channels. There has been tremendous effort on exploring the channel reciprocity to establish a secret key between two mobile nodes. For example, the work in [68] focuses on using spatial and temporal variations of the wireless channel, while [160] focuses on exploring multi-antenna diversity for

secret bit extraction. The work in [95] aims at group key establishment in star and chain networks, and the work in [99] targets secret key establishment in body area networks. The channel information used in [68, 95, 99, 160] is RSS (Received Signal Strength). In contrast, the work in [107] tries to extract a secret key from the channel response between two wireless devices. There is also research on using the phase change of received signals for secret key generation in UWB systems [150] and OFDM systems [125], respectively. This line of work [68, 95, 99, 107, 125, 150, 160] can be regarded as different realizations of the information-theoretical approach in [18] and lead to different approximations to the optimal key rate in Eq. (3.1). In addition, this line of work [68, 95, 99, 107, 125, 150, 160] belongs to non-cooperative key generation, as only the direct wireless channel between two wireless devices is explored. As such, the key generation rate in [68, 95, 99, 107, 125, 150, 160] can be very low in slowly changing wireless environments. In contrast, SYNERGY has a different focus on stimulating mobile nodes in helping others establish a secret key to get help in return. Once SYNERGY identifies the contributor cycles, the techniques in [68, 95, 99, 107, 125, 150, 160] can all be adopted to establish a secret key between two mobile nodes as well as between each node and each of their relays. The resulting keys can finally be combined to produce the actual secret key between the two nodes, as illustrated in Section 3.2.2.

Cooperative key generation [78, 79, 172] is a relatively new research topic. The work in [78, 79] investigates relay-assisted strategies to improve the key generation rate by incorporating additional randomness brought by the relay nodes who are the common neighbors of two mobile nodes under consideration. In addition, the work in [172] studies secret key generation in a two-way relay channel, where there is no direct wireless channel between two mobile nodes who want to establish a secret key. A critical issue that has been overlooked in [78, 79, 172] is that mobile nodes are

selfish in nature and will not act as relays for others without adequate reward in return. SYNERGY fills this great void.

Cooperative Communication via Social Reciprocity. SYNERGY is motivated by the recent work on cooperative communication [39]. Specifically, the work in [39] targets a multi-hop D2D communication scenario, in which each node can choose to serve as a relay for other nodes. A novel coalitional game-theoretical framework is developed to design cooperation strategies based on social trust and social reciprocity. The authors prove the existence of a core solution and propose a mechanism to implement the core solution by identifying reciprocal cycles, each of which contains the nodes motivated to act as relays for others in the same cycle. In contrast to [39], SYNERGY focuses on cooperative key generation, a very different problem. In addition, the game formulation in [39] cannot be directly applied, as each game player in our scenario corresponds to two nodes instead of one as in [39]. Moreover, each node in [39] can use at most one relay node in its vicinity, while each node in SYNERGY can use two relays to achieve a higher key generation rate than using one relay.

3.8 Summary

In this chapter, we study secret key establishment, a fundamental challenge for securing D2D communication. We propose SYNERGY, a game-theoretical approach for stimulating PHY-based cooperative key generation in wireless networks as the first work of its kind. In SYNERGY, incentive-aware cooperative key generation is formulated as a coalitional game. We design centralized and distributed protocols for finding a core solution to the coalitional game. With SYNERGY in place, selfish mobile nodes are strongly motivated to collaborate with others in the same coalition to improve their respective key generation rate. The efficacy and efficiency of SYNERGY have been confirmed by extensive simulations.

Chapter 4

SECUREFIND: SECURE AND PRIVACY-PRESERVING OBJECT FINDING VIA MOBILE CROWDSOURCING

4.1 Introduction

The loss and recovery of physical *object*s is a significant issue around the world. Here an object can refer to anything valuable such as personal assets, children, elderly with dementia, and pets. For example, about 800,000 US children are reported lost each year [1], 113 cell phones are lost/stolen every minute in the US [2], and 19,000 items are lost every year by New York subway and bus riders [2]. The predominant method for recovering lost objects is through a lost-and-found place, where lost objects are turned in and returned to their owners with proper identification. Many (if not most) lost objects, however, may not be found or turned in, and the object owner may not know which of the possibly many lost-and-found places he should resort to. The recovery rate for lost objects is thus very low. For instance, University of California Police reported only 19.3% of lost items recovered [2]. In addition, the recovery latency of this traditional method may be too long to be useful. As an example, by the time a lost object is found and turned in to an airport office, the object owner may have departed to a different city or country.

The plummeting cost and ultra-low energy consumption of Bluetooth tags make them very promising to revolutionize the lost-and-found service. In contrast to RFID tags, Bluetooth tags can directly communicate with any mobile device with a Bluetooth tag or interface within a long communication range up to 160 ft. Besides, Bluetooth tags can be used continuously for one year without changing the battery

[3, 4] by adopting the Bluetooth Low Energy (Bluetooth LE) technique, and they only cost several dollars which are often negligible in comparison with the value of lost objects. In the lost-and-found context, a cheap and miniature Bluetooth tag can be attached to every valuable object and contain its owner's identification information. Once finding his object missing, the owner can use his mobile device to search for the corresponding tag. If the tag gets queried, it can report its location or sound an alert to be located. There are growing commercial Bluetooth-based products for locating personal assets, such as Tile [3], BlueBee [5], and StickNFind [4]. These attractive products, however, often require that a lost object be sufficiently close to the searching device. For example, BlueBee tags [5] and StickNFind tags [4] support up to 160 ft and 100 ft, respectively. This inherent range limitation makes it infeasible to recover the lost objects far away from their owners.

A promising solution to overcoming the above range limitation is via mobile crowdsourcing, which refers to the practice of obtaining needed services or data by soliciting contributions from many mobile users. The emergence of mobile crowdsourcing is driven by the skyrocketing growth of mobile devices. For example, the number of mobile connected devices such as smartphones, tablets, laptops, eReaders, and Machine-to-Machine (M2M) modules will hit 11.6 billion by 2021, exceeding the world's projected population at that time (7.8 billion) [10]. Ubiquitous mobile devices can jointly sense and interact with the physical world at an unprecedented scale, thus enabling many otherwise infeasible applications [131, 166]. One can imagine a service provider offering the object-finding service. An object owner submits an object-finding request as a tag query to the service provider, which in turn forwards the query to selected mobile users referred to as *mobile detectors* hereafter. Every detector then locally broadcasts the query. The tag on the lost object responds to any such query, and the corresponding detector finally sends the tag response and his

own location via the service provider to the object owner. Every mobile detector can be rewarded at a fixed rate or in commensurate with the object value. Although the object owner may have to pay for the service, he can recover his valuable object with overwhelming probability.

Crowdsourcing the lost-and-found service faces some great challenges. First, the object in search may be of high value so that the mobile detector discovering it may want to keep it instead of reporting its whereabout to the service provider. Thus we need to alleviate the security concerns of the owners about their lost objects. Second, mobile users may be unwilling to disclose their locations which may indicate too much personal information. Therefore, we must protect the location privacy of mobile users to stimulate their participation in the lost-and-found system. Last, both Bluetooth tags and mobile devices are resource-constrained, so the object-finding process should be very efficient in computation and communication, especially for energy-constrained mobile detectors [64]. Although some companies such as Tile [3] and BlueBee [5] are offering the crowdsourced lost-and-found service, they ensure neither object security nor location privacy of involved mobile detectors.

This chapter presents SecureFind, a crowdsourced object-finding system that offers strong object security to the object owner and also location privacy to mobile detectors. The essential idea in SecureFind is to let some mobile detectors generate dummy tag responses which are indistinguishable from the real tag response in the eye of the service provider and other mobile detectors. Only the object owner can identify the real tag response, so strong object security can be ensured. In addition, the location of each mobile detector discovering the lost object is kept from the service provider and only disclosed to the object owner under a dynamic pseudonym. So the location privacy of mobile detectors can be well guaranteed.

Our contributions are mainly threefold. First, we are the first to formulate secure and privacy-preserving object finding via mobile crowdsourcing to the best of our knowledge. Second, we propose two solutions to this problem. The basic scheme provides strong object security at the cost of low efficiency. In contrast, the advanced scheme seeks to achieve a middle ground among object security, location privacy, and energy efficiency. Finally, we thoroughly evaluate the performance of our schemes by theoretical analysis and extensive simulations.

The rest of this chapter is organized as follows. Section 4.2 surveys the most related work. Section 4.3 outlines the system model, the adversary model, our design objectives, and a Framed Slotted ALOHA protocol underlying our design. Section 4.4 illustrates a basic scheme. Section 4.5 presents an advanced scheme. Section 4.6 evaluates the two schemes using simulations. Section 4.7 summarizes this chapter.

4.2 Related Work

Several schemes have been proposed for tracking and locating lost objects. AutoWitness [62] is a personal asset tracking system that uses an embedded tag with inertial sensor to estimate asset's position change and proactively transmit trajectory data to an external server via cellular link to facilitate asset retrieval. In contrast, SecureFind depends on low-cost Bluetooth tags without any inertial sensor or cellular communication capabilities, thus more suitable for wide adoption. Moreover, Sherlock [115] is a system designed to localize objects with embedded RFID tags in some closed space, which cannot be applied to find lost object in outdoor and is thus orthogonal to SecureFind.

Recent years have witnessed significant research on missing-tag detection [67, 80, 100, 101, 102, 111, 143, 144, 171] and identification [88, 163, 170] in RFID systems. This line of work aims to quickly detect whether or which tags are missing in a large

RFID system, while SecureFind targets a totally different problem. In particular, a lost tag in SecureFind is a tag lost by its owner but still in the SecureFind service provider's service region, and SecureFind aims to determine which mobile detector has the lost tag in his coverage in order to locate and retrieve the lost object without revealing such information to either the mobile detector or service provider. In contrast, a missing tag in [67, 80, 88, 100, 101, 102, 111, 143, 144, 163, 170, 171] means a tag taken away from the monitored area, and the goal there is to determine if any tag is missing. Therefore, existing missing-tag detection schemes are inapplicable to our problem.

Also related is the line of work on privacy-preserving tag identification and authentication in RFID systems, e.g., [19, 89, 97, 145, 155, 157]. These schemes allow efficient identification and authentication of an RFID tag without disclosing any information that can be used to uniquely identify the tag. All the RFID tags belong to the same administrator, and there is no attempt to hide the locations of the RFID tags from the administrator. In contrast, each Bluetooth tag in SecureFind belongs to the corresponding object owner, and its location should be protected from the service provider as well. Therefore, SecureFind differs significantly from these schemes in its aim and scope.

Protecting location privacy in crowdsourcing system is also loosely related to our work. In [147], the authors proposed a novel privacy-preserving framework for spatial crowdsourcing, which allows the service provider to assign spatiotemporal tasks to crowdsourcing workers without sacrificing their location privacy. In addition, Pournajaf *et al.* [122] studied the privacy-preserving spatial task assignment in which crowdsourcing workers obfuscate their locations using spatial cloaking technique. Although both [147] and [122] considered location privacy of crowdsourcing workers,

their problems are completely different from ours, and their solutions are not directly applicable.

4.3 Preliminaries

4.3.1 System Model

We assume a SecureFind service provider offering the object-finding service via mobile crowdsourcing. The service provider fulfils every object-finding request through a number of mobile users referred to as *mobile detector*s hereafter. Every detector has a mobile device such as a smartphone or tablet to communicate with the service provider and also nearby Bluetooth tags. Almost all mobile devices are having the Bluetooth functionality, and it has been shown in [156] that Bluetooth devices can communicate with each other without explicitly establishing a connection. In addition, nearby mobile detectors can communicate via WiFi-direct, Frequency Hopping, or other available Device-to-Device (D2D) technologies which are widely used in many other applications [135, 139, 142, 156, 165].

An object owner refers to a person who lost a valuable object. We assume that the lost object is attached with a Bluetooth tag hard to remove without breaking the object and use "lost tag" and "lost object" interchangeably henceforth. A Bluetooth tag is a small piece of device with an on-board battery, which can perform simple computation and communicate with nearby mobile devices via Bluetooth. Several off-the-shelf Bluetooth tags are currently commercially available for personal asset tracking, such as Tile [3], StickNFind [4], and BlueBee [5] tags. The cost of a Bluetooth tag is currently around a few dollars [3] and is plummeting due to rapid technological advance and growing market demand. It is thus reasonable to assume that every high-value object will be attached with a Bluetooth tag to enable object

finding in the near future. Moreover, we assume that every tag i has a unique ID ID_i known only to its owner.

The object-finding service in SecureFind works as follows. Assume that the object owner knows that his lost object is likely in a possibly large *target area*, e.g., lower Manhattan. He submits to the service provider an object-finding request containing some information about the lost tag and also the target area. The service provider then forwards the object-finding request to all mobile detectors in the target area, each of which in turn locally broadcasts the request. The lost tag responds to any object-finding request intended for it. Every detector hearing a tag response forwards it and his own location via the server to the object owner. Based on the tag responses, the object owner can derive an approximate location (area) of his lost object, e.g., by multilateral triangulation. Finally, the object owner can go to the derived location and send a tag query in person, in which case the lost tag can respond with its GPS location like a StickNFind tag [4] or sound an alert like a Tile [3] or BlueBee [5] tag. During this process, the object owner may initiate multiple requests to keep track of the dynamic locations of his lost tag (object) which may be carried and in motion. All the system operations are automatically executed without user involvement through an SecureFind app installed on each mobile device.

Sound incentives must be provided to all the involved parties to materialize SecureFind. The service provider can either charge the object owner at a rate commensurate with the object value, and it may also provide free services and profit by web advertisement when its service goes very popular. Every mobile detector can be rewarded either at a fixed rate or in accordance with the object value. Such rewarding mechanisms as perks or badges have been proved to be very successful in soliciting mobile users for crowdsourcing applications like Foursquare. The object owner may need to pay for the service, but he will be able to quickly recover his lost object of

high value. Here we assume the existence of such incentive mechanisms and refer readers to existing rich literature such as [154, 168] for incentive design for mobile crowdsourcing.

4.3.2 Adversary Model

We assume that the service provider is honest-but-curious (HBC) [60], which is a widely adopted assumption for rational service providers. In particular, the service provider is trusted to faithfully follow the protocol execution, but it may have interest in the location of the lost object and also the locations of mobile detectors. In addition, the service provider does not collude with any object owner or mobile detector.

Mobile detectors are curious and also location-sensitive. By curious, we mean that mobile detectors try to locate the lost object and take it away prior to the object owner's arrival. To do so, mobile detectors may attempt to infer whether the lost object is in their vicinity from the information they receive during protocol execution. By location-sensitive, we mean that mobile detectors do not want any party (including the server) to know their accurate locations or equivalently linking their accurate locations to their real IDs.

How to deal with other possible attacks on SecureFind is beyond the scope of this chapter. For example, an attacker may jam all radio transmissions, replay intercepted messages, and/or inject bogus messages. Such denial-of-service attacks can target any wireless/mobile system like SecureFind and can be mitigated by existing anti-jamming communication techniques and message authentication.

4.3.3 Design Objectives

We have the following major design objectives.

- *Correctness*: The object owner should be able to obtain an approximate location of the lost object as long as it is within the transmission range of at least one mobile detector.

- *Object security*: The location of the lost object should be known to the object owner only. Strong object security means that the reported data from detectors that have the lost object in their coverage and those not are indistinguishable, such that no mobile detector can infer whether the lost object is within its coverage.

- *Location privacy*: The mapping between the real ID and location of every mobile detector should be kept from any other party.

- *Efficiency*: The object-finding process should incur low communication and computation overhead.

Note that we do not intend to guarantee the recovery of the lost object, as it depends on whether the lost object is covered by at least one mobile detectors and further the density of mobile detectors in the target area. When the lost object is outside of mobile detectors' coverage, neither SecureFind nor any of the existing systems [3, 4, 5] would be able to recover the lost object.

4.3.4 Framed Slotted ALOHA Protocol

Our schemes depend on Framed Slotted ALOHA, which is a popular anti-collision MAC protocol adopted by many RFID systems [88, 126, 143, 144, 162]. Since Bluetooth tag is much more powerful than RFID tag, it is reasonable to assume that Bluetooth tag can support Framed Slotted ALOHA with minimal modification. In SecureFind, Framed Slotted ALOHA is executed between one mobile detector and

a number of nearby Bluetooth tags and works as follows. First, the mobile detector broadcasts a request with two parameters $\langle r, f \rangle$, where r is a random number, and f is the number of time slots in one frame where the f slots are numbered from 0 to $f - 1$. Upon receiving the request $\langle r, f \rangle$, each tag i responds in slot $h(ID_i||r)$ mod f, where ID_i denotes the unique ID of tag i, and $h(\cdot)$ denotes a publicly known hash function. Each of the f time slots can then be an *empty* slot without any tag response, a *singleton* slot with a single tag response, or a *collision* slot with more than one tag responses.

4.4 A Basic Scheme

In this section, we present a basic scheme for secure and privacy-preserving object finding. The essential idea is to let some mobile detectors in the target area act as *dummy tags* to send dummy tag responses for concealing the real tag response. Since the mobile detectors near the lost object cannot differentiate between real and dummy tag responses, the security of the lost object can be well protected. The major design challenge here is how to let the object owner discover the mobile detectors close to his lost object without drawing the attention of these mobile detectors or the service provider.

We propose an iterative multi-round protocol as a solution. In each round, each mobile detector executes the Framed Slotted ALOHA protocol in Section 4.3.4 and forwards the execution result to the object owner via the service provider. The object owner then excludes some mobile detectors who are unlikely near his lost object according to their execution results. The protocol completes when no more mobile detectors can be excluded. Finally, the object owner retrieves the locations of the remaining mobile detectors from the server provider using some specific cryptographic technique and then infers the location of his lost object. Our scheme ensures that

neither the service provider nor the remaining mobile detectors can learn the location of the lost object.

4.4.1 Scheme Description

The service provider divides its service region into multiple physical zones, and every mobile detector reports the index of the zone in which it resides when it decides to participate in object finding and whenever it moves into a new zone. The choice of zone size represents the tradeoff between the overhead and location privacy of mobile detectors. On the one hand, a large zone size can alleviate the mobile detectors' concerns about their location privacy to stimulate their participation, but some mobile detectors outside of the target area will participate in object finding and thus incur higher communication and computation overhead. On the other hand, a small zone size enables more accurate selection of mobile detectors but allows the service provider to infer mobile detectors' locations and thus jeopardize their location privacy. To strike a good balance, we suggest to divide the service area based on cellular tower's coverage, which does not reveal any additional information beyond what cellular service providers already know about mobile detectors' locations.

To initiate lost-object finding, the object owner submits an object-finding request $\langle H(\widehat{ID}||r), r, \mathsf{PK}\rangle$ and the target area to the service provider, where $\widehat{ID}$ denotes the ID of the lost tag, r is a random seed, $H(\cdot)$ denotes a publicly known cryptographic hash function, and PK is the object owner's public key. We can also replace PK with a public-key certificate to prevent the service provider from changing PK to its own choice.

Upon receiving the request, the service provider finds the set of candidate zones that enclose the target area and forwards the request to all the mobile detectors in the candidate zones. Each mobile detector can determine whether to participate in the

object-finding task according to the sensitivity of his spatiotemporal presence. For example, if a mobile user is present near hospital during working hours, he can choose not to participate in the object-finding task even if the location alone is not sensitive. Each participating mobile detector then locally broadcasts a tag query $\langle H(\widehat{ID}||r), r\rangle$. Here we assume a suitable MAC protocol to resolve potential collisions among mobile detectors; e.g., each mobile detector can wait for some random time before sending the tag query. Every tag seeing such a tag query can check whether it is the intended tag by comparing the hash over its ID and r with the received one, and only the lost tag gets prepared to respond. In addition, each mobile detector returns his location encrypted with PK to the service provider so that the service provider cannot figure out his accurate location. The service provider temporarily buffers these encrypted locations.

The object owner then initiates a polling phase consisting of multiple rounds. Consider round $x \geq 1$ as an example. The object owner sends a polling request $\langle r_x, f\rangle$ via the service provider to each mobile detector, where f denotes the frame length as a fixed system parameter, and r_x is a fresh random seed. Every detector i then locally broadcasts $\langle r_x, f\rangle$. Every other detector hearing the polling request from detector i chooses himself as a dummy tag with probability q, which is a tunable system parameter given by the service provider. Each dummy tag j also generates a random pseudonym ID_j. Let $\mathcal{T}_{x,i}$ denote a set of tags comprising all the dummy tags near detector i and also the lost tag if it hears the polling request from detector i as well. Let $h_1(\cdot), \ldots, h_k(\cdot)$ be k publicly known hash functions, where k is a system parameter. Every tag $j \in \mathcal{T}_{x,i}$ computes k slots to reply, where the αth slot is computed as $\mathsf{s}_{j,x}^{\alpha} = h_\alpha(ID_j||r_x) \mod f$ for all $\alpha \in [1, k]$. During the execution of Framed Slotted ALOHA, every tag j sends a one-bit short response in each of its k computed slots. In the end of round x, detector i obtains a bit vector $\mathsf{V}_{i,x} = \langle v_{i,x}[0], \ldots, v_{i,x}[f-1]\rangle$,

where $v_{i,x}[y] = 0$ if slot y is an empty slot and $v_{i,x}[y] = 1$ otherwise. Note that here we do not differentiate between singleton and collision slots, which would require each tag to reply a long multi-bit response and thus incur higher communication overhead. Then detector i sends its bit vector $\mathsf{V}_{i,x}$ to the object owner via the server.

Assuming that there are totally C mobile detectors in the target area, the object owner receives C bit vectors $\{\mathsf{V}_{i,x}\}_{i=1}^{C}$ in round x. He then checks if any mobile detector can be excluded, which is certainly not in the transmission range of his lost tag. To do so, the object owner maintains a candidate detector set. Let $\mathcal{C}_x$ be the candidate detector set at the beginning of round x, where $\mathcal{C}_1 = \{1, \ldots, C\}$. For each detector $i \in \mathcal{C}_x$, the object owners checks if at least one of the bit positions (or slots) $\{h_\alpha(\widehat{ID}||r_x) \mod f\}_{\alpha=1}^{k}$ in $\mathsf{V}_{i,x}$ is zero (or empty), where $\widehat{ID}$ is the ID of his lost tag. If so, the lost tag is certainly not around detector i, and no dummy tag replied in that slot either. So detector i can be safely removed from $\mathcal{C}_x$. The object owner terminates the polling phase if the number of candidate detectors drops to one or remains unchanged after $\tau \geq 2$ polling rounds, where τ is a system parameter. The latter case occurs when the lost tag lies in the coverage of multiple detectors. Also note that the candidate detector set remains confidential to the object owner, and all the C mobile detectors need to broadcast the polling request and process the responses in each round of the polling phase even if some of them may have been confidentially excluded by the object owner.

Once the polling phase is over, the object owner retrieves the encrypted locations of the remaining candidate detectors from the service provider. Finally, he can derive an approximate range for his lost object based on the decrypted detector locations. We can see that the service provider will know which mobile detectors are not excluded. Since the service provider knows the physical zone each mobile detector resides (instead of his real location), it can deduce that the lost object is in

one of the physical zones of the remaining detectors. There are two ways to alleviate this security concern. First, the object owner can request the encrypted locations of $c \geq 1$ detectors that include both the remaining detectors and some excluded detectors to confuse the service provider. Second, the object owner can execute an efficient Private-Information-Retrieval protocol [22] to retrieve the encrypted locations of the remaining candidate detectors without revealing whose locations are retrieved.

4.4.2 Performance Analysis

Now we analyze the performance of the basic scheme.

Correctness. The basic scheme can guarantee that the object owner obtains an approximate location for his lost object as long as it is within the transmission range of at least one mobile detector. Assume that there are totally N mobile users in a region of area S. Also suppose that the number of mobile detectors in any subregion of area s, denoted by $X(s)$, follows a homogeneous spatial Poisson process with intensity N/S: $\mathsf{Pr}(X(s) = k) = \frac{(Ns/S)^k e^{-Ns/S}}{k!}$. Let R denote the transmission range of the lost tag and also mobile detectors. It is easy to see that the basic scheme is correct with probability $1 - \mathsf{Pr}(X(\pi R^2) = 0) = 1 - e^{-\pi N R^2/S}$.

In addition, the basic scheme may incur false positives, which occur when the lost object is not close to any mobile detector (i.e., the given target area is wrong), but some dummy tags happen to respond just like the lost tag in each round of the polling phase. The object owner thus will be misled to wrong locations. We can estimate the false-positive probability as follows. Consider any of the C detectors in the target area, say detector i, which has on average $c = \lfloor \pi N R^2/S \rfloor$ other mobile detectors in his transmission range and does not have the lost tag $\widehat{ID}$ there. Since each mobile detector acts as a dummy tag with probability q, there are totally cq dummy tags in detector i's coverage. Recall that the lost tag needs to respond in

slots $\{\mathsf{s}_{j,x}^{\alpha} = h_{\alpha}(ID_j||r_x) \mod f\}_{\alpha=1}^{k}$ in round x if hearing a polling request. Assume that the output of every hash function is uniformly distributed in $[0, f-1]$. Then the average number of distinct slots the lost tag needs to respond is given by

$$\mu = \sum_{l=1}^{k} l \times \frac{\binom{f}{l}}{f^k} \ . \tag{4.1}$$

As said, each dummy tag also responds in up to k slots uniformly distributed in $[1, f]$. The probability that no dummy tag responds in a particular slot of the lost tag is given by $(1-1/f)^{kqc}$. For detector i to stay in the object owner's candidate detector set in round x, at least one dummy tag needs to respond in each of the μ distinct slots, which occurs with probability $p_{\text{one}} = \left(1-(1-1/f)^{kqc}\right)^{\mu}$. Assume that the polling phase terminates in t rounds. For the false positive to occur, at least one detector needs to survive all the t rounds, which occurs with probability $1-(1-p_{\text{one}}^{t})^{C}$.

Object Security. The basic scheme offers strong object security. In particular, the information the service provider can obtain during object finding includes the initial object-finding request $\langle H(\widehat{ID}||r), r, \mathsf{PK}\rangle$, the polling results in each round, and from which candidate detectors the object owner requested the location. Since the service provider knows neither ID of the lost tag nor the random pseudonym of each dummy tag, he cannot directly infer which detectors have the lost tag in their coverage from the polling results besides knowing that one of the detectors for which the object owner requested the locations does.

Can the service provider do better? To make quantitative analysis possible, we assume that the average number of tags in each detector's communication range are the same, e.g., *cq*. Under this assumption, the detector with the lost tag in its coverage may observe slightly more non-empty slots than those without during the polling phase. In particular, each detector covering the lost tag, called a real detector hereafter, observes a non-empty slot in each slot with probability $p_1 = 1-(1-$

$1/f)^{(cq+1)k}$, whereas each detector not covering the lost tag, called a fake detector hereafter, does so with probability $p_1' = 1 - (1 - 1/f)^{cqk}$. Although this is only a rough estimate because the number of dummy tags around each mobile detector are most likely different, the service provider may still try to gain some information from the polling results by ranking all the detectors according to the numbers of bit ones in their reported vectors. More specifically, the higher the rank of a detector (i.e., the more bit ones in reported vectors), the more likely the detector is a real one, and vice versa.

Now we analyze the probability distribution of the real detector's rank. Consider a real detector i and a fake detector j in round x as an example. Denote by b_i and b_j the numbers of bit-one positions in their reported vectors $\mathsf{V}_{i,x}$ and $\mathsf{V}_{j,x}$, respectively. Let $u = \min(f, (cq+1)k)$ and $u' = \min(f, cqk)$. The probability that detector i has more bit-one positions than detector j is given by

$$\begin{aligned} p_m &= \mathsf{Pr}(b_i \geq b_j) \\ &= \sum_{z=0}^{u'} \mathsf{Pr}(b_i \geq z) \cdot \mathsf{Pr}(b_j = z) \\ &= \sum_{z=1}^{u'} \sum_{z'=z}^{u} \mathsf{Pr}(b_i = z') \cdot \mathsf{Pr}(b_j = z) \\ &= \sum_{z=1}^{u'} \sum_{z'=z}^{u} \binom{u}{z'} p_1^{z'} (1-p_1)^{u-z'} \binom{u'}{z} {p'}_1^{z} (1-{p'}_1)^{u'-z} . \end{aligned} \tag{4.2}$$

For simplicity, assume that there is only one real detector. The p.d.f. of real detector's rank is then given by

$$\mathsf{Pr}(\text{rank} = r) = \binom{C-1}{r-1} p_m^{r-1} (1-p_m)^{C-r} . \tag{4.3}$$

We can see from Eqs. (4.2) and (4.3) that if the number of dummy tags (i.e., cq) is large, p_1 is very close to p_1'. This means that the real detector will be ranked in the

middle of all the detectors with high probability, and the object security can thus be guaranteed.

In addition, neither true or fake mobile detectors can distinguish the responses from the lost tag and from dummy tags and thus cannot determine whether the lost tag is in its vicinity.

Location Privacy. The basic scheme offers location privacy to mobile detectors. Specifically, each mobile detector can report a physical zone encompassing his location instead of his real location to the service provider to participate in SecureFind. Therefore, the service provider cannot get the accurate location of any detector. Even if the location of every responding detector is disclosed to the object owner, we can hide the real ID of the detector from the object owner by letting the service provider replace the real ID with a dynamic pseudonym. Since the object owner does not collude with the service provider as per our adversary model, the location privacy of every mobile detector is well preserved.

Efficiency. To analyze the communication overhead of the basic scheme, we first derive the expected number t of polling rounds. For any mobile detector not covering the lost tag, the object owner excludes it from the candidate detector set with probability

$$p_e = 1 - p_{\text{one}} = 1 - \left(1 - (1 - 1/f)^{kqc}\right)^{\mu} ,$$

where μ is given in Eq. (4.1). So the object owner can exclude p_{e} fraction of the remaining candidate detectors after each polling round. Assume that the number of candidate detectors drops to one after t rounds. Then we have $Cp_{\text{e}}^t = 1$ and thus

$$t = \lfloor \log_{p_{\text{e}}} \frac{1}{C} \rfloor . \tag{4.4}$$

Each mobile detector sends its encrypted location to the service provider at the beginning, and he also broadcasts a polling request and sends a f-bit vector to the service

provider in each polling round. In addition, since each tag needs to reply k one-bit responses in each round, the total communication overhead incurred by tag responses is about $cktC$ bits. Moreover, the object owner sends one object-finding request and t polling messages. Finally, the object owner retrieves λ encrypted detector locations from the service provider.

As for the computation overhead, each tag (dummy or lost) needs k efficient hash operations in each polling round, leading to $cktC$ hash operations in total. Moreover, each mobile detector performs one public-key encryption, and the object owner needs to carry out one public-key decryption for each non-excluded mobile detector. The most expensive public-key encryptions and decryptions can be done very efficiently on current mobile devices. For example, for the standard Elliptic Curve Integrated Encryption Scheme (ECIES), one point multiplication and two point multiplications are needed for one decryption and one encryption, respectively, and a point multiplication takes less than 7.3 ms on an Android Galaxy Nexus smartphone [54].

4.5 An Advanced Scheme: Selected Polling

The basic scheme provides strong object security. However, in each polling round, each mobile detector needs to send an f-bit vector to the service provider which incurs large communication overhead and low efficiency. In this section, we present an advanced scheme to strike a middle ground between object security and system efficiency.

4.5.1 Basic Idea

The advanced scheme stems from an observation about the basic scheme. Specifically, the response from every detector in each polling round is an f-bit vector. The

object owner excludes some candidate detectors in each round x by checking the bit values at k positions $\{\mathsf{s}_{j,x}^{\alpha} = h_{\alpha}(ID_j||r_x) \mod f\}_{\alpha=1}^{k}$, which we refer to as *real* positions. There are at most k real positions because some modular hash values may be the same. Accordingly, we refer to the rest no less than $f-k$ bit positions as *dummy* positions. The dummy positions can effectively hide the real positions so that the detector with the lost object in its coverage cannot tell. The efficiency can be improved if fewer dummy positions are used in each polling round, and the accompanying cost is that real positions will have a higher chance of exposure.

The advanced scheme implements the above thinking by letting the object owner selectively poll fewer than f bit positions in each round, among which the fraction of real positions is adjusted based on the results in previous polling rounds. Intuitively, the more real positions polled in each round, the fewer polling rounds needed to locate the lost tag, the lower the communication and computation overhead, the higher chance of exposing the lost tag, and vice versa. The challenge is how to characterize the exposure of the lost tag and then properly adjust the fraction of real positions.

What is the impact of polling fewer dummy positions on object security? Consider an arbitrary mobile detector, say i. If detector i has the lost tag in his coverage, he is more likely to observe more non-empty slots than other detectors not covering the lost tag. More specifically, assume that the object owner queries ω out of f bit positions, which consists of $\gamma \geq 1$ real positions and $\omega - \gamma$ dummy positions. Recall that each detector on average has $c = \lfloor \pi R^2 N/S \rfloor$ other detectors in his coverage, each acting as a dummy tag with probability q. If detector i covers the lost tag, the probability that a randomly queried bit position having a one (or equivalently the corresponding

slot is busy) can be estimated as

$$\begin{aligned} p_1 &= (1-(1-1/f)^{cqk})\frac{\omega-\gamma}{\omega}+\frac{\gamma}{\omega} \\ &= 1-(1-1/f)^{cqk}+(1-1/f)^{cqk}\frac{\gamma}{\omega}\ . \end{aligned} \tag{4.5}$$

If the lost tag is outside detector i's coverage, the above probability is $p_1' = 1-(1-1/f)^{cqk}$. It is easy to see that $p_1' < p_1$ for $\gamma \geq 1$. As we normally have $\gamma/\omega > k/f$, the gap between p_1 and p_1' becomes more noticeable in the advanced scheme, leading to lower object security. In addition, the larger γ, the more quickly the object owner ruling out the candidate detectors not covering the lost object, the fewer polling rounds needed, the larger the probability gap, the lower object security, and verse versa.

To strike a balance between object security and system efficiency, we let the object owner maximize the number of real positions in each polling round as long as the polling result (i.e., the ω-bit vector) observed by the detector covering the lost object is *statistically indistinguishable* from the one observed by a detector not covering the lost tag. More specifically, let the null hypothesis be that the ω-bit vector obtained by a detector is generated from the binomial distribution $B(\omega, p_1')$, i.e., the theoretical distribution. We can then test the hypothesis using Pearson's chi-squared test [42] with the test statistics given by

$$\chi^2 = \frac{(p_{\text{ob}}-p_1')^2}{p_1'} + \frac{((1-p_{\text{ob}})-(1-p_1'))^2}{(1-p_1')}\ , \tag{4.6}$$

where p_{ob} is the observed frequency of bit ones, and $p_1' = 1-(1-1/f)^{cqk}$ is the theoretical frequency. Finally, we can compute a p-value from χ^2 using the chi-squared distribution for one degree of freedom, which gives us the probability of observing such difference if the ω-bit vector is generated from $B(\omega, p_1')$.

4.5.2 Scheme Description

The pre-polling phase of the advanced scheme is exactly the same as that of the basic scheme, so we do not repeat it here for lack of space.

As in the basic scheme, the polling phase in the advanced scheme also consists of multiple rounds. Consider round $x \geq 1$ as an example. The object owner sends a polling request $\langle r_x, f, \mathsf{d}_{x,0}, \ldots, \mathsf{d}_{x,\omega-1} \rangle$ via the service provider to each mobile detector, where f denotes the frame length as a fixed system parameter, r_x is a fresh random seed, and $0 \leq \mathsf{d}_{x,0} < \mathsf{d}_{x,1} < \cdots < \mathsf{d}_{x,\omega-1} \leq f - 1$ are the ω bit positions that the object owner intends to poll in round x. These ω bit positions include γ_x real and $\omega - \gamma_x$ dummy positions, and how to choose them will be discussed shortly. Every detector i then locally broadcasts $\langle r_x, f, \mathsf{d}_{x,0}, \ldots, \mathsf{d}_{x,\omega-1} \rangle$. Every other detector hearing the polling request from detector i chooses himself as a dummy tag with probability q which is a system parameter. Let $\mathcal{T}_{x,i}$ denote the set of tags comprising all the dummy tags near detector i and also the lost tag if it is covered by detector i. The Framed Slotted ALOHA protocol is still used to collect tag responses. Every tag $j \in \mathcal{T}_{x,i}$ computes k candidate slots to reply, where the αth slot is computed as $\mathsf{s}_{j,x}^{\alpha} = h_\alpha(ID_j || r_x) \mod f$. Then for each $\mathsf{d}_{x,y}, y \in [0, \omega - 1]$, tag j checks if $\mathsf{d}_{x,y} = \mathsf{s}_{j,x}^{\alpha}$ for some α. If so, tag j knows that it should reply a one-bit response in slot y and keeps silent otherwise. In the end of round x, detector i obtains a ω-bit vector $\mathsf{W}_{i,x} = \langle \mathsf{w}_{i,x}[0], \ldots, \mathsf{w}_{i,x}[\omega - 1] \rangle$, where $\mathsf{w}_{i,x}[y] = 0$ if slot y is an empty slot and $\mathsf{w}_{i,x}[y] = 1$ otherwise. Then detector i sends $\mathsf{W}_{i,x}$ to the object owner via the service provider.

Given totally C mobile detectors in the target area, the object owner receives $\{\mathsf{W}_{i,x}\}_{i=1}^{C}$ in round x. As in the basic scheme, he maintains a set of candidate detectors which initially contain all the C detectors. After receiving $\{\mathsf{W}_{i,x}\}_{i=1}^{C}$, the object owner

eliminates all the detectors from the candidate set $\mathcal{C}_x$ with each having at least one zero at the γ_x real positions in his polling result. The polling phase stops when the number of candidate detectors drops to one or remains unchanged after $\tau \geq 2$ rounds, where τ is a system parameter.

After the polling phase, the object owner retrieves the encrypted locations of $\lambda \geq 1$ detectors that include both the remaining detectors and some excluded detectors from the service provider. Finally, he can derive an approximate range for his lost object based on the decrypted detector locations as in the basic scheme.

4.5.3 Choosing Polling Positions

Now we discuss how to choose the ω_x polling positions $\{\mathsf{d}_{x,j}\}_{j=0}^{\omega-1}$ in each round x.

The first step is to determine γ_x, the number of real positions in round x. We propose to derive γ_x based on the C polling results received in all previous rounds such that the expected polling results in round x are statistically indistinguishable from the results generated from the theoretical binomial distribution $B(\omega, p_1')$. In particular, recall that $\mathcal{C}_x$ denote the set of remaining candidate detectors at the beginning of round x. Let $b_{i,x-1}$ be the number of bit ones in $\mathsf{W}_{i,x-1}$ for all $i \in \mathcal{C}_x$, where we set $b_{i,0} = \lceil (1-(1-1/f)^{cqk})\omega \rceil$. As discussed, the probability of any bit position in $\mathsf{W}_{i,x}$ being one for any detector $i \in \mathcal{C}_x$ not covering the lost object can be derived as $p_{i,1} = 1-(1-1/f)^{cqk}$. Then the object owner tries to find $\gamma_{x,i}$ for each detector $i \in \mathcal{C}_x$, the largest number of real positions can be polled in round x, if detector i covers the lost tag. To do so, the object owner initially set $\gamma_{x,i} = 0$. According to Eq. (4.5), the probability of any bit position in $\mathsf{W}_{i,x}$ being one if detector i covers the lost tag is

$$\hat{p}_{i,1} = (1-(1-1/f)^{cqk}) \cdot \frac{\omega - \gamma_{x,i}}{\omega} + \frac{\gamma_{x,i}}{\omega} \ .$$

He then computes the expected fraction of bit ones in $\mathsf{W}_{i,x-1}||\mathsf{W}_{i,x}$ as $p_{\text{ob}} = \frac{\hat{p}_{i,1}\omega + b_{i,x-1}}{2\omega}$, the corresponding test statistics χ^2, and finally the p-value (denoted by $p_{\text{val},i}$). If $p_{\text{val},i} > p_{\text{thre}}$, where p_{thre} is the threshold chosen by the object owner, he increases $\gamma_{x,i}$ by one and repeats the above process until finding the largest possible $\gamma_{x,i} \leq k$. Finally, he chooses γ_x as the minimum among $\{\gamma_i | i \in \mathcal{C}_x\}$. After determining γ_x, the object owner then constructs $\mathsf{q}_{x,0}, \ldots, \mathsf{q}_{x,\omega-1}$ by randomly choosing γ_x real positions from $\{\mathsf{s}_{j,x}^{\alpha}\}_{\alpha=1}^{k}$ and $\omega - \gamma$ dummy positions. The above process is summarized in Algorithm 1.

4.5.4 Performance Analysis

The advanced scheme is correct with the same overwhelming probability and offers the same level of location privacy to mobile detectors as the basic scheme.

Object Security. Similar to that in the basic scheme, the service provider may rank the detectors based on the number of bit ones in their reported vectors. Since we normally have $\gamma/\omega > k/f$, the gap between p_1 and p_1' is more noticeable in the advanced scheme than that in the basic scheme. We thus expect that the advanced scheme offers lower object security than the basic scheme does. Since the number of real positions queried in each polling round is jointly determined by the previous polling results and p_{thre}, we have not been able to derive the rank distribution of the real detector. Instead, we evaluate the object security of the advanced scheme in Section 4.6.

Efficiency. The communication overhead of the advanced scheme depends on the number of polling rounds. Each mobile detector sends its encrypted location to the service provider at the beginning, and he also broadcasts a polling request and sends a ω-bit vector to the service provider in each polling round. In addition, each tag needs to reply $k\omega/f$ one-bit responses on average in each round, so the total communication

Algorithm 1: Computing γ_x for Round x

input : Bit vectors $\{b_{i,x-1} | i \in \mathcal{C}_x\}$, frame length f, p-value threshold p_{thre}

output: γ_x: the number of real positions in round x

$\gamma_x \longleftarrow \min(k, \omega)$;

foreach $i \in \mathcal{C}_x$ **do**

  $\gamma_{x,i} \longleftarrow 0, p_{\text{val},i} \longleftarrow 1$;

  $p_{i,1} \longleftarrow 1 - (1 - 1/f)^{cqk}$;

  while $p_{\text{val},i} > p_{\text{thre}}$ **do**

    $\hat{p}_{i,1} \longleftarrow (1 - (1 - 1/f)^{cqk}) \cdot \frac{\omega - \gamma_i}{\omega} + \frac{\gamma_i}{\omega}$;

    $p_{\text{ob}} \longleftarrow \frac{\hat{p}_{i,1}\omega + b_{i,x-1}}{2\omega}$;

    $\chi^2 = \frac{(p_{\text{ob}} - p_{i,1})^2}{p_{i,1}} + \frac{((1-p_{\text{ob}}) - (1-p_{i,1}))^2}{(1-p_{i,1})}$;

    Update $p_{\text{val},i}$ according to χ^2 based on chi-square distribution;

    if $p_{\text{val},i} > p_{\text{thre}}$ **then**

      $\gamma_{x,i} \longleftarrow \gamma_{x,i} + 1$;

    else

      $\gamma_{x,i} \longleftarrow \gamma_{x,i} - 1$;

  if $\gamma_{x,i} < \gamma_x$ **then**

    $\gamma_x \longleftarrow \gamma_{x,i}$;

return γ_x;

overhead incurred by tag response is about $ck\omega tC/f$ bits. Moreover, the object owner sends one object-finding request and t polling messages. Finally, the object owner retrieves λ encrypted detector locations from the service provider.

As for the computation overhead, each tag (dummy or lost) needs k efficient hash operations in each polling round, leading to $cktC$ hash operations in total. Because the number of polled real positions in the advanced scheme is smaller than that in the basic scheme, the number of polling rounds is also larger in the advanced scheme, resulting in more hash operations and thus larger tag computation overhead. Moreover, each mobile detector performs one public-key encryption, and the object owner needs to carry out one public-key decryption for each non-excluded mobile detector. As said, such public-key encryptions and decryptions can be efficiently done on modern mobile devices.

Again, since the number of polling rounds is jointly determined by the previous polling results and p_{thre}, we have not been able to derive a closed-form result for the communication and computation overhead of the advanced scheme, which is evaluated via simulations in Section 4.6.

4.6 Performance Evaluation

In this section, we evaluate the proposed schemes via extensive simulations.

4.6.1 Simulation Setting

We consider a square region with a side length of $4,000$m, in which 10,000 mobile detectors are distributed uniformly, or 625 mobile detectors per square kilometer. Such density is approximately one sixth of the population density of the downtown area of Austin, TX [6] or one tenth of that of Portland, OR [7]. We assume that each mobile detector acts as a dummy tag with probability $q = 0.9$, which is a tunable

system parameter. We set the transmission ranges of both mobile detectors and the lost tag 100m, which is the lower bound of the transmission range of Bluetooth Low Energy technique [8]. In addition, we assume that the number of hash functions is 10, and the frame length in Frame Slotted ALOHA is 300. The two parameters can be adjusted to ensure that the ratio between the number of bit-one positions and the frame length is not too close to zero or one. The number of polled positions ω is set to be 15. Larger ω incurs higher communication overhead but less rounds to find the object. Other simulation parameters are summarized in Table 4.1 unless stated otherwise.

Table 4.1: Default Simulation Settings.

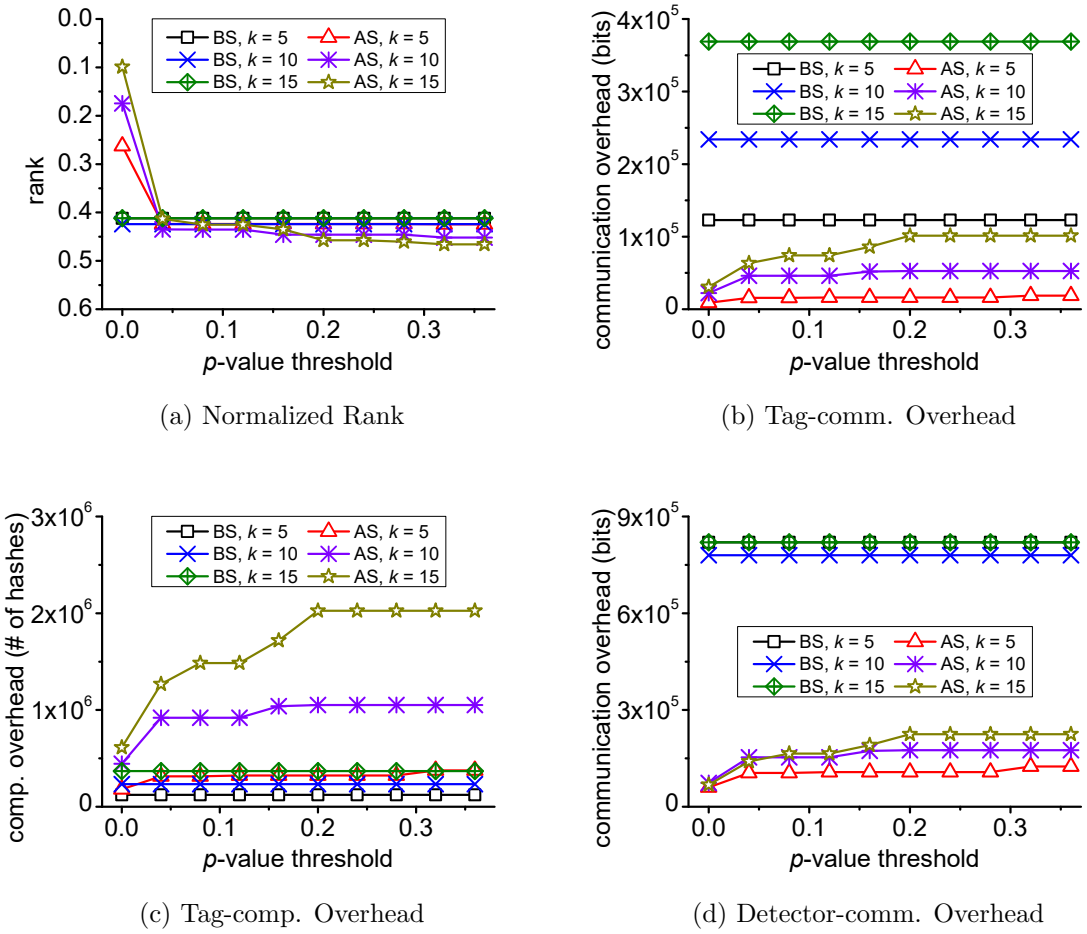
Para.	**Value**	**Meaning**
C	10000	The number of mobile detectors
q	0.9	The probability of acting as dummy tag
f	300	The frame length in Frame Slotted ALOHA
k	10	The number of hash functions
ω	15	The number of polled positions

Since both the basic and the advanced schemes can offer mobile detectors' location privacy and also ensure that the lost object is recoverable almost for sure in all our simulations, our subsequent evaluation focuses on object security, communication overhead, and computation overhead. We assume that the following strategy is adopted by the service provider. On receiving the polling results from all the detectors, the service provider runs the Pearson's chi-squared test as the owner does in the advanced scheme and computes a p-value for each detector. The service provider then ranks all the detectors based on their p-values. The lower the p-value of a detector,

the more likely that the lost tag is in his coverage. We then use the relative rank of the detector covering the lost tag to measure the security of the lost object. If the lost tag is covered by multiple detectors, we use the highest rank available. Note that this strategy is a generalization of ranking collectors according to the numbers of bit-one positions discussed in Section 4.4.2, as it additionally considers the possible different numbers of dummy tags around each collector.

4.6.2 Simulation Results

Impact of p_{thre}. Figs. 4.1a to 4.1d show the object security in terms of the real detector's normalized rank, tag-communication overhead, tag-computation overhead in the number of hash computations performed, and detector-communication overhead of the basic and advanced schemes, respectively. Since the basic scheme is not affected by p_{thre} (the p-value threshold), its performance is plotted for reference only. We can see from Fig. 4.1a that as p_{thre} increases from 0 to 0.3, the real detector's normalized rank under the advanced scheme increases from around 0.1 to 0.4. This is anticipated, as the higher p_{thre}, the fewer real positions polled in each polling round, the smaller the gap between p_1 and p'_1, the lower the rank of the real detector, the higher object security, and vice versa. In addition, we can see from Figs. 4.1b to 4.1d that the tag-communication overhead, tag-computation overhead, and detector-communication overhead of the advanced scheme all increase as p_{thre} increases. The reason is that higher p_{thre} leads to fewer real positions polled in each round and thus more polling rounds needed to locate the lost object. Moreover, the advanced scheme incurs higher tag-computation overhead than the basic scheme, as the advanced scheme requires more polling rounds than the basic scheme and thus each every tag to perform more hash computations. Finally, Figs. 4.1b and 4.1d show that the advanced scheme incurs lower tag- and detector-communication overhead



(a) Normalized Rank

(b) Tag-comm. Overhead

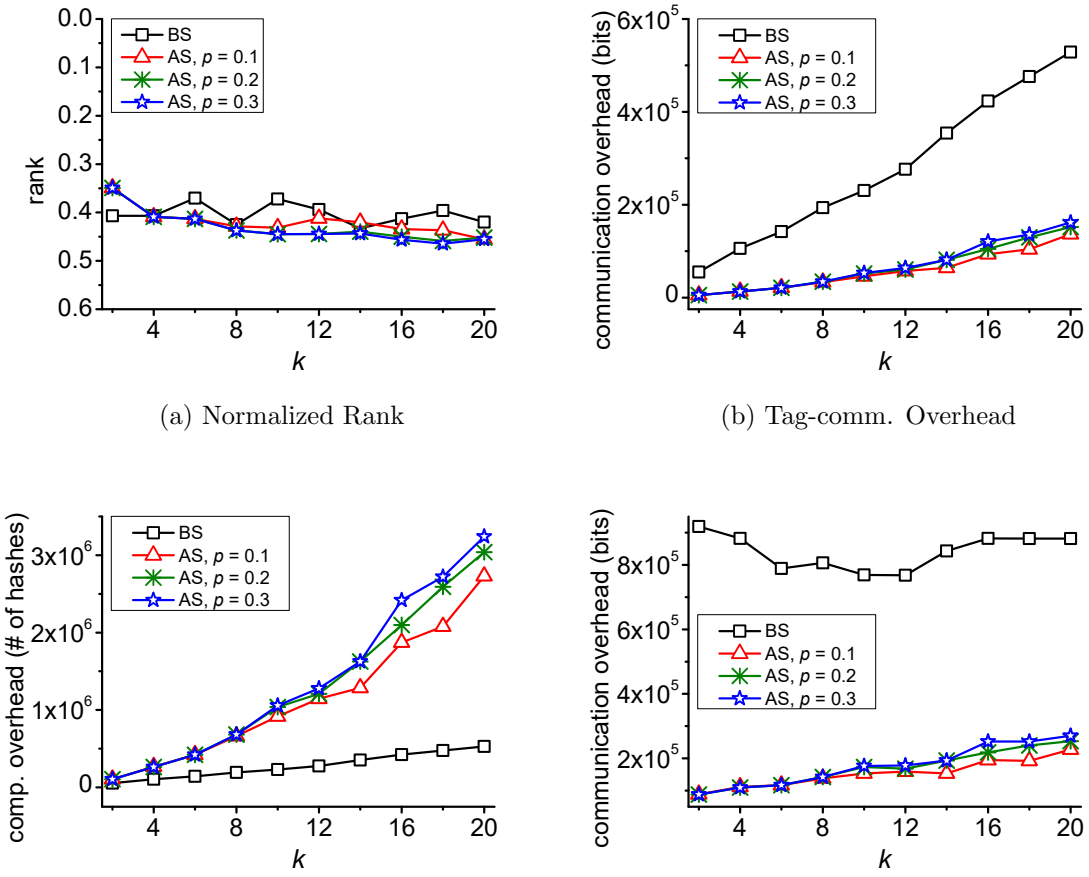
(c) Tag-comp. Overhead

(d) Detector-comm. Overhead

Figure 4.1: Impact of p_{thre}, Where BS and AS Stand for the Basic and Advanced Schemes, Respectively.

than the basic scheme. This is of no surprise because much fewer bits are transmitted from each detector to the service provider in each round under the advanced scheme.

Impact of k. Figs. 4.2a to 4.2d compare the basic and advanced schemes when k (the number of hash functions) varies from 2 to 20. We can see from Fig. 4.2a that the real collector's normalized rank fluctuates as k increases under both schemes. The reason is that the increase in k leads to higher p_1 for the real detector as well as higher p_1' for fake collectors, which nevertheless has little impact on the gap between p_1 and



(a) Normalized Rank

(b) Tag-comm. Overhead

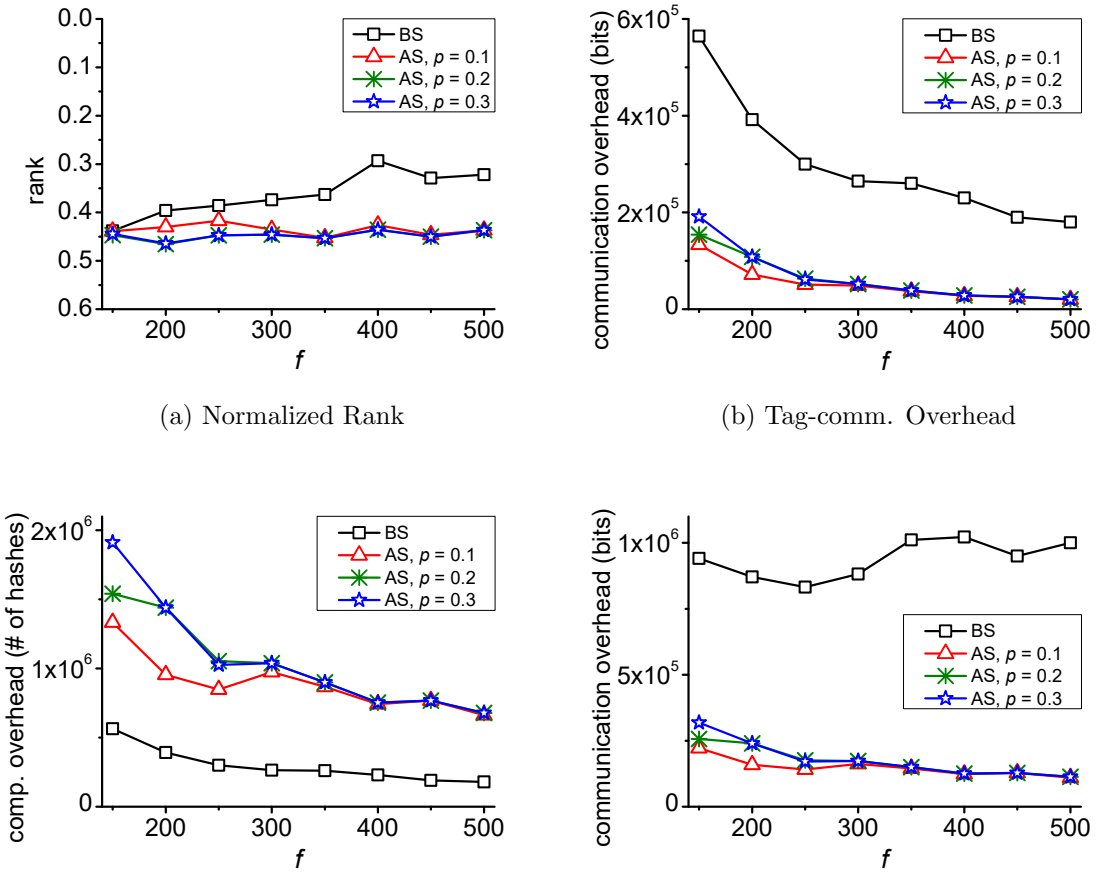
(c) Tag-comp. Overhead

(d) Detector-comm. Overhead

Figure 4.2: Impact of k.

p_1'. In addition, Fig. 4.2b shows that the tag-communication overhead of both schemes increases with k. The reason is that the larger k is, the more slots every tag needs to respond in each polling round, which leads to higher tag-communication overhead. In addition, the advanced scheme incurs much lower communication overhead than the basic scheme, which is expected. Moreover, we can see from Fig. 4.2c that the tag-computation overhead of both schemes increases as k increases and that the advanced scheme incurs higher computation overhead. The reason is that the larger k is, the more hash computations each tag needs to perform in each polling round. In addition,

since we generally have $\gamma < k$ in the advanced scheme, it requires more rounds to locate the lost tag, while every tag needs to perform k hash computations in each round.



(a) Normalized Rank

(b) Tag-comm. Overhead

(c) Tag-comp. Overhead

(d) Detector-comm. Overhead

Figure 4.3: Impact of f.

Impacts of f. Figs. 4.3a to 4.3d show the object security in terms of the real detector's normalized rank, tag-communication overhead, tag-computation overhead in the number of hash computations performed, and detector-communication overhead of the basic and advanced schemes, respectively. Similar to k, f has very limited impact on the normalized rank of the real detector. In addition, we can see from

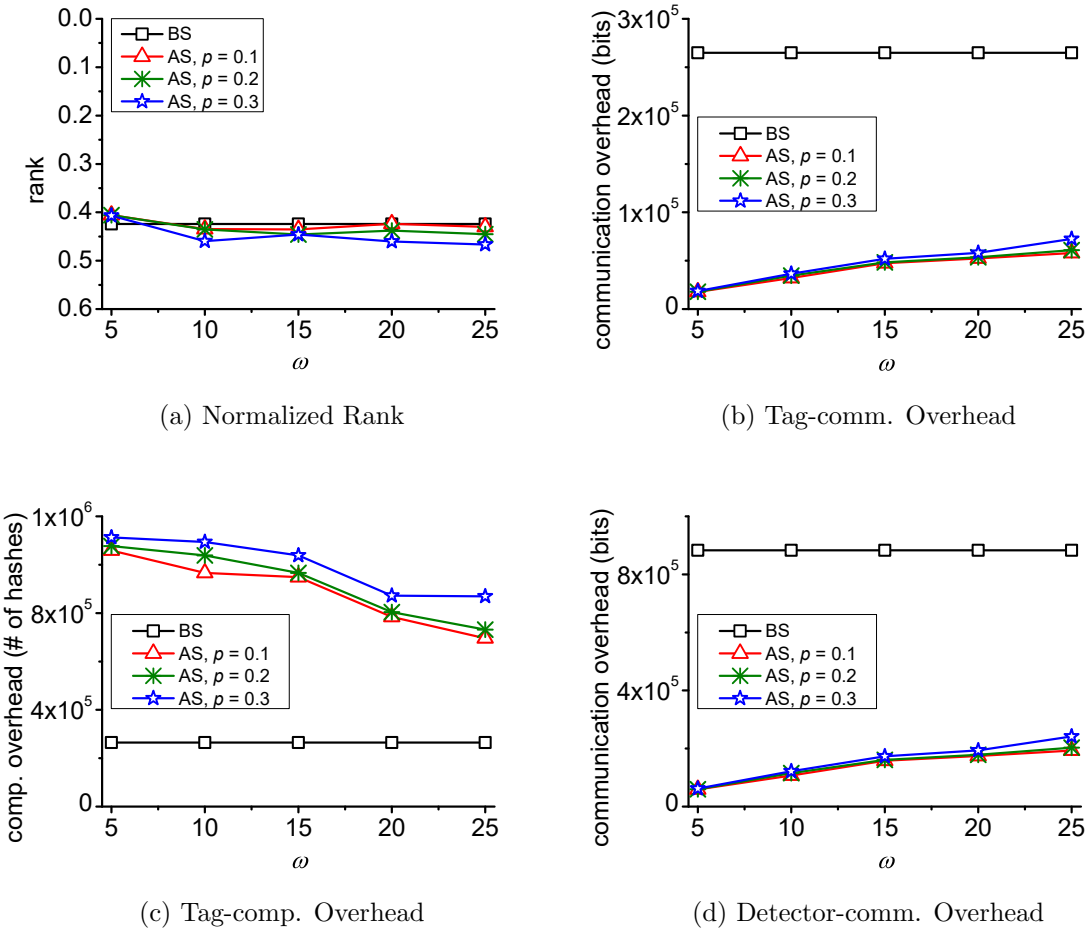
Fig. 4.3b and Fig. 4.3c that the tag-communication and tag-computation overhead of both schemes decrease as f increases. The reason is that the larger f, the fewer polling rounds needed to locate the lost tag, the lower tag-communication and tag-computation overhead for both schemes, and vice versa. In addition, the advanced scheme incurs lower tag-communication overhead but higher tag-computation overhead. Moreover, we can see from Fig. 4.3d that the detector-communication overhead of the advanced scheme decreases as f increases. The reason is that in each polling rounds, each detector needs to transmit a ω-bit vector which is not affected by f. Fewer polling rounds thus lead to lower detector-communication overhead. In contrast, the detector-communication overhead of the basic scheme remains stable as f increases. The reason is that the detector-communication overhead of the basic scheme is the product of the number of polling rounds and the frame length. Since the increase in f leads to the decrease in the number of polling rounds, the detector-communication overhead of the basic scheme is relatively stable.

Impacts of ω. Figs. 4.4a to 4.4d show the impact of ω on the advanced scheme, where the performance of the basic scheme is plotted for reference only. We can see from Fig. 4.4a that ω has very limited impact on the object security. In addition, we can see from Figs. 4.4b to 4.4d that the tag-communication and detector communication overhead both increase and the tag-computation overhead decreases as ω increases.

Impact of mobile detector density. As we mentioned in Section 4.3.3, SecureFind can find the lost object only if it is within the transmission range of at least one mobile detector, which is affected by the density of mobile detectors. Fig. 4.5a shows the impact of C on the probability that the lost object is within the transmission range of at least one mobile detector, i.e., the probability that the lost object can be recovered. As we can see, the probability of the lost object being found increases

as the number of mobile detectors increases, which is expected. In particular, as the number of mobile detectors increases from 2000 to 12000, i.e., the mobile detector density increases from 125 to 750 per square kilometer, the probability of the lost object being found increases from 35% to 90%. We would like to stress that the density of mobile detectors affects only the probability of the lost object being found but not the correctness of SecureFind.

We also evaluated the impact of non-uniform distribution of mobile detectors. In particular, we divided the whole region into 100 equal-size square cells. The mobile



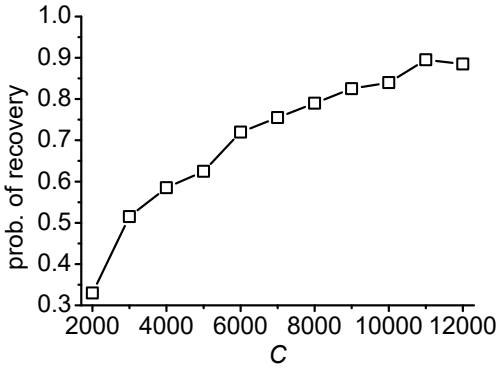
(a) Normalized Rank

(b) Tag-comm. Overhead

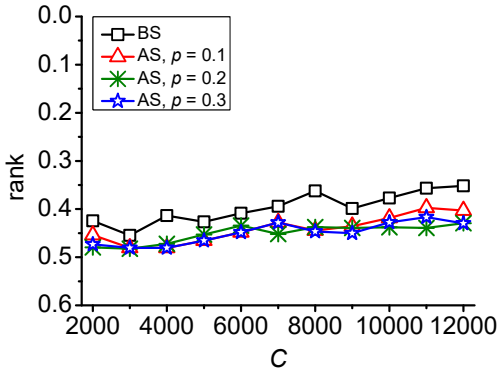
(c) Tag-comp. Overhead

(d) Detector-comm. Overhead

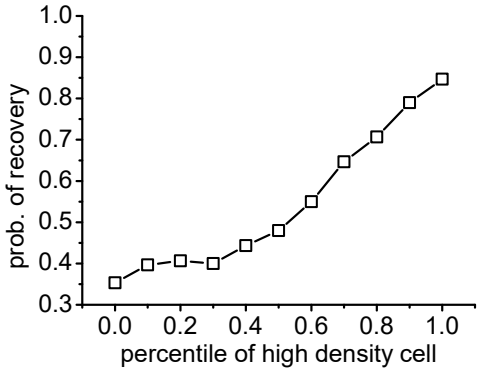
Figure 4.4: Impact of ω.



(a) Probability of Recovery Under Uniform Distribution



(b) Normalized Rank



(c) Probability of Recovery Under Non-uniform Distribution

Figure 4.5: Impacts of C and Non-uniform Detector Distribution.

detector density in each cell is either 20 per cell or 100 per cell, which correspond to low and high density, respectively. Fig. 4.5c shows the probability of the lost object being recovered with the ratio of high density cells from 0 to 1. We can see that the probability of the lost object being recovered increases from 35% to 90% as the ratio of high density cells increases, which is expected.

Fig. 4.5b shows the impact of C on the object security in terms of the real detector's normalized rank in the basic and advanced schemes, respectively, given that the lost object is within the transmission range of at least one mobile detector. We can see that the rank is relatively insensitive to the change in C or mobile detector density, and both the basic and advanced schemes can offer high object security.

Energy consumption. We measured the latency and the energy consumption of hash operation and Bluetooth transmission (i.e., the two major operations in Secure-Find) on two Nexus 7 tablets with Android 4.3. Our experiments show that $7,500,000$ hash operations take 75s and consume 61.25 J of energy on Nexus 7 tablet. This indicates that one hash operation takes 0.01 ms and consumes 8.17×10^{-6} J on average. We measured that the transmission rate of Bluetooth Low Energy is approximately $109 \sim 113$ KB/s, which consumes energy at a rate of $202 \sim 223$ mW. This means that transmitting one bit consumes approximately $2.24 \sim 2.49 \times 10^{-7}$ J of energy.

Based on our measurement results, we further estimate the energy consumption of mobile detector and dummy tag during one object-finding operation. We assume the parameter settings in Table 4.1 where $f = 300$, $k = 10$, $\omega = 15$, and $p = 0.3$. Consider the simulation results shown in Fig. 4.1 as an example. It takes 2.56 rounds on average to find the lost object by adopting the basic scheme. In the basic scheme, each mobile detector needs to transmit $2.56 \times 300 = 768$ bits to the service provider, incurring 7.68×10^{-3} J of energy [1] . In addition, each dummy tag needs to perform

[1] According to [66], the energy consumption of LTE upload link is 1×10^{-5}J/bit.

on average $2.56 \times 10 = 25.6$ hash operations and transmit $2.56 \times 10 = 25.6$ bits, which incur 2.1×10^{-4} J and $5.7 \sim 6.4 \times 10^{-6}$ J of energy, respectively. For the advanced scheme, it takes 10 rounds on average to find the lost object, during which each mobile detector needs to transmit $10 \times 15 = 150$ bits to the service provider and incur 1.5×10^{-3} J of energy consumption. Moreover, each dummy tag needs to perform on average $10 \times 10 = 100$ hash operations and transmit $10 \times 0.5 = 5$ bits, which incur 8.17×10^{-4} J and $1.12 \sim 1.25 \times 10^{-6}$ J of energy consumption, respectively. In general, a typical smartphone's battery stores approximately $15,000 \sim 20,000$ J of energy [9]. Therefore, the operations of SecureFind have negligible impact on a mobile device's battery life.

4.6.3 Discussion

Our above evaluations have shown that both the basic and the advanced schemes can enable object finding while ensuring the security of the lost object and also the location privacy of the mobile users participating in object finding. Now we discuss some additional factors that may impact SecureFind's performance.

Impact of insufficient dummy tags. SecureFind relies on mobile detectors serving as dummy tags to offer object security. If there are insufficient detectors around the lost object to serve as dummy tags, the mobile detector that receives response from the lost object may be able to infer that the lost object is nearby and the object security cannot be guaranteed. However, this is only possible if the malicious mobile detector can distinguish whether the response he receives is indeed from the lost object or dummy tag. Even if there is no dummy tag near the lost object, as long as there are normal people around, a malicious mobile detector would be unable to to tell whether the response is from the lost object, as it is extremely difficult to tell whether any particular person nearby is serving as mobile detector and dummy tag.

In the most extreme case when there is no people around, the malicious detector can determine that the lost object is nearby. We note that in such cases neither SecureFind nor any existing Bluetooth-tag-based scheme such as Tile [3], BlueBee [5], and StickNFind [4] is capable of recovering the lost object, as there lacks honest mobile detector (including the object owner himself) that has the lost object in the transmission range. Since the owner has already lost the object, it makes no difference between the object being recovered by some malicious mobile detector or unknown person. Therefore, SecureFind can help the object owner recover the lost object if the mobile detector density is not extremely low and does not cause any extra damage to the object owner otherwise.

Impact of detector mobility. During the object finding process, some mobile detectors and dummy tags may move into or out of the transmission range of the lost object due to mobile detector's mobility, which may affect the object-finding result in different ways. First, some dummy tags may move into or out of the transmission ranges of the mobile detectors that collect polling result. For any mobile detector that collects polling result, the increase (or decrease) in the number of surround dummy tags will result in the increase (or decrease) in the number of bit-one positions in bit vector at each round, which makes it less (or more) likely for the object owner to filter out fake detectors at the end of object-finding process and thus more (or fewer) false positives. Second, an initially real detector may move out of the transmission range of the lost object before the end of the object-finding process, making the object owner unable to find the lost object via this particular detector. Third, an initially fake detector may move into the transmission range of the lost object before the end of the object-finding process. If the detector is not ruled out by the polling results before the movement, this fake detector become a real detector and would help the owner find the lost object.

We expect that the above events happen rarely in practice in due to the low latency of the polling phase in both the basic and the advanced schemes. In particular, each slot takes 321 µs in Slotted ALOHA according to [101]. Under the parameter settings in Table 4.1, each polling round needs 96.3 ms and 4.8 ms for the basic and advanced schemes, respectively. Take the simulation results shown in Fig. 4.1 as an example, it takes about 250 ms and 48 ms to finish all polling rounds for the basic and advanced schemes, respectively. Since our simulation results show that a single object-finding process takes less than one second in most cases, we expect detector mobility has very limited impact on SecureFind's performance.

4.7 Summary

This chapter presents the design, analysis, and evaluation of SecureFind, the first secure and privacy-preserving crowdsourced object-finding system. In particular, we first introduce a basic scheme which provides strong object security at the cost of system efficiency, and then present an advanced scheme to strike a good balance between object security and system efficiency. Detailed simulations confirm that SecureFind can enable very fast and efficient object finding while ensuring the security of the lost object and also the location privacy of the mobile users participating in object finding.

There are still many open challenges to tackle. For example, in the current design, all the mobile detectors in the target area specified by the object owner need to participate in object finding. Since some of them may have overlapping coverage, there may be significant room for reducing the communication and computation overhead. One possible solution is to let the service provider select the minimum number of mobile detectors that can jointly cover the target area. This solution, however, requires the service provider to know more accurate locations of mobile detectors. Such

tradeoff between system efficiency and location privacy deserves careful investigation. In addition, our current design assumes that mobile detectors are honest-but-curious. There may be dishonest mobile detectors who report fake search results to earn reward without actually performing the object search. How to catch and then punish such dishonest mobile detectors is nontrivial and may conflict with the location-privacy requirement of mobile detectors. We hope that the study in this chapter can stimulate further interest in crowdsourced object finding and other exciting mobile crowdsourcing applications.

Chapter 5

PRISTREAM: PRIVACY-PRESERVING DISTRIBUTED STREAM MONITORING OF THRESHOLDED PERCENTILE STATISTICS

5.1 Introduction

Distributed stream monitoring, which monitors functions over distributed and continuous data streams in real time, has great potential in future smart cities driven by the emerging Internet-of-Things paradigm. For example, with the help of a distributed mobile health monitoring system, a public health authority can monitor the health data collected by each user's mobile and wearable devices to enable various services such as public health condition monitoring, early detection of disease outbreaks, and epidemiology research studies. In addition, a waste management company can achieve cost-efficient trash collection scheduling by monitoring the sensors installed in trash containers. As another example, a utility company (e.g., electricity, natural gas, or water) can improve the efficiency and reliability of its utility infrastructure by gathering fine-grained information from the sensors at consumers' places.

Communication efficiency is a key challenge for a practical distributed stream monitoring system. In particular, the system may contain thousands of distributed sensors, e.g., in future smart-city applications. In addition, the reporting frequency should be high enough, e.g., every five minutes, to enable approximately real-time monitoring and decision making. Since most monitoring sensors are expected to have tight resource constraints, significant effort should be made to minimize their communication overhead for data reporting.

Data privacy is another major challenge for distributed stream monitoring systems. In particular, many monitoring systems rely on sensors affiliated with human users, and the raw sensor data may be sensitive in nature. For example, the data from a biomedical sensor will disclose the user's health conditions, and the data from a utility sensor can enable the profiling of the corresponding consumer's life pattern and routine. Without strong guarantee of their data privacy, users will be reluctant to join distributed monitoring systems.

There are some attempts to achieve the goal of communication-efficient and/or privacy-preserving distributed stream monitoring with aggregation thresholds. In such systems, time is divided into fixed time intervals, and each node records new data generated in each interval whereby to compute a statistic value. The goal of the monitoring service provider is to aggregate all the users' statistic values in each interval and compares the aggregation result with some predefined threshold. Such thresholded monitoring systems have important applications such as anomaly detection. Previous work sought to trade aggregation accuracy for communication efficiency by letting each node independently decide whether its data submission can contribute to the service provider's decision making; if not, the node will not submit his data. The MEAN aggregate function is addressed in [72, 118, 128], SUM and COUNT are considered in [72, 118], and MIN and MAX are addressed in [118]. In addition, the work in [57] incorporates differential privacy guarantees into the scheme in [128].

In this chapter, we study distributed stream monitoring of thresholded PERCENTILE aggregates with high communication efficiency and also strong privacy guarantees. In particular, the monitoring service provider wants to monitor when $f(\chi_r) > \tau$ happens, where χ_r denotes the rth percentile among the statistic values from all the distributed nodes, $f(\cdot)$ denotes an arbitrary single-parameter function

chosen by the service provider (say, a squaring or square root function), and τ denotes a predefined threshold. The rth percentile of a data set refers to the value greater than or equal to $r\%$ of the data values. The PERCENTILE aggregate is much more robust than other statistic metrics such as MEAN, SUM, and MIN/MAX which can be easily manipulated by the data from a single or small set of dysfunctional or compromised nodes.

Our system is designed with three objectives. First, it should be *correct* in the sense that the monitoring service provider can decide when $f(\chi_r) > \tau$ happens with extremely low false positives and negatives. Second, it should be *communication-efficient* such that each node submits its data only when necessary. Last, it should be *privacy-preserving* in keeping individual users' data confidential.

This chapter makes the following contributions.

- We are the first to motivate and formulate the problem of communication-efficient and privacy-preserving distributed stream monitoring for thresholded PERCENTILE aggregates to the best of our knowledge.

- We propose a novel technique for distributed stream monitoring of thresholded PERCENTILE aggregates with high communication efficiency and differential privacy guarantees. In our technique, the monitoring service provider constructs one or several safe (data) ranges based on the desired function $f(\cdot)$ and threshold τ. Each node can independently decide whether his statistic value in a new interval should be submitted based on the safe ranges and his statistic value in last interval. Powered by the differential privacy theory [50], our technique also ensures that each node's submitted data are not substantially different if one element of the node's data stream changes. Differential privacy guarantees can effectively prevent the monitoring service provider or any other

internal/external adversary with arbitrary background knowledge from identifying the actual content of any particular data stream to breach the privacy of the corresponding node (user).

- We thoroughly evaluate the accuracy, communication efficiency, and privacy guarantees of our system through theoretical analysis and detailed simulation studies. Our results show that PriStream significantly reduces communication overhead and maintains differential privacy simultaneously.

The rest of this chapter is organized as follows. Section 5.2 briefs the related work. Section 5.3 introduces the system and adversary models. Section 5.4 outlines the background on differential privacy. Section 5.5 details our system design and analyzes its performance. Section 5.6 evaluates our system through detailed MATLAB simulations based on both real-world and synthetic datasets. Section 5.7 summarizes this chapter.

5.2 Related Work

A large chunk of work [17, 20, 44, 57, 58, 72, 118, 128] studies communication-efficient monitoring of distributed streams. A wide range of aggregate functions sought by the monitoring service provider have been covered, including SUM and COUNT [72, 118], inner products [44], and entropy [20], as well as MEAN and MIN/MAX in [118]. The work [17] aims to achieve efficient detection of distributed constraint violations. In addition, a novel geometric approach is proposed in [128] for monitoring threshold functions over distributed data streams. In this approach, a global monitoring task is decomposed into a set of geometric constraints applied locally in each node for deciding whether to submit the data. This geometric approach has been adopted by others for achieving various distributed stream monitor-

ing goals [58, 73, 81]. Although elegant, these schemes cannot be applied to enable communication-efficient distributed stream monitoring of thresholded PERCENTILE aggregates.

Significant efforts have been made on privacy-preserving aggregation for distributed time-series data and/or providing differential privacy for individual data streams [15, 28, 36, 38, 53, 69, 70, 87, 124, 129, 151]. The PASTE algorithm in [124] targets historical time-series data and requires the pre-processing of all possible query results, so it cannot be applied for distributed real-time monitoring tasks. In addition, the algorithm in [129] enables an untrusted aggregator to compute the sum of distributed time-series data with differential privacy guarantees to all the data sources. This algorithm cannot be directly applied to distributed stream monitoring of thresholded PERCENTILE aggregates. Moreover, the framework in [57] enables monitoring arbitrary threshold functions over the MEAN aggregate of the statistics from distributed data-stream sources in a differentially privacy-preserving fashion. In addition, references [15, 28, 36, 69, 70, 151] further offer fault tolerance against sensor failure. All these work focuses on additive aggregation and thus cannot be applied to our problem. In contrast, our PriStream system is the first work targeting differentially privacy-preserving distributed stream monitoring of thresholded PERCENTILE aggregates.

Privacy-preserving data aggregation is also studied in mobile sensing and wireless sensor networks [65, 71, 85, 86, 130, 164]. The work [65, 130, 164] addresses privacy-preserving data aggregation by data slicing and mixing, but these schemes involve cooperation among peer nodes and does not apply to our scenario where sensor nodes work independently. Li *et al.* studied privacy-preserving MIN [86] and SUM [94] aggregations in mobile sensing systems, and these schemes cannot be ap-

plied to thresholded PERCENTILE aggregations. In addition, no differential privacy is guaranteed in [65, 71, 86, 130].

5.3 System and Adversary Models

5.3.1 System Model

We use a widely adopted model [57, 72, 118, 128, 137, 166] which consists of a service provider and k nodes denoted by $n_1, n_2, \cdots, n_k$. Affiliated with a human user or organization, each node n_i continuously performs the predetermined sensing task and can directly communicate with the service provider to submit data or receive instructions. In addition, unlike [135, 139], PriStream does not require communications or collaborations among the nodes.

We make the following assumptions for distributed stream monitoring of thresholded PERCENTILE aggregates. Time is divided into equal-length intervals, denoted by t_l for $l \in [1, \infty)$, and each node may generate new data items in each interval. Let $S_i = \{d_{i,1}, d_{i,2}, \cdots\}$ denote the data set of node n_i from the beginning, where $d_{i,l}$ for $l \in [1, \infty)$ refers to the lth data item in the data domain $\mathcal{D}$. In addition, we use $S_i(t_l) \subseteq S_i$ to denote the data items node n_i generated in interval t_l. In interval t_l, each node n_i can compute a statistic value decided by the service provider as $v_i(t_l) = g(S_i(t_l)) \in \mathcal{R}$, where $g(\cdot)$ is a publicly known function that generates statistic value based on the input data set. For example, $g(\cdot)$ can be the mean, average, count, or any other function.

The service provider aims to monitor whether the global condition $f(\chi_r(t_l)) > \tau$ holds in each interval j. Here $f(\cdot) : \mathcal{R} \to \mathcal{R}$ refers to an arbitrary single-parameter function chosen by the service provider; $\tau \in \mathcal{R}$ is the predetermined monitoring threshold; and $\chi_r(t_l)$ denotes the rth percentile of the statistic values from k sensor

nodes. There is no universal definition for the rth percentile, and we adopt the nearest rank method for its simplicity. In particular, we first sort the k data values in the ascending order. $\chi_{r,j}$ is the smallest value in the list such that r percent of the data values is no larger than that value. More specifically, $\chi_r(t_l)$ is the value at position $\lceil rk/100 \rceil$ of the ordered list. Whenever the global condition is satisfied, the service provider takes corresponding actions such as broadcasting public safety alarms.

5.3.2 Adversary Model

The adversary can be internal to PriStream. An internal attacker can be the PriStream service provider, which is assumed to be honest-but-curious in the sense that it faithfully performs the system operations but is interested in the raw data of distributed nodes. This assumption is commonly adopted for system operators in the literature. An internal attacker can also be any distributed PriStream node which is curious about other nodes' raw data. In addition, the PriStream node can be honest by submitting real sensing data or malicious by reporting fake data. We assume that malicious PriStream nodes are the minority so that PriStream is always functional.

We also consider external attackers interested in the raw data of PriStream nodes to breach their privacy. An external attacker may compromise some PriStream nodes to be come internal attackers, but we assume that compromised nodes if any are the minority.

There can be collusion among internal attackers alone, external attackers alone, or internal and external attackers. We make a reasonable assumption that the number of attackers involved in a collusion is much smaller than the number of PriStream nodes which can be in thousands or more.

5.4 Preliminaries on Differential Privacy

Differential privacy [50] is a recently proposed privacy model which guarantees strong privacy. It originally comes from the database discipline and has been applied in many other related areas [57]. In what follows, we first introduce the definition of ϵ-differential privacy and its properties. Then we outline two schemes to achieve ϵ-differential privacy.

Definition 5.4.1. ***(Adjacent Streams [52]).*** *Two streams S_i and S_i' of n_i are defined as adjacent streams iff there exist $d, d' \in \mathcal{D}$ such that replacing d in S_i with d' will result in S_i'.*

Definition 5.4.2. ***(ϵ-Differential Privacy [50]).*** *A randomized algorithm* Alg *provides ϵ-differential privacy iff for any adjacent streams S_i and S_i' and any set O of possible outputs,*

$$\mathbf{Pr}[\mathsf{Alg}(S_i) \in O] \leq \mathbf{Pr}[\mathsf{Alg}(S_i') \in O] \times e^{\epsilon}, \tag{5.1}$$

where the probability is taken over the randomness of Alg.

The above definition means that a differentially private algorithm Alg will generate the same output over two streams with only one different element with almost the same probability. In general, ϵ is positive, and the smaller ϵ is, the stronger the differential privacy.

Definition 5.4.3. ***(ℓ_ρ-Sensitivity [51]).*** *The ℓ_ρ-sensitivity of a function $g : S_i \rightarrow \mathcal{R}$ is defined as*

$$\Delta_\rho(g) = \max_{S_i \approx S_i'} ||g(S_i) - g(S_i')||_\rho, \tag{5.2}$$

where S_i and S_i' are two adjacent streams of node n_i which only differ in one element.

Composition properties. Differential privacy maintains a sequential composition property. In particular, a sequence of computations that each provides differential privacy independently also guarantee differential privacy, and the privacy cost of each computation is accumulated. For example, a sequential differentially private computation conducted by algorithms $\mathsf{Alg}_1, \mathsf{Alg}_2, \ldots, \mathsf{Alg}_n$, each with a privacy cost $\epsilon_1, \epsilon_2, \ldots, \epsilon_n$, respectively, can be processed as long as its privacy cost ϵ is greater or equal to $\sum_{i=1}^{n} \epsilon_i$.

Laplace [51] and exponential [108] mechanisms are commonly employed to achieve ϵ-differential privacy.

Definition 5.4.4. ***(*Laplace Mechanism** ***[51]).*** *This mechanism is designed for real-valued outputs, and it directly adds noise drawn from a Laplace distribution to each original output value to achieve ϵ-differential privacy. More specifically, given a function $g : S \rightarrow \mathcal{R}$, the Laplace mechanism is defined as $g'(S) = g(S) + \mathsf{Laplace}(\Delta_1(g)/\epsilon)$, where $\mathsf{Laplace}(\lambda)$ for any λ denotes a Laplace distribution with probability density function $\mathbf{Pr}(x|\lambda) = \frac{1}{2\lambda} e^{\frac{-|x|}{\lambda}}$.*

Definition 5.4.5. ***(*Exponential Mechanism** ***[108])*** *This mechanism applies when target outputs are not real values or cannot be added with noises. An example is to sample one of several options while considering the desirability of each option. In particular, given a utility function $h : (\mathcal{D} \times O) \rightarrow \mathcal{R}$ which assigns a score to each output $r \in \mathcal{R}$, the exponential mechanism M which chooses an output $r \in \mathcal{R}$ based on a stream S_i of node n_i is defined as*

$$\mathsf{M}(S_i, h) = \left\{ r \text{ with probability} \propto \exp\left(\frac{\epsilon h(S_i, r)}{2\Delta_1(h)}\right) \right\}. \tag{5.3}$$

5.5 PriStream Design

In this section, we elaborate on the design of PriStream.

5.5.1 Overview

The most intuitive method for monitoring whether global condition $f(\chi_r(t_l)) > \tau$ holds in every interval is to let each node n_i report its statistic value $v_i(t_l) = g(S_i(t_l))$ to the service provider, which can in turn decide $\chi_r(t_l)$ and test whether the global condition holds.

The above method has two obvious limitations. First, letting every node report its statistic value in every interval incurs significant communication overhead, especially if the reporting frequency need be sufficiently high (say, every five minutes) for real-time decision making. Since most sensor nodes in future smart cities are expected to have limited energy, their batteries will be quickly drained out and very difficult to replenish. Second, the service provider can learn the original data of all the nodes and thus violate the privacy of the corresponding users.

To address the first limitation, we propose a novel ranging technique to enable communication-efficient distributed monitoring. Specifically, we observe that testing whether the global condition $f(\chi_r(t_l)) > \tau$ holds does not require the service provider to know the actual value of $\chi_r(t_l)$. Instead, it suffices to know whether $\chi_r(t_l)$ falls into the range where $f(\chi_r(t_l)) > \tau$ holds. Recall that $\mathcal{R}$ is the domain of the statistic value at each node. It follows that $\chi_r(t_l) \in \mathcal{R}$ for every interval as well. Given function $f(\cdot)$ chosen by the service provider, we can divide $\mathcal{R}$ into a safe area $\mathcal{R}^+(t_l)$ and a unsafe area $\mathcal{R}^-(t_l)$, such that $\mathcal{R} = \mathcal{R}^+(t_l) \bigcup \mathcal{R}^-(t_l)$, $\mathcal{R}^+(t_l) \bigcap \mathcal{R}^-(t_l) = \emptyset$, and $f(\chi_r(t_l)) > \tau$ if and only if $\chi_r(t_l) \in \mathcal{R}^-(t_l)$. The global monitoring task can then be converted into testing whether $\chi_r(t_l) \in \mathcal{R}^-(t_l)$, which can be accomplished without knowing the actual value of $\chi_r(t_l)$ in every interval.

More specifically, for a given function $f(\cdot)$, the safe range $\mathcal{R}^+(t_l)$ and the unsafe range $\mathcal{R}^-(t_l)$ may each comprise multiple disjoint ranges. Without loss of generality,

assume that $\mathcal{R}^+(t_l)$ and $\mathcal{R}^-(t_l)$ together comprise θ disjoint ranges $R_1(t_l), \ldots, R_\theta(t_l)$, where $\bigcup_{i=1}^{\theta} R_i(t_l) = \mathcal{R}$, $R_i(t_l) \bigcap R_j(t_l) = \emptyset$ for all $i \neq j$, and each $R_j(t_l)$ is either an open or closed range with left and right boundaries $\mathrm{l}_j(t_l)$ and $\mathrm{r}_j(t_l)$, respectively. Let $k_j(t_l)$ be the number of nodes with statistic values in R_j in interval t_l for all $j \in [1, \theta]$. It follows that $k = \sum_{j=1}^{\theta} k_j(t_l)$ for every interval $t_l = 1, 2, \ldots$. In every interval t_l, each node n_i reports to the service provider the index of range that its statistic value $v_i(t_l)$ falls into, which allows the service provider to compute $k_1(t_l), \ldots, k_\theta(t_l)$, and further determine which range $\chi_r(t_l)$ falls into and whether $f(\chi_r(t_l)) > \tau$ holds.

To tackle the second limitation, PriStream adopts the Laplace and the exponential mechanisms to provide differential privacy to participating nodes. Consider node n_i with statistic value $v_i(t_l) \in R_x(t_l)$ in interval t_l as an example. To ensure differential privacy, node n_i reports a perturbed interval index x' generated via a combination of the Laplace and the exponential mechanisms.

5.5.2 Detailed PriStream Operations

We now illustrate the detailed PriStream operations, which comprise initialization and communication-efficient phases.

Initialization Phase

The service provider starts the initialization phase at the end of interval t_l, where t_l refers to either the first interval t_1 or any subsequent interval (i.e., $l \geq 2$) for which the global monitoring condition has been changed in the preceding interval t_{l-1}. The purpose is to assign the range parameters to all nodes and collect each node's range index.

Operation Details

At the end of interval t_l, the service provider issues a system-wide query, which specifies the desired statistic metric generation function $g(\cdot)$, the precomputed disjoint ranges $\{R_j(t_l)\}_{j=1}^{\theta}$ with corresponding left and right boundaries $\{\langle \mathrm{l}_j(t_l), \mathrm{r}_j(t_l)\rangle\}_{j=1}^{\theta}$, the differential-privacy parameter ϵ.

Upon receiving the query, each node n_i for $\forall i \in [1, k]$ with its data vector $S_i(t_l)$ does the following in sequel.

1. Compute the desired statistic value $v_i(t_l) = g(S_i(t_l))$.
2. Find the real range index $x \in [1, \theta]$ such that $v_i(t_l) \in R_x(t_l)$.
3. Compute a perturbed range index $I_i(t_l)$ from x according to Alg. 2.
4. Send $I_i(t_l)$ to the service provider.

After receiving all the perturbed range indexes, the service provider processes them as follows.

1. Count the number of perturbed range indexes being j as $k_j(t_l)$ for every $j \in [1, \theta]$.
2. Calculate the percentile value $P_j(t_l)$ at the range $R_j(t_l)$'s left boundary $\mathrm{l}_j(t_l)$ as $\sum_{i=j}^{\theta} k_i(t_l)/k$ for every $j \in [1, \theta]$.
3. Determine the range $R_x(t_l)$ in which $\chi_r(t_l)$ falls into by finding x such that $P_x > r/100 > P_{x+1}$.
4. Check whether $R_x(t_l)$ is a safe range. If not, take the predetermined action such as issuing a public safety alarm. Otherwise, keep silent. For both cases, proceed to communication-efficient phases.

Algorithm 2: Generating Perturbed Range Index

input : $\{(\mathrm{l}_i(t_l), \mathrm{r}_i(t_l))\}_{i=1}^{\theta}, \epsilon, v_i(t_l), \Delta_1(g)$

output: $I_i(t_l)$

Generate noise $\alpha_i \sim \mathsf{Laplace}\left(\frac{\Delta_1(g)}{\epsilon}\right)$;

for $j = 1, \ldots, \theta$ **do**

  Calculate $c_j(t_l) = \frac{\mathrm{l}_j(t_l)+\mathrm{r}_j(t_l)}{2}$;

  Generate a perturbed range $[\mathrm{l}_j(t_l) - \alpha_i, \mathrm{r}_j(t_l) + \alpha_i]$;

  if $v_i(t_l) < c_j(t_l)$ **then**

    $\mu_j(t_l) = \epsilon \cdot \frac{|c_j(t_l) - \mathrm{l}_j(t_l) + \alpha_i| - |c_j(t_l) - v_i(t_l)|}{2\Delta_1(g)}$;

  else

    $\mu_j(t_l) = \epsilon \cdot \frac{|r_j(t_l) - c_j(t_l) + \alpha_i| - |c_j(t_l) - v_i(t_l)|}{2\Delta_1(g)}$;

for $j = 1, \ldots, \theta$ **do**

  Calculate the probability that $v_i(t_l)$ locates in range j as $\hat{P}_j(t_l) = \frac{\exp(\mu_j(t_l))}{\sum_{j=1}^{\theta} \exp(\mu_j(t_l))}$;

Generate $u_i(t_l) \sim U[0, 1]$;

$\hat{P}_0 \leftarrow 0$;

for $j = 1, \ldots, \theta$ **do**

  if $\sum_{j'=0}^{j-1} \hat{P}_{j'}(t_l) \leq u_i(t_l) < \sum_{j'=0}^{j} \hat{P}_{j'}(t_l)$ **then**

    return $I_i(t_l) = j$;

Communication-Efficient Phase

In every interval $t_{l'}$ $(l' > l)$, each node n_i computes a perturbed range index $I_i(t'_l)$ according to Alg. 2 and reports $I_i(t'_l)$ to the service provider only if $I_i(t'_l) \neq I_i(t_{l'-1})$.

The detailed operations at each node n_i in communication-efficient phase are as follows.

1. Find the range $R_x(t_{l'})$ where $v_i(t_{l'})$ falls into.
2. Generate a perturbed range index $I_i(t_{l'})$ from the original range index x according to Alg. 2.
3. If $I_i(t_{l'}) \neq I_i(t_{l'-1})$, send $I_i(t_{l'})$ to the service provider and keep silent otherwise.

Upon receiving all the perturbed range indexes, the service provider processes them as follows.

1. For every node n_i that did not send $I_i(t_{l'})$, set $I_i(t_{l'}) = I_i(t_{l'-1})$.
2. Count the number of nodes with $I_i(t_{l'}) = j$ for every $j \in [1, \theta]$.
3. Calculate the percentile value $P_j(t_{l'})$ at the range $R_j(t_{l'})$'s left boundary $\mathrm{l}_j(t_{l'})$ as $\sum_{i=j}^{\theta} k_j(t_{l'})/k$ for every $j \in [1, \theta]$.
4. Determine the range $R_x(t_{l'})$ in which $\chi_r(t_{l'})$ falls into by finding x such that $P_x(t_{l'}) > r/100 > P_{x+1}(t_{l'})$.
5. Check whether $R_x(t_{l'})$ is a safe range. If not, take the predetermined action such as issuing a public safety alarm and keep silent otherwise.
6. Continue with the communication-efficient phase or start another initialization phase by broadcasting a new system-wide query in the next interval.

5.5.3 Performance Analysis

Correctness

The correctness of PriStream is affected by both the correctness of our proposed communication-efficient scheme and the accuracy guarantee after adopting mechanisms for differential privacy provision.

We first consider the correctness of our proposed scheme while ignoring the provision of differential privacy.

Theorem 5.5.1. *Let $k_j(t_l)$ be the number of nodes with statistic value in range $R_j(t_l)$ for all $j \in [1, \theta]$ and $\chi_r(t_l)$ be the rth percentile of a set of statistic values $\{v_i(t_l)\}_{i=1}^{k}$. The global condition $f(\chi_r(t_l)) > \tau$ holds at time interval t_l for some predefined threshold τ if there exists an unsafe range $R_j(t_l)$ such that $P_j(t_l) > r/100 > P_{j+1}(t_l)$, where $P_x(t_l) = \sum_{i=x}^{\theta} k_i(t_l)/k$ for all $x \in [1, \theta]$.*

Proof. Recall that $P_j(t_l) = \sum_{i=j}^{\theta} k_i(t_l)/k$ for all $j \in [1, \theta]$, where $k_j(t_l)$ is the number of nodes with statistic values in $R_j(t_l)$. Since $P_j(t_l) > r/100 > P_{j+1}(t_l)$, we have that $\sum_{i=j}^{\theta} k_i(t_l)/k > r/100 > \sum_{i=j+1}^{\theta} k_i(t_l)/k$. It follows that $\chi_r(t_l)$ is between the $\sum_{i=j+1}^{\theta} k_i(t_l)$th and the $\sum_{i=j}^{\theta} k_i(t_l)$th largest numbers among $\{v_i(t_l)\}_{i=1}^{k}$. We therefore have $\chi_r(t_l) \in R_j(t_l)$. Since $R_j(t_l)$ is an unsafe region, by definition, we have $f(\chi_r(t_l)) > \tau$. □

Next, we consider the accuracy of PriStream after each node perturbs its range index using Alg. 2. Specifically, the accuracy of PriStream depends on how accurate the service provider can learn the number of values in each range in each interval, which in turn depends on how accurate Alg. 2 perturbs a range index. The following theorem guarantees that the perturbed range index output by Alg. 2 would not be very different from the range index before perturbation.

Theorem 5.5.2. *If node n_i's statistic value $v_i(t_l)$ is outside of range $R_j(t_l)$ and the distance between $v_i(t_l)$ and the closer boundary of $R_j(t_l)$ is at least $\frac{2\Delta_1(g)}{\epsilon}\log\frac{1-\delta}{\delta^{1.5}(\theta-1)}$, then Alg. 2 will output a perturbed range index $I_i(t_l)=j$ with probability at most 2δ.*

Proof. In time interval t_l, the error introduced in either initialization or communication-efficient phase comes from the perturbation of the range with Laplace noise α_i and the exponential mechanism. For the range perturbation operation, since α_i is sampled from a Laplace distribution $\mathsf{Laplace}\left(\frac{\Delta_1(g)}{\epsilon}\right)$ with the cumulative distribution function

$$F(x)=\begin{cases}\dfrac{1}{2}\exp\left(\dfrac{x\epsilon}{\Delta_1(g)}\right), & \text{if } x<0,\\[2ex] 1-\dfrac{1}{2}\exp\left(\dfrac{-x\epsilon}{\Delta_1(g)}\right), & \text{if } x\geq 0.\end{cases} \tag{5.4}$$

Assume that $|\alpha_i|$ is at most ψ with probability $1-\delta$, where $\psi>0$. We have

$$1-2\cdot\frac{1}{2}\exp\left(-\frac{\psi\epsilon}{\Delta_1(g)}\right)=1-\delta.$$

Solving the above equation, we have $\psi=\frac{\Delta_1(g)}{\epsilon}\log\frac{1}{\delta}$. Therefore, we know that

$$\mathbf{Pr}\left[|\alpha_i|\leq\frac{\Delta_1(g)}{\epsilon}\log\frac{1}{\delta}\right]\geq 1-\delta. \tag{5.5}$$

In addition, assume that with probability $1-\delta$, the statistic value $v_i(t_l)$ at a node n_i exceeds a range boundary by φ. We further have

$$1-\frac{\exp(-\frac{\epsilon\varphi}{2\Delta_1(g)})}{(\theta-1)\exp(0)+\exp(-\frac{\epsilon\varphi}{2\Delta_1(g)})}=1-\delta.$$

Solving the above equation, we obtain

$$\varphi=\frac{2\Delta_1(g)}{\epsilon}\log\frac{1-\delta}{\delta(\theta-1)},$$

and

$$\mathbf{Pr}\left[\varphi\geq\frac{2\Delta_1(g)}{\epsilon}\log\frac{1-\delta}{\delta(\theta-1)}\right]\geq 1-\delta. \tag{5.6}$$

Considering the above two factors simultaneously, if the distance between $v_i(t_l)$ and $R_j(t_l)$ is larger than $\frac{2\Delta_1(g)}{\epsilon}\log\frac{1-\delta}{\delta^{1.5}(\theta-1)}$, Alg. 2 will output a perturbed range index $I_i(t_l)=j$ with probability at most 2δ, which can also be written as

$$\mathbf{Pr}\left[\varphi \leq \frac{2\Delta_1(g)}{\epsilon}\log\frac{1-\delta}{\delta^{1.5}(\theta-1)}\right] \leq 2\delta. \tag{5.7}$$

□

Theorem 5.5.3. *If node n_i's statistic value $v_i(t_l)$ is inside the range R_j and the distance between $v_i(t_l)$ and the closer boundary of R_j is more than $\frac{2\Delta_1(g)}{\epsilon}\log\frac{1-\delta}{\delta^{1.5}(\theta-1)}$, then Alg. 2 will output a perturbed range index $I_i(t_l)=j$ with probability at least $(1-2\delta)$.*

The proof of Theorem 5.5.3 is similar to that of Theorem 5.5.2.

Communication Overhead

The following theorem gives the communication overhead incurred by PriStream.

Theorem 5.5.4. *Given a PriStream execution process with a initialization and b communication-efficient phases, PriStream incurs the communication overhead of $a(|\mathcal{M}|+\varpi k)+\varpi\sum_{l=1}^{b}\lambda_l$, where $|\mathcal{M}|$ is the communication overhead incurred by broadcasting system information, $\varpi=\lfloor\log_2(\theta-1)\rfloor+1$ is the size of a range index, k is the number of nodes, λ_l is the number of nodes that submit range index to the service provider in lth communication-efficient phase, and $l\in[1,b]$.*

Proof. We analyze the communication overhead incurred in initialization and communication-efficient phases separately.

In the initialization phase, the service provider need send a message $\mathcal{M}$ containing the desired statistic metric generation function $g(\cdot)$, the precomputed range parameters $\{R_j(t_l)\}_{j=1}^{\theta}$, the differential-privacy parameter ϵ for differential-privacy

mechanism. We denote the communication overhead incurred by transmitting $\mathcal{M}$ by $|\mathcal{M}|$. In addition, each node sends a ϖ-bit range index to the service provider, totaling to ϖk bits.

In every communication-efficient phase, the node whose perturbed range index is different from that in last interval sends a ϖ-bit range index to the service provider, which incurs total communication overhead of $\lambda_l \varpi$ bits, where $\lambda_l \in [1, k]$ is the number of nodes that submit range index to the service provider in lth communication-efficient phase, where $l \in [1, b]$.

In summary, the overall communication overhead incurred by a PriStream execution process with a initialization and b communication-efficient phases is given by $a(|\mathcal{M}| + \varpi k) + \varpi \sum_{l=1}^{b} \lambda_l$. □

Privacy Analysis

The privacy of our proposed scheme is guaranteed by the following theorem.

Theorem 5.5.5. *PriStream consisting of a initialization and b communication-efficient phases maintains $2(a + b)\epsilon$-differential privacy.*

Proof. We follow the proof technique in [52] to prove that PriStream guarantees ϵ-differential privacy for each participating node in each phase. We consider all the noise components added in our scheme to obtain privacy guarantees for a multi-round process. For each node n_i, given two adjacent data streams $S_i(t_l)$ and $S_i'(t_l)$ that differ in only one element, we consider the scheme execution process as a sequential process consisting of a initialization and b communication-efficient phases.

In what follows, we analyze each phase of the PriStream execution in detail to show how different operations on $S_i(t_l)$ and $S_i'(t_l)$ will lead to the same output. For

convenience, we use E and E' to denote the PriStream execution process over two adjacent data streams $S_i(t_l)$ and $S'_i(t_l)$, respectively.

Initialization phase. Assume that each node n_i generates its statistic value as $v_i(t_l) = g(S_i(t_l))$ and $v'_i(t_l) = g(S'_i(t_l))$ in two executions E and E', respectively. For execution E, each node n_i generates Laplace noise α_i from a Laplace distribution. For execution E', the Laplace noise generated by node n_i is α_i. In the initialization phase, the execution E outputs range index j with probability $\hat{P}_j(t_l)$, which is given by

$$\hat{P}_j(t_l) = \frac{\exp(\mu_j(t_l))}{\sum_{j=1}^{\theta} \exp(\mu_j(t_l))}, \tag{5.8}$$

where

$$\mu_j(t_l) = \epsilon \cdot \frac{h}{2\Delta_1(g)}, \tag{5.9}$$

$$h = \begin{cases} |c_j(t_l) - \mathrm{l}_j(t_l) + \alpha_i| - |c_j(t_l) - v_i(t_l)|, \\ \qquad \forall\ v_i(t_l) < c_j(t_l), 1 \le j \le \theta, \\ |\mathrm{r}_j(t_l) - c_j(t_l) + \alpha_i| - |c_j(t_l) - v_i(t_l)|, \\ \qquad \forall\ v_i(t_l) \ge c_j(t_l), 1 \le j \le \theta, \end{cases} \tag{5.10}$$

$c_j(t_l) = \frac{\mathrm{l}_j(t_l)+\mathrm{r}_j(t_l)}{2}$, $\mathrm{l}_j(t_l)$ and $\mathrm{r}_j(t_l)$ are the left and right boundaries of the jth range in which $v_i(t_l)$ actually locates in time interval t_l, respectively. The ℓ_1-sensitivity of utility function h is $\Delta_1(g)$.

Assume $v_i(t_l) \in R_j(t_l)$ in time interval t_l, we have

$$\begin{aligned}
\frac{\Pr[\mathsf{M}(S_i(t_l), h) = j]}{\Pr[\mathsf{M}(S'_i(t_l), h) = j]} =& \frac{\left(\frac{\exp(\mu_j(t_l))}{\sum_{x=1}^{\theta}\exp(\mu_x(t_l))}\right)}{\left(\frac{\exp(\mu'_j(t_l))}{\sum_{x=1}^{\theta}\exp(\mu'_x(t_l))}\right)} \\
=& \left(\frac{\exp(\mu_j(t_l))}{\exp(\mu'_j(t_l))}\right) \cdot \left(\frac{\sum_{x=1}^{\theta}\exp(\mu'_x(t_l))}{\sum_{x=1}^{\theta}\exp(\mu_x(t_l))}\right) \\
=& \exp\left(\mu_j(t_l) - \mu'_j(t_l)\right) \cdot \left(\frac{\sum_{x=1}^{\theta}\exp(\mu'_x(t_l))}{\sum_{x=1}^{\theta}\exp(\mu_x(t_l))}\right)
\end{aligned} \tag{5.11}$$

For $v_i(t_l) < c_j(t_l)$, we have

$$\begin{aligned}
\mu_j(t_l) - \mu'_j(t_l) =& \epsilon \cdot \frac{|c_j(t_l) - \mathrm{l}_j(t_l) + \alpha_i| - |c_j(t_l) - v_i(t_l)|}{2\Delta_1(g)} \\
& - \epsilon \cdot \frac{|c_j(t_l) - \mathrm{l}_j(t_l) + \alpha'_i| - |c_j(t_l) - v'_i(t_l)|}{2\Delta_1(g)} \\
\leq& \frac{\epsilon}{2} \cdot \frac{2\Delta_1(g)}{\Delta_1(g)} \\
=& \epsilon,
\end{aligned} \tag{5.12}$$

$$\begin{aligned}
\mu'_j(t_l) &= \epsilon \cdot \frac{|c_j(t_l) - \mathrm{l}_j(t_l) + \alpha'_i| - |c_j(t_l) - v'_i(t_l)|}{2\Delta_1(g)} \\
&\leq \epsilon \cdot \frac{|c_j(t_l) - \mathrm{l}_j(t_l) + \alpha_i| + \Delta_1(g)}{2\Delta_1(g)} \\
&\quad - \epsilon \cdot \frac{|c_j(t_l) - v_i(t_l)| - \Delta_1(g)}{2\Delta_1(g)} \\
&= \epsilon \cdot \frac{|c_j(t_l) - \mathrm{l}_j(t_l) + \alpha_i| - |c_j(t_l) - v_i(t_l)| + 2\Delta_1(g)}{2\Delta_1(g)} \\
&= \mu_j(t_l) + \epsilon.
\end{aligned} \tag{5.13}$$

For $v_i(t_l) \geq c_j(t_l)$, we have

$$\begin{aligned}
\mu_j(t_l) - \mu'_j(t_l) =& \epsilon \cdot \frac{|\mathrm{r}_j(t_l) - c_j(t_l) + \alpha_i| - |c_j(t_l) - v_i(t_l)|}{2\Delta_1(g)} \\
& - \epsilon \cdot \frac{|\mathrm{r}_j(t_l) - c_j(t_l) + \alpha'_i| - |c_j(t_l) - v'_i(t_l)|}{2\Delta_1(g)} \\
\leq& \epsilon \cdot \frac{2\Delta_1(g)}{2\Delta_1(g)} \\
=& \epsilon,
\end{aligned} \tag{5.14}$$

$$
\begin{aligned}
\mu'_j(t_l) &= \epsilon \cdot \frac{|\mathrm{r}_j(t_l) - c_j(t_l) + \alpha'_i| - |c_j(t_l) - v'_i(t_l)|}{2\Delta_1(g)} \\
&\leq \epsilon \Bigg(\cdot \frac{|\mathrm{r}_j(t_l) - c_j(t_l) + \alpha_i| + \Delta_1(g)}{2\Delta_1(g)} \\
&\quad + \cdot \frac{-|c_j(t_l) - v_i(t_l)| + \Delta_1(g)}{2\Delta_1(g)} \Bigg) \\
&= \epsilon \cdot \frac{|\mathrm{r}_j(t_l) - c_j(t_l) + \alpha_i| - |c_j(t_l) - v_i(t_l)| + 2\Delta_1(g)}{2\Delta_1(g)} \\
&= \mu_j(t_l) + \epsilon.
\end{aligned} \tag{5.15}
$$

Therefore, we have

$$
\begin{aligned}
\frac{\Pr[\mathsf{M}(S_i(t_l), h) = j]}{\Pr[\mathsf{M}(S'_i(t_l), h) = j]} &\leq \exp(\epsilon) \Bigg(\frac{\sum_{x=1}^{\theta} \exp\left(\mu_x(t_l) + \epsilon\right)}{\sum_{x=1}^{\theta} \exp(\mu_x(t_l))} \Bigg) \\
&= \exp(\epsilon) \Bigg(\frac{\sum_{x=1}^{\theta} \exp(\mu_x(t_l)) \cdot \exp(\epsilon)}{\sum_{x=1}^{\theta} \exp(\mu_x(t_l))} \Bigg) \\
&= \exp(2\epsilon),
\end{aligned} \tag{5.16}
$$

which indicates that the initialization phase guarantees 2ϵ-differential privacy.

Communication-efficient phase. The operations of each communication-efficient phase is similar to those of the initialization phase except for the case in which some nodes do not need to send the index of the range in which it resides to the service provider if the statistic value remains in the same range as that in the previous phase. Therefore, we can similarly obtain the same result that each communication-efficient phase guarantees 2ϵ-differential privacy.

The whole PriStream execution process. As mentioned, the whole execution process is a sequential process consisting of a initialization and b communication-efficient phases. Since the noise added in each phase is drawn independently, the probability difference of obtaining the same output for the whole execution process can be considered the multiplication of the probability difference in each phase. Therefore, for the whole execution process, the probability of obtaining the output based

on execution E' differs from that of execution E by a factor of at most $\exp(2(a+b)\epsilon)$, which guarantees $2(a+b)\epsilon$-differential privacy. □

5.6 Performance Evaluation

In this section, we evaluate the performance of PriStream via MATLAB simulations based on both real-world and synthetic datasets.

5.6.1 Simulation Setup

We adopt the following metrics to evaluate the performance of PriStream.

- *Communication overhead*: We quantify the communication overhead by the number of bits transmitted between the service provider and nodes during stream monitoring.

- *Accuracy*: The accuracy is used to evaluate the utility after introducing Alg. 2. We treat the range index of each round in communication-efficient scheme as the ground truth and compare it with the range index of the corresponding round in PriStream. The accuracy is defined as the ratio of the number of the same indexes over the total rounds.

- *Privacy loss*: The privacy loss is defined as

$$\hat{\epsilon} = \max \ln \frac{\mathbf{Pr}[\mathsf{M}(S_i(t_l), h) = j]}{\mathbf{Pr}[\mathsf{M}(S_i'(t_l), h) = j]}, \tag{5.17}$$

  where S_i and S_i' are two adjacent streams which differ in only one element. Obviously, the smaller $\hat{\epsilon}$, the less impact of the change of one element on the range index generation algorithm, the higher level of differential privacy is offered, and vice versa.

We use two datasets to evaluate the performance of PriStream. The first dataset is MHEALTH [27], a mobile health dataset that comprises body motion and vital sign measures for several volunteers of diverse profiles while performing 12 physical activities such as walking, running and climbing stairs. The dataset contains totally 1,215,745 recordings, each of which is composed of 24 types of signals from the sensors such as accelerometer, gyroscope, and magnetometer. In this chapter, we used all the 1,215,745 recordings for one type of signal because they are at the same scale and are the focus of this chapter; we leave the monitoring and evaluation of multi-dimension streams with different scales as future work. We then randomly partition them into 1000 subsets, representing 1000 distributed nodes, each of which has about 1216 data items, corresponding to 1216 intervals. The service provider starts the initialization phase at interval 608 and then conducts subsequent 608 rounds of queries, and each node will generate $v_i(t_l)$ based on its previous 608 data items. The second dataset is a synthetic dataset generated by MATLAB used to simulate the case with different data distribution. In particular, the data in MHEALTH dataset follow Gaussian distribution. We extract the data range from MHEALTH dataset and then generate a synthetic dataset which is uniformly distributed in the same data range from. All other metrics such as the number of nodes, the number of intervals, and the number of query rounds in synthetic dataset are the same as that in MHEALTH dataset.

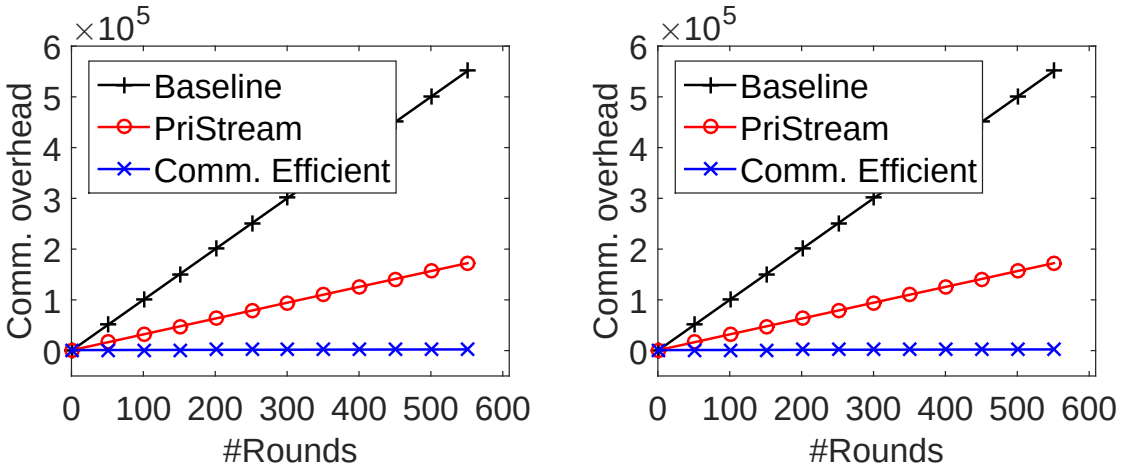
The default simulation settings are summarized in Tab. 5.1.

5.6.2 Simulation Results

We report the simulation results of a communication-efficient scheme (e.g., PriStream without Alg. 2), PriStream and a baseline scheme that lets each node directly submit its statistic value to the service provider.

Table 5.1: Default Simulation Settings.

Para.	Value	Meaning
k	1000	The number of nodes
ϵ	0.15	The differential privacy parameter
θ	100	The number of ranges
r	80	The percentile value



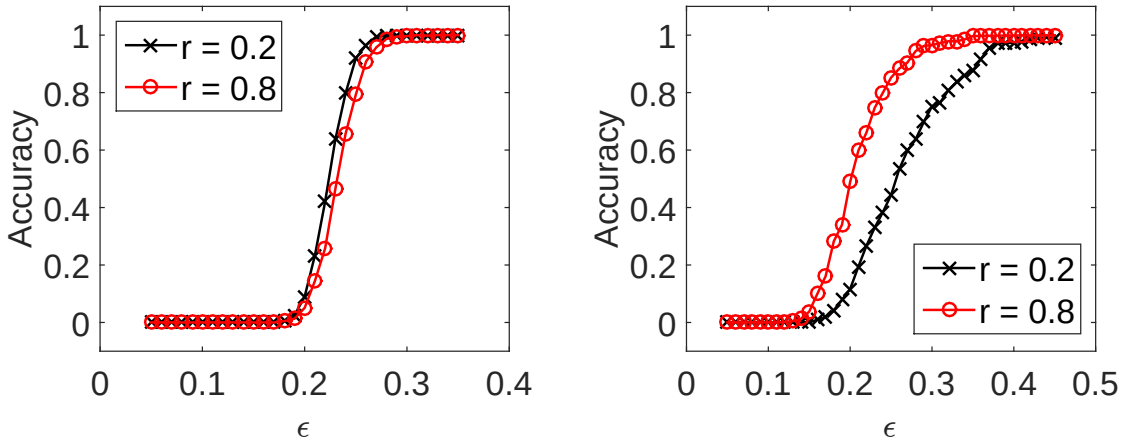
(a) MHEALTH Dataset

(b) Synthetic Dataset

Figure 5.1: Impact of the Number of Rounds on Communication Overhead.

Fig. 5.1 compares the communication overhead of the baseline, the communication-efficient, and PriStream schemes with b (i.e., the number of rounds) varying from 1 to 600. We can see that the communication-efficient and PriStream schemes incur much lower communication overhead than the baseline scheme does. The reason is than the number of nodes that submit data to the service provider in communication-efficient and PriStream schemes is much smaller than in the baseline scheme. Besides, by reporting range index instead of statistic value, communication overhead is further

reduced. In addition, we can see that PriStream scheme incurs higher communication overhead than that of the communication-efficient scheme. The reason is that the range index of each node's statistic value is perturbed to other range indexes for the protection of data privacy, resulting in more range index being submitted to the service provider.



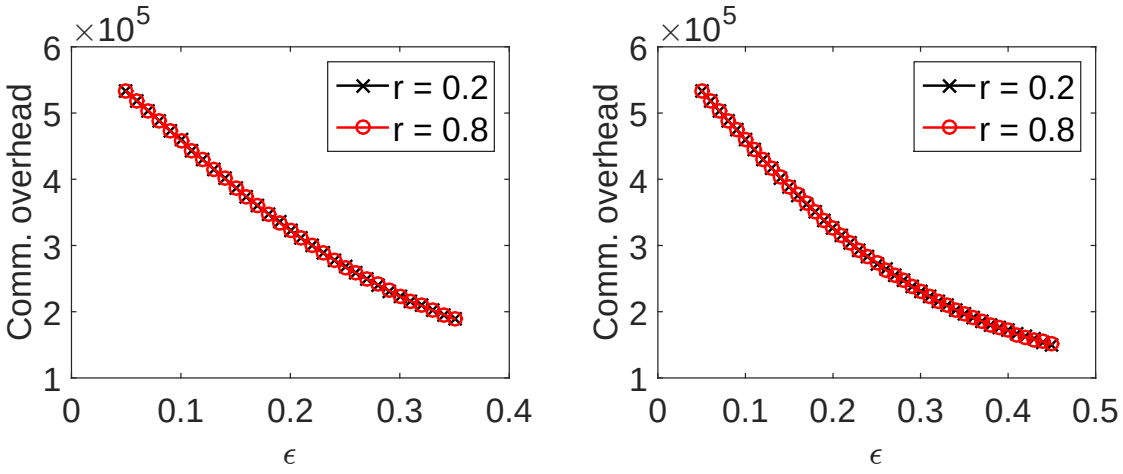
(a) MHEALTH Dataset

(b) Synthetic Dataset

Figure 5.2: Impact of ϵ on Accuracy.

Fig. 5.2 shows the relationship between the accuracy and the differential privacy parameter ϵ. Obviously, the baseline scheme, which does not consider the data privacy, achieves 100% accuracy without being affected by the change of ϵ. However, the accuracy of PriStream increases as the differential privacy parameter ϵ increases. The reason is that as ϵ increases, the perturbed range index generated by Alg. 2 is more likely to be the same as its perturbed range index in previous interval, leading to less range index updates.

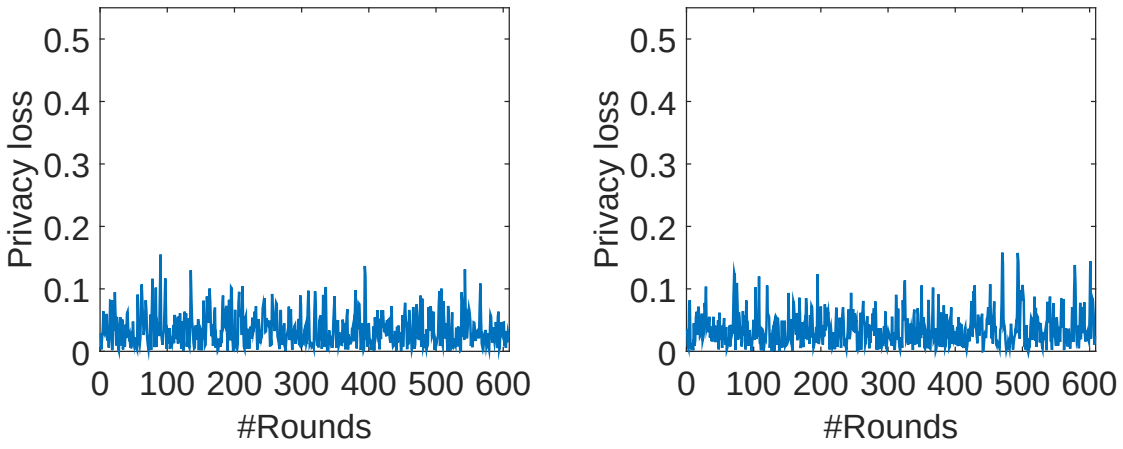
Fig. 5.3 shows the impact of differential privacy parameter ϵ on the communication overhead of PriStream. We can see that the communication overhead decreases as ϵ increases, demonstrating a trade-off between ϵ and communication overhead. The



(a) MHEALTH Dataset (b) Synthetic Dataset

Figure 5.3: Impact of ϵ on Communication Overhead.

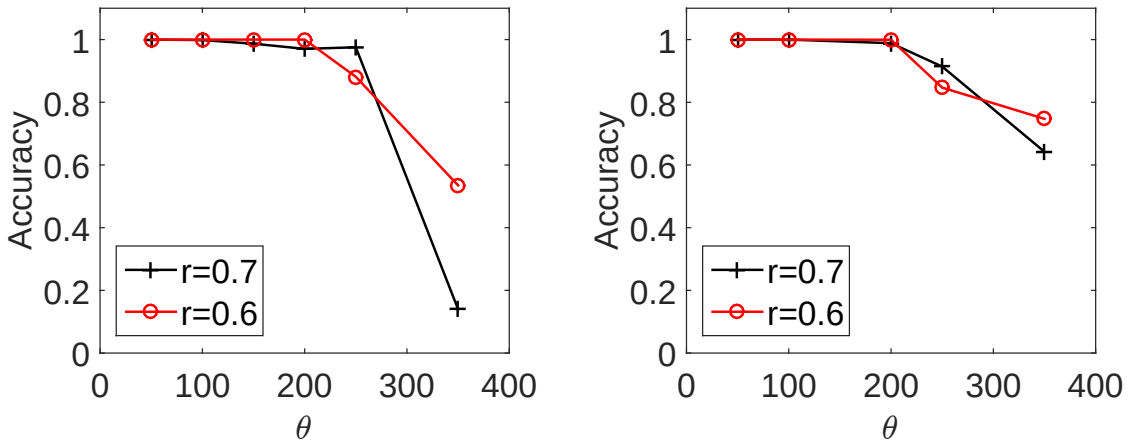
reason is that the higher ϵ, the higher the probability that a node's statistic value remains in the same range after perturbation, and the fewer nodes that need to report range index updates.



(a) MHEALTH Dataset (b) Synthetic Dataset

Figure 5.4: Privacy Loss of PriStream.

Fig. 5.4 illustrates the privacy loss after using PriStream. We can see that the privacy loss is always below 0.15. According to Tab. 5.1, the simulation setting of differential privacy parameter is $\epsilon = 0.15$, which indicates that our designed scheme can always guarantee the desired 2ϵ-differential privacy.



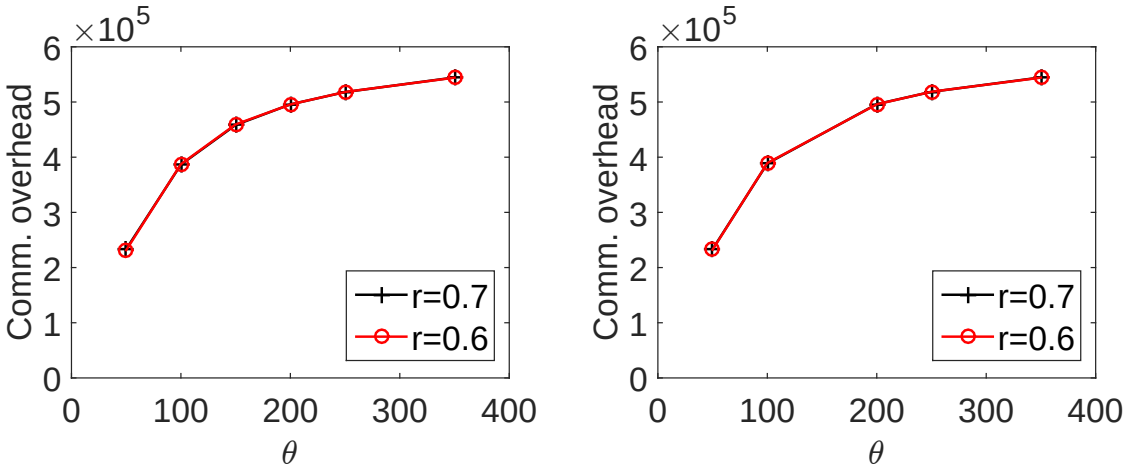
(a) MHEALTH Dataset

(b) Synthetic Dataset

Figure 5.5: Impact of θ on Accuracy.

Fig. 5.5 shows the impact of the number of ranges on accuracy. We can easily find that the accuracy decreases as the number of ranges increases for both datasets. The reason is that the larger the number of ranges, the smaller the range size, the higher the probability that the statistic value is perturbed to other ranges.

Fig. 5.6 shows the impact of θ on communication overhead. We can see that the communication overhead increases as θ increases. The reason is that the larger the θ is, the smaller the range size, the higher the probability that the statistic value is perturbed to a different range, and the higher communication overhead.



(a) MHEALTH Dataset

(b) Synthetic Dataset

Figure 5.6: Impact of θ on Communication Overhead.

5.7 Summary

This chapter proposes PriStream [141], a novel privacy-preserving and communication-efficient distributed stream monitoring system. Different from previous work on monitoring the function of mean statistic value, our proposed scheme monitors the statistic value at the given percentile rank in a privacy-preserving and communication-efficient fashion. The efficacy and efficiency of our PriStream are confirmed by detailed MATLAB simulations.

Chapter 6

VIDEO-ASSISTED KEYSTROKE INFERENCE FROM TABLET BACKSIDE MOTION

6.1 Introduction

The past few years have witnessed the proliferation of tablets in everyday life. According to a Gartner report [13], global tablet shipments will reach 321 million and surpass PC shipments in 2015. Being lighter than laptops and having larger touchscreens than smartphones, tablets perfectly fill the gap between laptops and smartphones and have become an indispensable category of mobile computing devices. People are increasingly using tablets in every aspect of life, including voice/video communications, Internet browsing, web transactions, online banking, reading, multimedia playing, etc.

The deep penetration of tablets in people's daily life has made them attractive targets for various keystroke inference attacks that aim to infer a user's typed inputs (such as usernames, passwords, SSNs, and emails) on the tablet touchscreen. Although existing authentication schemes [40, 41, 83, 90, 127, 138] can prevent unauthorized access to mobile devices, prior work has shown that an attacker can successfully infer the PIN or even the words entered on the soft (tablet) keyboard by surreptitiously video-recording a target user's input process and then analyzing the reflection of the touchscreen, spatial hand dynamics, or the relative finger positions on the touchscreen [24, 25, 26, 103, 123, 132, 152, 159]. These studies commonly assume that the attacker can capture a user's interaction with the touchscreen with little or no visual obstruction, which greatly limits the applicability of these attacks.

In this chapter, we propose VISIBLE [140], a novel video-assisted keystroke inference framework that allows an attacker to infer a tablet user's typed inputs on the touchscreen by recording and analyzing the video of the tablet backside during the user's input process. VISIBLE is motivated by our observation that the keystrokes on different positions of the tablet's soft keyboard cause its backside to exhibit different motion patterns. In contrast to previous keystroke inference techniques [24, 25, 26, 103, 123, 132, 152, 159], VISIBLE does not require the attacker to visually see the victim's input process and thus enables much more surreptitious keystroke inference from a distance.

The design of VISIBLE faces two major challenges. First, the backside motion caused by user keystrokes is very subtle and requires effective methods to detect and quantify. Second, since the soft keyboard on the tablet is much smaller than a normal keyboard, the motion patterns caused by tapping adjacent keys are close, making accurate differentiation particularly challenging. To tackle these two challenges, VISIBLE uses complex steerable pyramid decomposition to detect and quantify the subtle keystroke-induced motion patterns of the tablet backside, differentiates different motion patterns using a multi-class Support Vector Machine, and refines the inference results using a dictionary and linguistic models.

We thoroughly evaluate the performance of VISIBLE via comprehensive experiments on an Apple iPad 2 tablet and a Google Nexus 7 tablet. Our experiment results show that VISIBLE can infer a single key entered on the alphabetical soft keyboard with an average accuracy of 36.2% and that the correct key is within the inferred key's one-hop and two-hop neighbors with probabilities 83.6% and 97.9%, respectively. Similarly, VISIBLE achieves an accuracy of 38% for single-key inference on the PIN soft keyboard, and the correct key is within the inferred key's one-hop neighbors with probability 68%. For word inference, VISIBLE can produce a list of

candidate words, and the correct word is in the top-5, top-10, top-25, and top-50 candidate words with probabilities 48.0%, 63.0%, 77.8%, and 92.6%, respectively. We also show that the attacker can successfully infer typed sentences based on the linguistic relationship between adjacent words. These experiment results confirm the high efficacy of VISIBLE.

The rest of this chapter is organized as follows. Section 6.2 presents the related work. Section 6.3 introduces some background knowledge for video processing. Section 6.4 describes the adversary model. Section 6.5 details the design of VISIBLE. Section 6.6 evaluates VISIBLE through extensive experiments. Section 6.7 summarizes this chapter and discusses possible countermeasures and future work.

6.2 Related Work

In this section, we briefly introduce the prior work most related to VISIBLE. Prior keystroke inference attacks can be broadly classified into two categories: video-based attacks and sensor-based attacks.

Video-based Attacks In this category, the adversary uses video-based side channels in combination with computer vision techniques to infer a user's typed inputs. Early work along this line focuses on physical keyboards. Backes *et al.* [24, 25] exploited the reflections of screens on glasses, tea pots, spoons, plastic bottles, eyes of the user, walls, and even the user's clothes to recover the content displayed on the computer monitor. Balzarotti *et al.* [26] introduced an attack that automatically recovers the typed text solely from a video of the user typings by analyzing the light diffusion surrounding the key change. This attack requires a camera to directly record the finger typings on the physical keyboard.

There have also been some video-based attacks on the soft keyboards of touchscreen mobile devices. In [103], Maggi *et al.* presented an attack that automatically recognizes typed inputs from the key magnifications of touchscreen mobile devices. Raguram *et al.* [123] showed how to automatically reconstruct the text input on a soft keyboard from the reflection of the device's screen on the victim's sunglasses. Xu *et al.* [152] introduced an attack to accurately reconstruct the text input on a mobile device by tracking the positions of the victim's fingers as they move across the soft keyboard. In [159], Yue *et al.* showed how to infer the user input even if neither text nor popup can be seen from the video of user typings. This attack exploits the homographic relationship between touching images and a reference image showing a soft keyboard. All these attacks require the attacker to acquire a video capturing the victim's typings on the touchscreen or the touchscreen reflection.

Our work is most related to [132], in which Shukla *et al.* introduced a video-based attack on the PIN-entry process of a smartphone that decodes the typed PIN by exploiting the spatiotemporal dynamics of the hands during typing. Both VISIBLE and the attack proposed in[132] only require the attacker to video-record the backside of a smartphone, which was considered safe previously. VISIBLE, however, has much wider applicability than [132]. In particular, the attack introduced in [132] requires the attacker to record the victim's hand movements during the PIN-entry process, which is not always possible. For example, the victim's hand movements are very likely to be obscured by the tablet itself. In contrast, VISIBLE works even if the victim's hand movements are not visible from the video of the device backside.

Sensor-based Attacks Tremendous efforts have been made on inferring user inputs on mobile devices from the data generated by various on-board sensors. It has been shown in [35] and [119] that the user's password can be inferred from the

smartphone's accelerometer data. Moreover, some recent work [109, 153] demonstrated a similar attack that exploits the data from both accelerometer and gyroscope. Other on-board sensors that have been exploited include microphones and front cameras [113, 133]. All these work require the attacker to obtain sensor data via either malicious applications (e.g., malicious Apps or web scripts) or unprotected network transmissions, which limit their applicability. In contrast, VISIBLE only requires the attacker to record the video of the tablet backside during the victim's typing process, which is both easier to launch and more difficult to detect.

Also related is the work on using on-board sensors to infer the keystrokes of nearby physical keyboards. In [175], Zhuang *et al.* showed how to recover typed keys from sound recordings of a user's typings on a physical keyboard. Berger *et al.* [29] presented another attack that infers the user input from the acoustic emanations of the physical keyboard with the assistance of a dictionary. A similar attack was presented in [174], which also uses acoustic emanations of the physical keyboard but does not need a language model or dictionary. In [106], the authors demonstrated an attack that infers the typed keys of a physical keyboard from the vibration caused by each keystroke detected by a nearby smartphone's accelerometer. Such attacks, although effective, can only be used when the attacker is near the victim due to the short transmission range of acoustic and vibration signals. In contrast, VISIBLE can be launched from a much larger distance.

6.3 Video Processing Basics

In this section, we introduce two computer vision techniques, phase-based optical flow estimation and complex steerable pyramid decomposition, underlying VISIBLE.

6.3.1 Phase-based Optical Flow Estimation

An optical flow refers to apparent motion patterns of image objects between two consecutive frames caused by the object or camera movement. Optical flow estimation is the process of characterizing and quantifying the object motions in a video stream, often for motion-based object detection and tracking systems. Phase-based optical flow is a popular optical flow estimation technique which estimates the motion field using phase information. For example, constant phase contours are tracked by computing the phase gradient of spatiotemporally bandpassed images, which provides a good approximation to the motion in [55]. As another example, Gautama *et al.* [59] proposed to estimate motion by computing the temporal gradient of the phases of a partially bandpassed video. In comparison with other flow estimation techniques, phased-based estimation methods are more robust to smooth shading, lighting variations, and small deviations from image translations.

6.3.2 Complex Steerable Pyramid Decomposition

Steerable pyramid decomposition [134] is a standard technique that decomposes an image according to spatial scale, orientation, and position to capture the variance of a texture in both intensity and orientation, which has been widely used in image processing and motion detection. Since an image may contain multiple objects of different sizes, and these objects may contain features of different sizes and be at different distances from the viewer, any analysis procedure that is only applied at a single scale may lose information at other scales. To simultaneously detect multiple objects' motion patterns, analysis need be carried out at different scales simultaneously [16]. In addition, the same object may also exhibit totally different features

in different orientations. To comprehensively analyze the features of an object and detect its motion pattern, it is necessary to decompose it in different orientations.

Complex steerable pyramid decomposition [121] extends the original steerable pyramid decomposition by representing an image in a complex form comprising real and imaginary parts. In comparison with steerable pyramid decomposition, complex steerable pyramid decomposition additionally measures local phase and energy in some texture descriptors. Such measures have proved important throughout computer vision. Using complex steerable pyramid decomposition, we can obtain the phase and amplitude of each pixel of an image at each spatial scale and orientation over time.

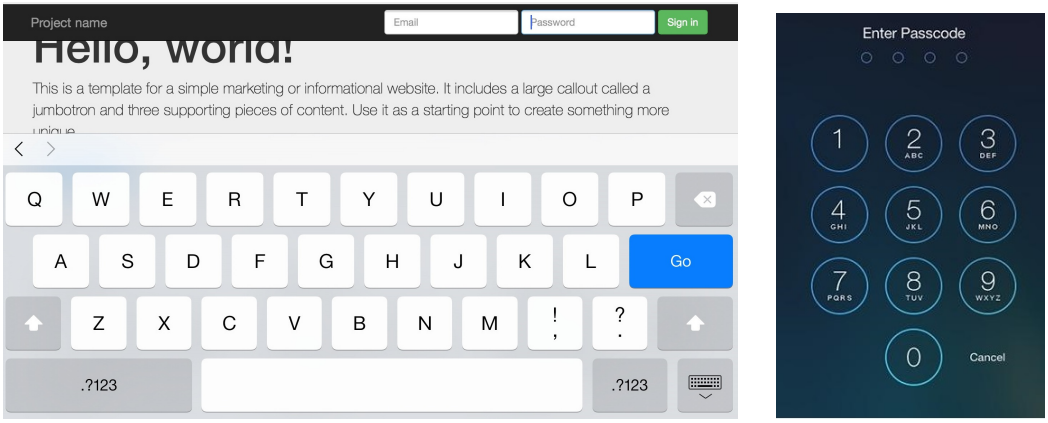
6.4 Adversary Model

(a) An iPad and a Holder. (b) Attack Scenario. (c) The Same Attack Scenario From a Different Angle.

Figure 6.1: Examples of a Tablet Holder and an Attack Scenario.

We consider a victim user with a tablet such as iPad 2 or Nexus 7. We assume that the victim places the tablet on a tablet holder (e.g., the one shown in Fig. 6.1a) on a desk and types on a soft keyboard. Such scenarios are very common in daily life, e.g., in conferences or seminars where researchers take notes or write emails. We focus on two types of soft keyboards in this chapter, the alphabetical and PIN keyboards,

as shown in Fig. 6.2. The extension of VISIBLE to the alphanumeric soft keyboard is left as future work.



(a) Alphabetical Keyboard. (b) PIN Keyboard.

Figure 6.2: Alphabetical and PIN Soft Keyboard Illustrations.

We assume that an attacker intends to infer any typed input on the victim's tablet, which can be single keys, words, or sentences. This enables VISIBLE to infer any sensitive typed input on the tablet, such as usernames, passwords, and emails. The victim is alert to shoulder-surfing attacks in the sense that the attacker cannot get too close to the victim during his typing process. In addition, the attacker is unable to see the tablet touchscreen or the victim's hand movement during his typing process from any direction. Moreover, we assume that the attacker cannot obtain the sensor data by running malware such as Trojans or malicious web scripts on the victim's tablet. These assumptions make previous video-based attacks [24, 25, 26, 103, 123, 132, 152, 159] and sensor-based attacks [29, 35, 106, 109, 113, 119, 133, 153, 174, 175] inapplicable.

We assume that the attacker has the following capabilities. First, the attacker can use camcorders with advanced lens to record the backside of the victim's tablet during his input process, possibly from a long distance. Second, the attacker can

record the attack scenario and reconstruct it afterwards. Specifically, the attacker can measure the angle between the victim's tablet and the desk, the angle between the tablet and the camcorder, and the distance between the tablet and the camcorder by analyzing multiple images taken from different angles and positions using distance and angle estimation algorithms [148]. Finally, the attacker has the same holder and the tablet with the same soft keyboard layouts as the victim's.

6.5 VISIBLE Framework

In this section, we give an overview of VISIBLE and then detail each step of the attack.

6.5.1 VISIBLE Overview

VISIBLE infers the victim's typed inputs from the video of tablet backside motion. The high-level design of VISIBLE is shown in Fig. 6.3, which consists of eight steps as follows.

1. *Video Recording and Preprocessing*: In this step, we record a video capturing the motion of the tablet backside during the victim's typing process. We assume neither the touchscreen nor the victim's hand movement can be seen from the video. We crop the video clip to keep the text-input part only.

2. *Areas of Interests (AOIs) Detection and Selection*: In this step, we detect all the areas with texture information on the tablet backside and select a few areas as AOIs for further processing. Exemplary AOIs are the buttons, camera, loudspeaker, logo, and texts on the tablet backside.

3. *AOI Decomposition*: In this step, we decompose each selected AOI in each frame using complex steerable pyramid decomposition.

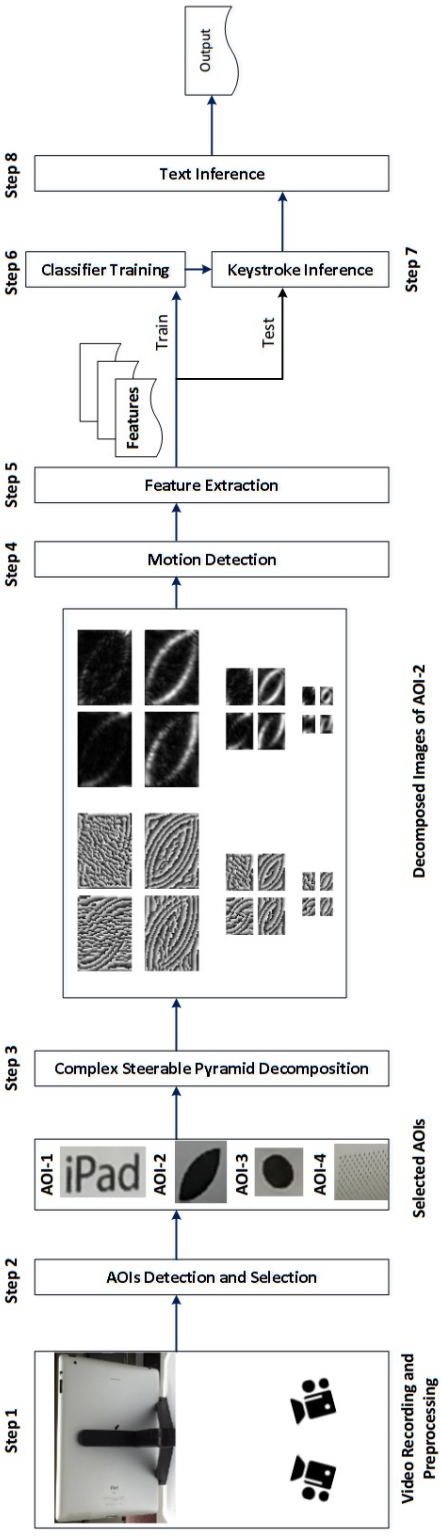


Figure 6.3: The VISIBLE Framework.

4. *Motion Detection via Phase Variances*: In this step, we analyze the phase variances of each AOI over time and quantify the corresponding motion amplitude.

5. *Feature Extraction*. In this step, we extract features in both temporal and spatial domains to represent the motion patterns of each AOI.

6. *Classifier Training*. In this step, we let multiple attackers mimic the victim's typing process, record videos of their tablet backsides, and process their videos to train a classifier.

7. *Keystroke Inference*. In this step, we use a classifier trained in Step 6) to test the collected data from the victim's typing process.

8. *Text Inference*. In this step, we infer the possible words by considering meaningful alphabetical combinations using a dictionary and exploit the relationship between adjacent words to further infer sentences.

In what follows, we present each step in detail.

6.5.2 Video Recording and Preprocessing

One or more camcorders can be used to video-record the tablet backside during the victim's typing process. For more accurate inference results, the video should capture the entire backside of the tablet, and the image resolution should be sufficiently high. In our experiments, we use two camcorders that focus on the left-half and right-half of the tablet backside, respectively. By doing so, each camcorder only needs to focus on a relatively small area with sufficiently high resolution to capture the detailed texture information on the tablet backside.

In our study, we find that the following four factors affect subsequent motion detection.

- *Image resolution and frame rate.* The image resolution determines the amount of detailed textural information that can be obtained from the image and should be as high as possible. The frame rate is the number of frames taken within one second, and a higher frame rate could help capture more detailed motion information. We recommend 1080p60 [1] HD or higher resolutions and frame rates.

- *Zoom setting.* The zoom setting of the camcorder is jointly determined by the distance between the camcorder and the tablet and the lens properties. We recommend zooming in as much as possible while capturing the entire tablet backside.

- *Light condition.* The bright light condition can result in better motion detection, as the imaging component of camcorders generates larger random noise in low-light condition that pollutes the motion signal.

- *Recording angle.* The angle between the camcorder and tablet need be adjusted to capture the entire tablet backside, which can be easily satisfied in practice.

We also video-record the attack scenario layout to measure the distances and angles between the camcorders and the target tablet as well as the angle between the tablet and the desk. This is important for reconstructing the attack scenario later. In practice, the attacker can take multiple images of the attack scenario from different angles and estimate the distances and angles of interest using standard distance and angle estimation algorithms such as [148].

After obtaining the video of the tablet backside, we manually crop the unwanted part such that the remaining video contains only the typing process of interest that

[1] 1080p60 denotes that the camcorder can take 1920x1080 videos at a rate of 60 frames per second.

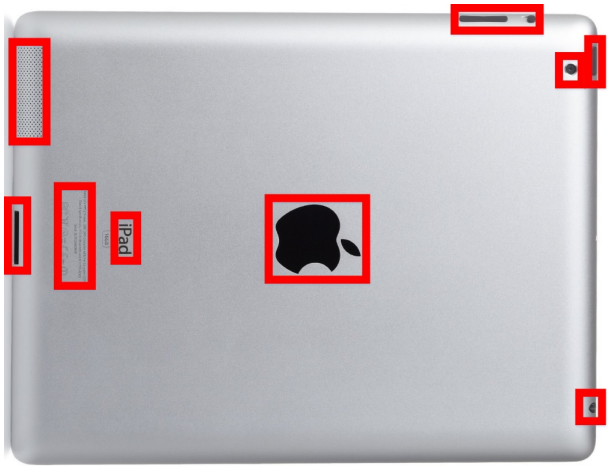
is easily identifiable in practice. In practice, for the PIN keyboard, since the user usually needs to press power or home button first, the keys entered subsequently are very likely to be PINs. Similarly, for the alphabetical keyboard, the typing process can be identified by continuous typings (e.g., small arm movements). Finally, we decompose the cropped video into a series of frames. Note that video croppings can be automated after more sophisticated and extensive programming effort.

6.5.3 AOIs Detection and Selection

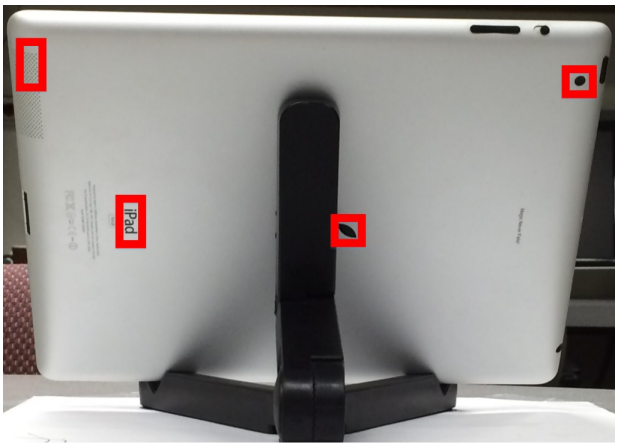
After obtaining a series of frames, we proceed to identify all the Areas of Interests (AOIs) on the tablet backside, where each AOI refers to an area containing unique texture information. In computer vision, the texture information of an image refers to the relationship of nearby pixels. Given a frame, we use the textured area detection algorithm [30] to identify the areas with texture information and select them as AOIs for subsequent motion detection.

Fig. 6.4a shows an image of the iPad 2 backside, where the areas enclosed by rectangles are identified AOIs, which include the power button, the voice up and down buttons, the silence button, the backside camera, the ear plug, the loudspeaker, the logo, the manufacture information, and other decorative patterns. These AOIs can be used to detect the tablet backside motion. In practice, almost every tablet's backside contains adequate areas with rich texture information. Even if the tablet is protected by a backside cover like iPad Smart Case, the backside of the cover itself still contains areas with rich texture information making it easy to find enough AOIs for subsequent motion detection.

We then select a subset of the AOIs that are near the edges of the tablet backside and separated from each other, as these AOIs tend to have larger and more distinctive motions than others, making them more capable of differentiating the motion patterns



(a) Possible AOIs.



(b) Selected AOIs.

Figure 6.4: Possible and Selected AOIs on an iPad 2's Backside.

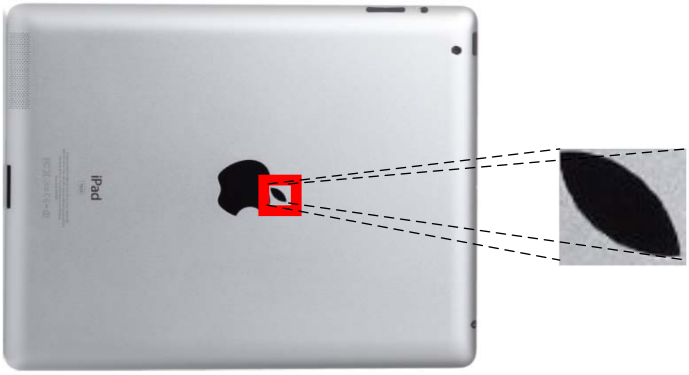
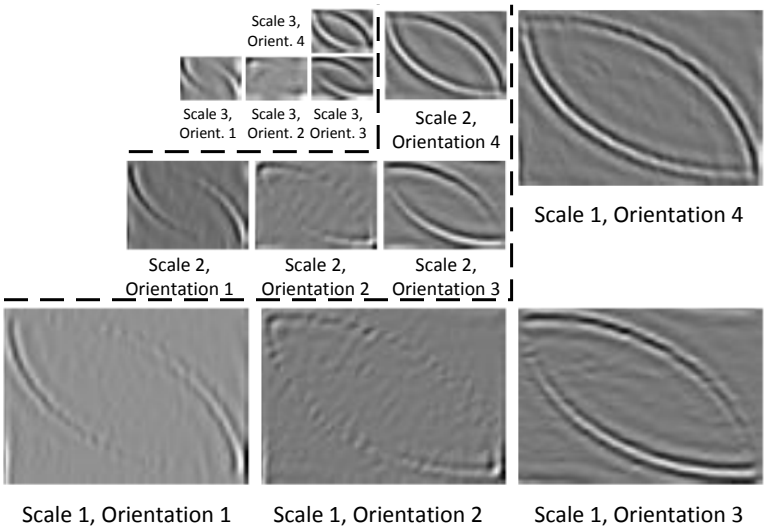
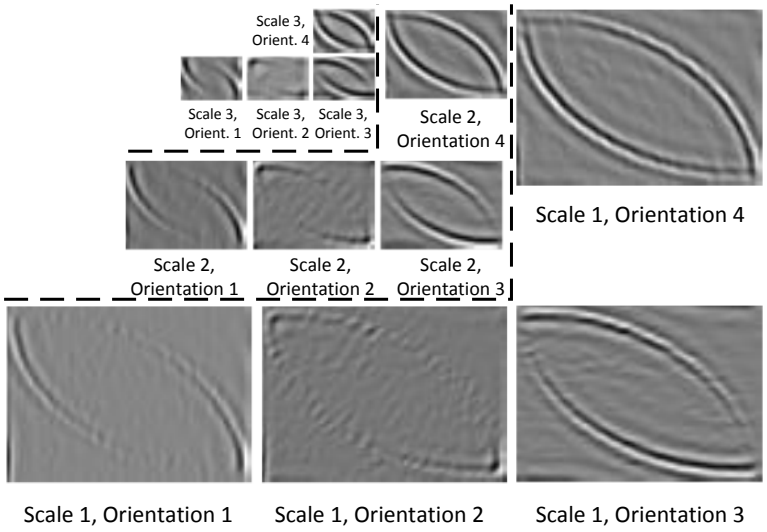


Figure 6.5: An Example of a Selected AOI, AOI-2.

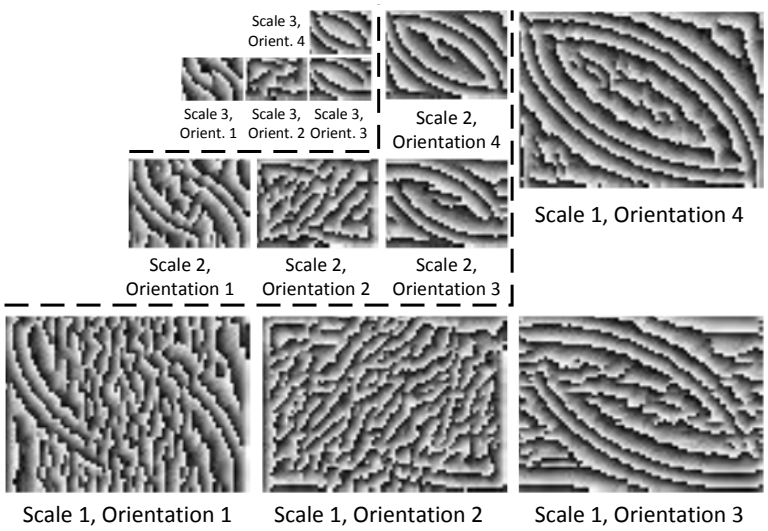


(a) Real Parts of the Decomposed Images at Each Scale and Orientation.

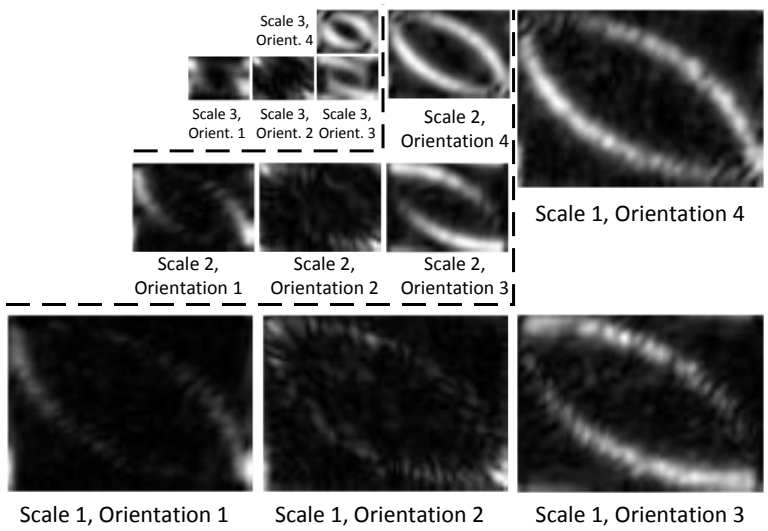


(b) Imaginary Parts of the Decomposed Images at the Same Scale and Orientation.

Figure 6.6: Real and Imaginary Parts of a 3-scale, 4-orientation Complex Steerable Pyramid Representation of AOI-2.



(a) Phase of Oriented Band-pass Images at Each Scale and Orientation.



(b) Amplitude of the Decomposed Images at the Same Scale and Orientation.

Figure 6.7: Phase and Amplitude of a 3-scale, 4-orientation Complex Steerable Pyramid Representation of AOI-2.

caused by different keystrokes. Fig. 6.4b shows the backside image of an iPad 2 placed on a holder, where the areas specified by rectangles are the selected AOIs.

6.5.4 Decompositions of Selected AOIs

Now we have a series of frames extracted from the cropped video, each containing the same set of selected AOIs. For each frame, we use complex steerable pyramid decomposition [121] to decompose each selected AOI into complex sub-bands. As an example, Fig. 6.5 shows an AOI that is part of the Apple logo, and Fig. 6.6 and Fig. 6.7 show the image decomposed via complex steerable pyramid decomposition. More specifically, Figs. 6.6a to 6.7b show the real part, imaginary part, phase, and amplitude of the decomposed image at three scales and four orientations, respectively. We can see from Fig. 6.7a that the image has more features at orientation 3 and 4 and less features at orientation 2. Since the phase variations are proportional to subtle motions, Fig. 6.7a indicates that more subtle motions can be detected at orientation 3 and 4. Finally, we obtain a decomposed complex steerable pyramid for each selected AOI in each frame.

6.5.5 Motion Detection via Phase Variances

To estimate the motion for each selected AOI over time, we first compute the pixel-level motion. As in [47, 149], for each selected AOI, we first decompose its frame series using complex steerable pyramid and then compute the pixel-level motion from the amplitude and phase of its pixels. Specifically, complex steerable pyramid decomposition adopts a filter bank to decompose each frame into complex sub-bands corresponding to each scale and orientation. The complex steerable pyramid decomposition of a frame at time t at scale r and orientation θ can be written as

$$A(t, x, y, r, \theta)e^{i\phi(t,x,y,r,\theta)}, \tag{6.1}$$

where x and y are the pixel coordinates in x-axis and y-axis at scale r, respectively, and $A(t, x, y, r, \theta)$ and $\phi(t, x, y, r, \theta)$ are the amplitude and phase at coordinate (x, y) of the decomposition at scale r and orientation θ, respectively.

We then calculate the phase variance at scale r and orientation θ as

$$\Delta\phi(t, x, y, r, \theta) = (\phi(t, x, y, r, \theta) - \phi(t_0, x, y, r, \theta)) \bmod 2\pi, \tag{6.2}$$

where t_0 is the time for any initial frame. According to [59], the phase variations $\Delta\phi(t, x, y, r, \theta)$ are approximately proportional to displacements to image structures along the corresponding scale and orientation.

Finally, we estimate each selected AOI's motion using its pixel-level motion. Since the pixel-level phase variance $\Delta\phi(t, x, y, r, \theta)$ is an approximation of the pixel-level motion, an intuitive way to estimate the motion of the AOI is to sum the phase variation $\Delta\phi(t, x, y, r, \theta)$ of all its pixels. However, the pixel-level phase variance approximates the pixel-level motion only if the area has rich texture information. For areas with little texture information, the pixel-level phase variance is random due to background noise. To simultaneously strengthen the pixel-level phase variation for areas with rich texture information and weaken the pixel-level phase variation for areas with little texture information, we compute a weighted sum of phase variances at scale r and orientation θ as

$$\Phi(t, r, \theta) = \sum_{x,y} A(t, x, y, r, \theta)^2 \Delta\phi, \tag{6.3}$$

where $A(t, x, y, r, \theta)$ is the measure of texture strength.

Since a frame is decomposed into multiple scales and different orientations, we sum the motions for all the scales and orientations to obtain the estimated motion for the specific AOI as

$$\Psi(t) = \sum_{r,\theta} \Phi(t, r, \theta) = \sum_{r,\theta,x,y} A(t, x, y, r, \theta)^2 \Delta\phi. \tag{6.4}$$

Fig. 6.8 depicts the motion signals of the apple stem in Fig. 6.5 during a word-entry process. We can see the typed word with thirteen letters each corresponding to a peak in amplitude, i.e., a sudden significant change in $|\Psi(t)|$.

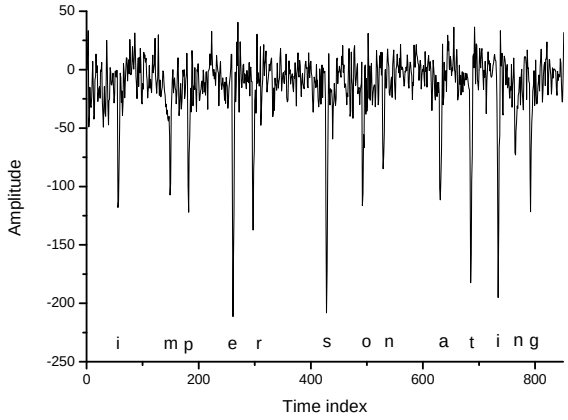


Figure 6.8: Motions of the Apple Stem in Fig. 6.5 for Typing "Impersonating".

6.5.6 Feature Extraction

Now we extract temporal and spatial features from selected AOIs' motion signals to represent the motion patterns. The former are obtained from the motion signals' time domain, and spatial features depict the motion relationship among different AOIs that is capable of reflecting the posture of the tablet.

To extract temporal features, we represent the motion sequence of each AOI as a vector that specifies the time-varying motion amplitude and then derive the following features for each AOI.

- *Skewness*. This refers to the third central moment which measures the asymmetry of the vector.
- *Kurtosis*. This is the fourth central moment which measures the peakedness or flatness of the vector.

- *Maximum motion amplitude.* The maximum motion amplitudes are different for different AOIs.

- *Relative and absolute differences between maximum motion amplitudes.* Assume that there are n selected AOIs. We define a ratio vector which comprises the ratio of the maximum motion amplitude of the i-th AOI to that of the $(i+1)$-th AOI for all $i \in [1, n-1]$. We also define a *difference vector* comprising the maximum motion amplitude of the i-th AOI subtracted by that of the $(i+1)$-th AOI for all $i \in [1, n-1]$.

To extract spatial features, we denote the motions of all AOIs by matrix $A_{m\times n}$, where m is the time index, n is the number of AOIs, and $A_{i,j}$ is the j-th AOI's motion amplitude at time i. We derive the following spatial features.

- *1-norm.* The 1-norm of $A_{m\times n}$ is calculated as

$$||A_{m\times n}||_1 = \max_{1\le j\le n} \sum_{i=1}^{m} |a_{ij}| \ ,$$

  which is its maximum absolute column sum.

- *2-norm.* As in [109] we calculate three 2-norm features from $A_{m\times n}$. Let R_i denote the ith row of $A_{m\times n}$ for all $i \in [1, m]$. The 2-norm of each R_i is given by

$$||R_i||_2 = \sqrt{\sum_{j=1}^{n} |a_{ij}|^2}.$$

  We then extract the mean, maximum, and minimum from

$$[||R_1||_2, ||R_2||_2, \ldots, ||R_m||_2]^T.$$

- *Infinity-norm.* The infinity-norm of $A_{m\times n}$ is

$$||A_{m\times n}||_\infty = \max_{1\le i\le m} \sum_{j=1}^{n} |a_{ij}| \ ,$$

  which is its maximum absolute row sum.

- *Frobenium-norm*. The frobenium-norm is calculated as

$$||A_{m\times n}||_F = \sqrt{\sum_{i=1}^{m}\sum_{j=1}^{n}|a_{ij}|^2}\ ,$$

  which is the square root of the squared sum of the matrix elements.

- *Pearson correlation*. The Pearson correlation measures the correlation of the motion vectors of different AOIs during the same typing process. For two motion vectors V_i and V_j of two AOIs i and j, respectively, the Pearson correlation is defined as

$$P_{ij} = \frac{\mathsf{cov}(V_i, V_j)}{\sigma_i \sigma_j},$$

  where $\mathsf{cov}(V_i, V_j)$ is the covariance between V_i and V_j, and σ_i and σ_j are the standard deviation of V_i and V_j, respectively.

6.5.7 Classifier Training

To train a classifier, we first reconstruct the attack scenario from the images taken in the video recording phase using standard distance and angle estimation algorithms such as [148].

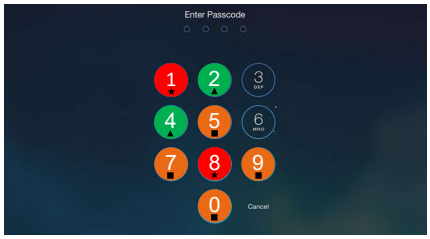
We then let multiple attackers type on every key position of the soft keyboard for multiple times, during which we record the videos of the tablet backside as well as the typed keys. We finally obtain the training data set consisting of NKM samples, where N is the number of attackers that mimic the victim, K is the number of keys on the soft keyboard, and M is the number of times each key is typed by each attacker.

We use a multi-class Support Vector Machine (SVM) [37] with C-SVC type and linear kernel to distinguish different typed keys. Specifically, we use the implementation in WEKA [63] with default parameters. Since we have already obtained NKM labeled typing samples, we feed them into WEKA to obtain a trained multi-class SVM classifier.

6.5.8 *Keystroke Inference*

In this step, we use the motion data extracted from the video recording of the victim's tablet backside and the trained classifier to obtain a candidate key set for each key the victim typed. Specifically, for a backside video capturing the victim's typing process, it is processed through Steps 1 to 5 to output 36 features in total, including four skewness features, four kurtosis features, four maximum motion features, six relative difference features, six absolute difference features, six Pearson correlation features, one one-norm feature, three two-norm features, one infinity-norm feature, and one Frobenium-norm feature. We then use the trained multi-class SVM classifier to predict one key. Since the distance between two adjacent keys in both alphabetical and PIN keyboards is very small, it is possible for the key entered by the victim to be misclassified as neighboring keys. We therefore let the SVM classifier output a candidate key set consisting of all the keys that are no more than h hops from the predicated key, where h is a parameter determined by the attacker.

(a) One-hop and Two-hop Neighbors of Letters "a" and "j".



(b) One-hop Neighbors of Keys 1 and 8.

Figure 6.9: Examples of One and Two-hop Neighbors on Alphabetical and PIN Keyboards.

6.5.9 Text Inference

In this step, we further infer the entered text by using a dictionary and a linguistic relationship between adjacent words. Specifically, for a word consisting of W letters, we can obtain W candidate letter sets in the previous step. We use the "corn-cob" dictionary [11] that contains over 58,000 lower-case English words and is also used by previous work [29]. First, we list all the combinations of the possible words and filter out the combinations that are not in the dictionary. We then manually select one word from each of the candidate lists to form a meaningful sentence by considering the linguistic relationship between adjacent words. As an alternative, given the candidate word list for each word, we may use a well-studied n-gram model such as [34] generated from linguistic statistics to generate candidate sentences.

6.6 Performance Evaluation

In this section, we evaluate the performance of VISIBLE through extensive experiments on a 9.7-inch Apple iPad 2 tablet with iOS 8 and a 7-inch Google Nexus 7 tablet with Android 4.4. The experiments involved eight participants in total, and the data collection process has been approved by Institutional Review Board (IRB) at our institution. We intend to answer the following five questions in our evaluations.

1. What is the (single-)key inference accuracy on alphabetical and PIN keyboards, respectively?

2. What is the word inference accuracy on the alphabetical keyboard?

3. Is it possible to infer a victim's typing sentences?

4. How do the inference results differ on different tablets (e.g., an iPad 2 and a Nexus 7)?

5. What are the impacts of environmental factors (e.g., the light conditions, the angle between the tablet and the camcorders, and the imperfect reconstruction of attack scenario) on keystroke inference accuracy?

6.6.1 Experiment Design

In our experiment, we used two commercial off-the-shelf (COTS) camcorders to video-record the tablet backside during the victim's typing process. One camcorder is a Panasonic HC-V700 with 21× zoom lens, which can record 1080p60 HD videos and feature an intelligent zoom function that supports up to 46× zoom. The second camcorder is a Sony FDR-AX100 with 10× zoom lens, which can record 4Kp30 [2] or 1080p60 HD videos and support up to 160× zoom.

We placed an Apple iPad 2 tablet with iOS 8 on a holder as shown in Fig. 6.4b and two camcorders 1.8 meters away from the tablet. The distance between the attacker's camcorders and the victim's tablet can be increased as long as the attacker is equipped with more advanced lens (e.g., telephoto lens) to video-record the tablet backside at a distance. The angle between each camcorder and the tablet was 90 degree by default, and we evaluated the impact of different angles as well. The two camcorders focused on the left-half and right-half of the tablet backside, respectively. We simultaneously used two camcorders because one camcorder cannot simultaneously include all the AOIs and have sufficiently high resolution for each AOI.

Let $\Omega^h(i)$ be key i's h-hop neighborhood, including key i itself. As two examples, Fig. 6.9a shows the one-hop and two-hop neighbors of letters "a" and "j", where the orange and yellow keys (marked by triangle and square) are the one-hop and two-hop neighbors, respectively. Fig. 6.9b shows the one-hop neighbors of keys 1 and 8, where

[2] 4Kp30 denotes that the camcorder can take 3840×2160 video at a rate of 30 frames per second.

the green and orange keys (marked by triangle and rectangle) are the neighbors of key 1 and key 8, respectively.

We use the following two metrics to evaluate the inference accuracy of VISIBLE.

- *Pinpoint accuracy* P_i. The probability that a key i typed by the victim is correctly inferred as i.
- *h-hop accuracy* P_i^h. The probability that a key i typed by the victim is inferred as some key in $\Omega^h(i)$.

By letting $\Omega^0(i) = \{i\}$, we can see that the pinpoint accuracy is a special case of h-hop accuracy, as $P_i = P_i^0$. We consider both pinpoint and h-hop accuracies for two reasons. First, the capability of narrowing down a typed key to a small area still poses a serious threat to user privacy, as the attacker can still learn sensitive information. Second, considering the neighborhood of a key instead of only the key itself is particularly important for word and sentence inference, as we will see shortly. In this chapter, we consider $h = 0, 1, 2$, and 3 for alphabetical keyboard and $h = 0$ and 1 for PIN keyboard.

6.6.2 Alphabetical Keyboard Experiment

We first report the performance of VISIBLE on the alphabetical keyboard of an iPad 2 tablet with iOS 8, on which keystroke inference is challenging for two reasons. First, the distance between two adjacent keys is very small, while we need to distinguish at least 26 different keys. Second, the alphabetical keyboard is usually located at the bottom of the touchscreen which makes the motions caused by keystrokes less noticeable.

In this experiment, we involved four participants and let each participant type each English letter 20 times and collected $20 \times 26 \times 4 = 2080$ keystrokes in total. We

Table 6.1: Key Inference Results for Alphabetical Keyboard, Where VIS and RG Denote VISIBLE and Random Guess, Respectively.

Key	P_i		$\|\Omega^1(i)\|$	P_i^1		$\|\Omega^2(i)\|$	P_i^2		$\|\Omega^3(i)\|$	P_i^3	
	VIS	RG		VIS	RG		VIS	RG		VIS	RG
a	33.8%	3.84%	5	78.8%	19.2%	8	95.0%	30.7%	11	100%	42.2%
b	36.3%	3.84%	5	78.8%	19.2%	14	98.8%	53.8%	18	100%	69.1%
c	52.5%	3.84%	5	71.3%	19.2%	14	93.8%	53.8%	18	98.8%	69.1%
d	21.3%	3.84%	8	91.3%	30.7%	14	98.8%	53.8%	17	100%	65.3%
e	22.5%	3.84%	6	70.0%	23.0%	14	98.8%	53.8%	17	98.8%	65.3%
f	27.5%	3.84%	8	91.3%	30.7%	14	98.8%	53.8%	20	100%	76.8%
g	25.0%	3.84%	8	88.8%	30.7%	14	98.8%	53.8%	20	100%	76.8%
h	16.3%	3.84%	8	95.0%	30.7%	14	100%	53.8%	19	100%	73.0%
i	21.3%	3.84%	6	85.0%	23.0%	11	100%	42.2%	14	100%	53.8%
j	20.0%	3.84%	8	83.8%	30.7%	13	98.8%	49.9%	17	100%	65.3%
k	22.5%	3.84%	7	88.8%	26.9%	11	98.8%	42.2%	14	100%	53.8%
l	42.5%	3.84%	5	85.0%	19.2%	9	100%	34.6%	12	100%	46.1%
m	50.0%	3.84%	4	80.0%	15.4%	11	98.8%	42.2%	15	100%	57.6%
n	31.3%	3.84%	5	77.5%	19.2%	13	97.5%	49.9%	17	100%	65.3%
o	41.3%	3.84%	5	88.8%	19.2%	8	98.8%	30.7%	11	100%	42.2%
p	47.5%	3.84%	3	81.3%	11.5%	7	98.8%	26.9%	8	100%	30.7%
q	30.0%	3.84%	4	70.0%	15.4%	8	90.0%	30.7%	11	98.8%	42.2%
r	40.0%	3.84%	6	80.0%	23.0%	14	95.0%	53.8%	20	97.5%	76.8%
s	28.8%	3.84%	8	76.3%	30.7%	11	95.0%	42.2%	14	100%	53.8%
t	30.0%	3.84%	6	86.3%	23.0%	14	97.5%	53.8%	20	100%	76.8%
u	51.3%	3.84%	6	97.5%	23.0%	13	100%	49.9%	17	100%	65.3%
v	45.0%	3.84%	6	83.8%	23.0%	12	98.8%	46.1%	19	100%	73.0%
w	31.3%	3.84%	6	72.5%	23.0%	11	95.0%	42.2%	14	100%	53.8%
x	41.3%	3.84%	6	92.5%	23.0%	11	100%	42.2%	15	100%	57.6%
y	27.5%	3.84%	6	81.3%	23.0%	12	98.8%	46.1%	19	100%	73.0%
z	77.5%	3.84%	4	98.8%	15.4%	9	100%	34.6%	12	100%	46.1%
Avg.	36.2%	3.84%	5.9	83.6%	22.7%	11.7	97.9%	44.9%	15.7	99.8%	60.3%

selected a portion of data from the collected dataset as a training set and used the rest of the collected data as a test set. We trained a multi-class SVM classifier using the training set and then tested it using the test set. We used 10-fold cross-validation to test the performance of the key inference of VISIBLE.

Table 6.1 compares the pinpoint and h-hop accuracies of VISIBLE and random guess for each English letter on the alphabetical keyboard. We can see that the pinpoint accuracy of VISIBLE for each key ranges from 16.3% for letter "h" to 77.5% for letter "z". The average pinpoint accuracy of VISIBLE across all 26 letters is 36.2%, which is almost an order of magnitude higher than 3.84% of random guess. Since different keys' h-hop neighborhoods have different sizes for the same h, we calculate each P_i^h for random guess based on the actual number of keys within key i's h-hop neighborhood (i.e., $|\Omega^h(i)|$) to ensure fair comparisons. We can see that for both VISIBLE and random guess, the P_i^h for each key i increases as h increases, which is expected, as the larger the neighborhood being considered, the higher the probability that a key inferred as some key in the neighborhood, and vice versa. Moreover, the $P^h(i)$ of VISIBLE is always significantly higher than the corresponding $P^h(i)$ of random guess. We also calculate the average P_i^1, P_i^2, and P_i^3 of VISIBLE across all 26 keys as 83.6%, 97.9%, and 99.8%, respectively, which are much higher than corresponding 22.7%, 44.9%, and 60.3% of random guess. Meanwhile, note that the average P_i^h may only be used with caution to compare the performance of two different techniques, due to the difference in the size of keys' neighborhood.

We also notice that pinpoint and h-hop accuracies of the letters at corner positions (i.e., "q", "z", "p", and "m") are higher than those of the letters at the center (e.g., "g" and "h"). This is because typing the letters at corner positions causes more distinguishable motion patterns than those at the center. Moreover, we can see that the pinpoint and h-hop accuracies of letter "z" are much higher than those of other

three letters at the corner positions. The reason behind such disparity is that our selected AOIs are not evenly distributed. As shown in Fig. 6.4b, the distances between letter "z" and selected AOIs are greater than those of other letters, and typing "z" thus causes more distinguishable motion patterns.

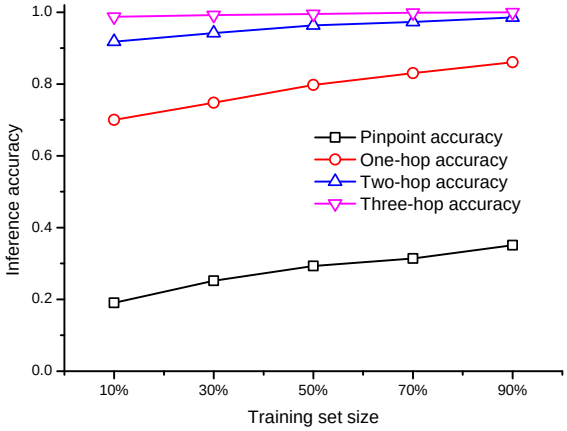


Figure 6.10: Impact of the Training Set Size.

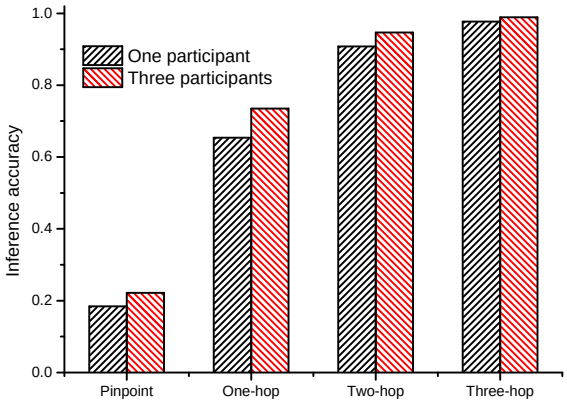


Figure 6.11: Impact of the Number of Participants.

Fig. 6.10 shows the impact of the training set size on the inference accuracy. As expected, increasing the training set size can slightly improve the key inference accuracy. Fig. 6.11 shows the impact of the number of participants in the training set. We can see that as the number of participants increases, the key inference accuracy

slightly increases in all the cases. In addition, a small number of participants is sufficient for achieving acceptable key inference accuracy, which means that VISIBLE requires very few attackers to launch.

6.6.3 Word Inference Experiment

We now report the experimental results of the word inference attack on the iPad 2 tablet. In this experiment, we involved two participants and let each participant enter each word in Table 6.2 (which is also used in [29]) once to evaluate the word inference accuracy. In total, we collected 2×27 words with 7∼13 letters, where all the letters in the words are in lower-case. As in [29], we used the "corn-cob" dictionary [11] consisting of more than $58,000$ English words. For each letter in each word, we first used the trained multi-class SVM classifier to predict one key and obtained a candidate key set consisting of all the keys that are less than two hops from the predicated key. Then for each word, we obtained a candidate word list by filtering out the combinations that are not in the "corn-cob" dictionary [11].

Table 6.2: List of Words Used to Test the Attack.

Word	Length	Word	Length	Word	Length	Word	Length
paediatrician	13	pomegranate	11	unphysical	10	platinum	8
interceptions	13	feasibility	11	institute	9	homeland	8
abbreviations	13	polytechnic	11	extremely	9	security	8
impersonating	13	obfuscating	11	sacrament	9	between	7
soulsearching	13	difference	10	dangerous	9	spanish	7
hydromagnetic	13	wristwatch	10	identity	8	nuclear	7
inquisition	11	processing	10	emirates	8		

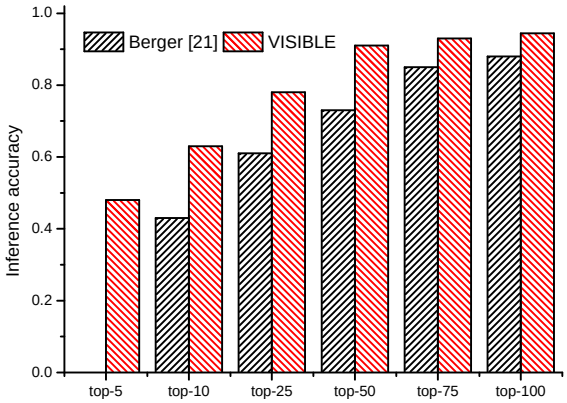


Figure 6.12: Word Inference Accuracy.

Fig. 6.12 compares the overall word inference accuracy of VISIBLE and the technique proposed in [29] for the tested words in Table 6.2. To enable direct comparison, we view the size of the candidate word list output by VISIBLE as the lowest possible rank of the correct word in the candidate word list if the correct word is in candidate word list. In other words, if a candidate word list of c words contains the correct word typed by the victim, then we say the correct word is among the top-k candidate words for any $k \geq c$. As shown in Fig. 6.12, the correct word is among the top-5 candidate words output by VISIBLE 48% of the time, which means that nearly half of the words in Table 6.2 have a candidate word list with no more than 5 words. Besides, we can see that the correct word is among top-10, top-25, and top-50 candidate words with probabilities 63%, 78%, and 93%, respectively. In contrast, the technique in [29] infers the correct word in top-10, top-25, and top-50 candidate words with probabilities 43%, 61%, and 73%, respectively. VISIBLE thus achieves much higher accuracy for word inference than [29].

Fig. 6.13 shows that word inference accuracy increases as the word length increases. Two reasons account for this trend. First, a longer word has more constraints in letter combinations and thus fewer candidate words in the dictionary. Second, ac-

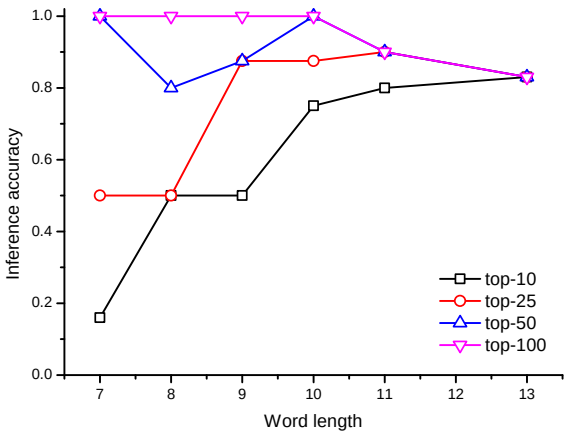


Figure 6.13: Word Inference Accuracy vs. Word Length.

cording to statistics of English words, the number of words of length seven is the largest among all words, so words with seven letters have the most candidate words in the dictionary, which leads to a lower inference accuracy.

6.6.4 Sentence Inference Experiment

Next, we report VISIBLE's performance for inferring complete sentences on the iPad 2 tablet. For this experiment, we used Enron Email Dataset [12, 14, 76], which comprises over 600,000 emails generated by 158 employees of the Enron Corporation and is also used in previous work [120] to test the sentence inference accuracy. We asked one participant to select two sentences from the dataset and enter the selected sentences using the alphabetical keyboard of the iPad 2 tablet. The attacker video-recorded the sentence-entry process using two camcorders and used a multi-class SVM classifier trained by the keystroke data. The attacker then performed word inference to obtain a candidate word list for each word and finally chose one word from each candidate word list to form a meaningful sentence based on the linguistic relationship between adjacent words.

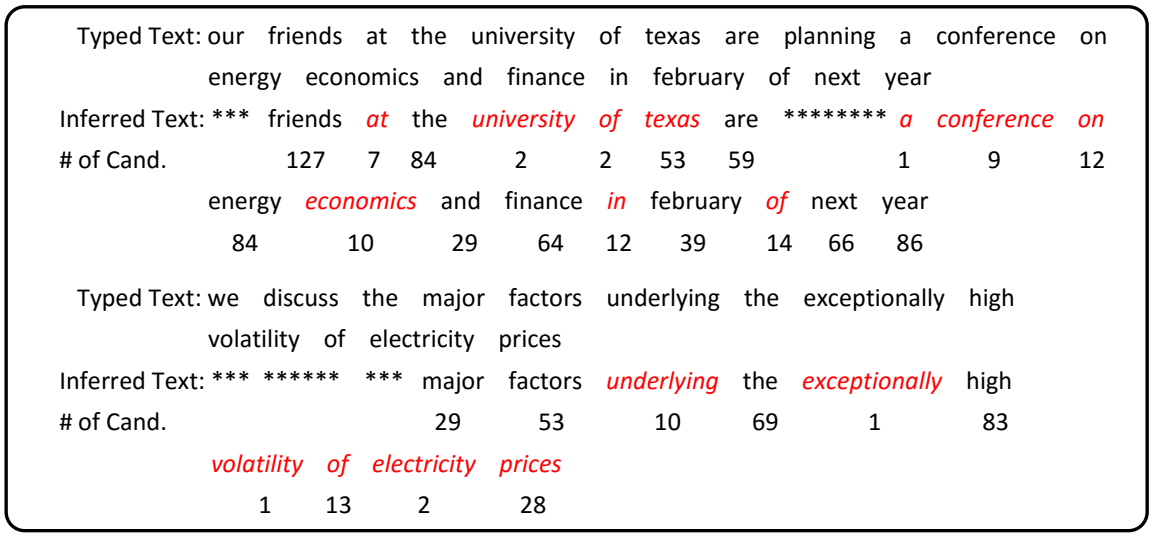


Figure 6.14: Sentence Inference Results.

Fig. 6.14 illustrates the input sentences and the results inferred by VISIBLE. The number under each word is the number of candidate words output by VISIBLE. The red italic words are the ones correctly chosen by the attacker. The black non-italic words are the ones in the candidate word list but hard to choose. Symbol "*" indicates that the corresponding word is not correctly inferred during word inference. More detailed investigations find that the incorrectly inferred words are due to one or two misclassified letters. We expect that the sentence inference accuracy of VISIBLE can be dramatically improved by incorporating more advanced linguistic models.

6.6.5 PIN Keyboard Experiment

We now evaluate the key inference performance on the PIN keyboard of the iPad 2 tablet. In this experiment, we involved three participants. Intuitively, the key inference on the PIN keyboard is more difficult than that on the alphabetical keyboard for mainly two reasons. First, all the keys are located in a relatively small area in the central part of the touchscreen. Second, the typed keys are very likely to be

random, and there are no relationships (e.g., linguistic relationship) between adjacent keystrokes.

Table 6.3: Key Inference Results for PIN Keyboard.

Key	P_i		$\Omega^1(i)$	$\lvert\Omega^1(i)\rvert$	P_i^1	
	VIS	RG			VIS	RG
1	21%	9%	1, 2, 4	3	58%	27%
2	25%	9%	1, 2, 3, 5	4	75%	36%
3	45%	9%	2, 3, 6	3	63%	27%
4	55%	9%	1, 4, 5, 7	4	81%	36%
5	40%	9%	2, 4, 5, 6, 8	5	66%	45%
6	35%	9%	3, 5, 6, 9	4	64%	36%
7	44%	9%	4, 7, 8	3	73%	27%
8	23%	9%	5, 7, 8, 9, 0	5	53%	45%
9	27%	9%	6, 8, 9, c	4	61%	36%
0	47%	9%	0, 8, c	3	72%	27%
c	61%	9%	0, 9, c	3	80%	27%
Avg.	38%	9%	-	4	68%	36%

Table 6.3 compares the pinpoint and 1-hop accuracies of VISIBLE and random guess for each key on the PIN keyboard. We can see that the pinpoint accuracy of each key ranges from 21% for number "9" to 61% for "c" cancel key. The average pinpoint accuracy of VISIBLE across all 26 letters is 38%, which is more than four times of that of random guess, i.e., $\frac{100}{11} = 9\%$. When considering one-hop neighborhood, the P_i^1 of VISIBLE for each key is still much higher than that of random guess. We can

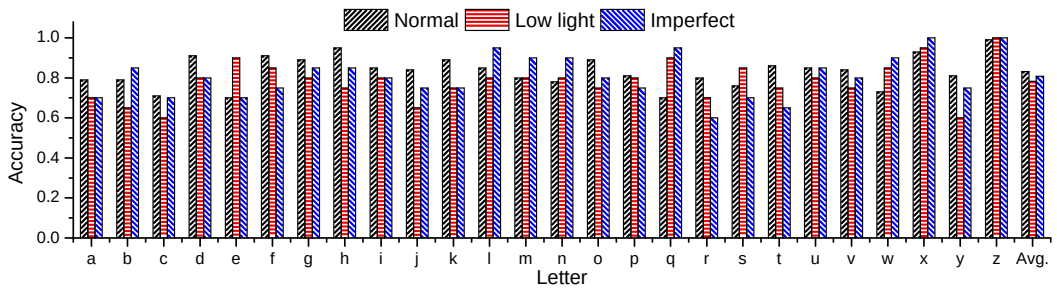
see that the average P_i^1 of VISIBLE across all 11 keys is 68%, which is much higher than 36% of random guess. Again, the average P_i^1 should be only used with caution to compare the inference accuracies of two techniques.

Moreover, comparing Tables 6.1 and 6.3, we can see that although the PIN keyboard has fewer keys than the alphabetical keyboard, the key inference accuracy of the PIN keyboard is not dramatically higher than that of the alphabetical keyboard. The reason is that the keys of the PIN keyboard reside in a relatively small area in the central part of the touchscreen, and the motion patterns caused by different keys are not so distinguishable. In contrast, even though the alphabetical keyboard has more keys, they are located in a relatively large area from the left side to the right side of the touchscreen. The motion patterns of different keys, especially distant ones, cause more distinguishable motion patterns.

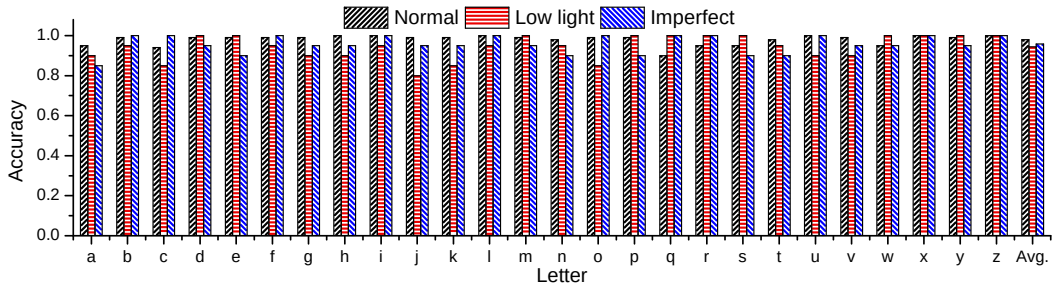
6.6.6 Impact of Environmental Factors

We also evaluate the impact of a number of environmental factors on the performance of VISIBLE.

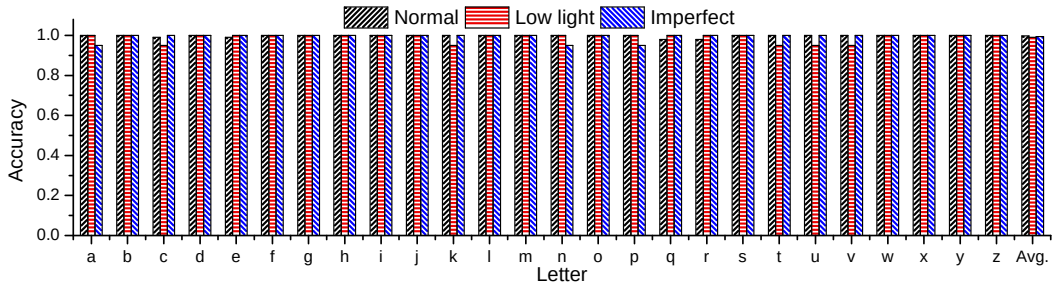
a). Different Light Conditions. Our attack relies on analyzing the video recordings of the tablet backside during the victim's typing process, while the video quality is affected by light conditions. In general, the low-light condition will lead to increased video noise, streaking, blurred motion, and poor focus. We did key inference experiments under light conditions of 400 lux (normal) and 180 lux (low light). Fig. 6.15 shows the key inference results for each key. We can see that the key inference accuracy decreases slightly as the light condition changes from 400 lux to 180 lux, which is expected. However, the key inference result under 180 lux is still quite acceptable, which highlights the wide applicability of VISIBLE in low-light conditions.



(a) One-hop Accuracy.

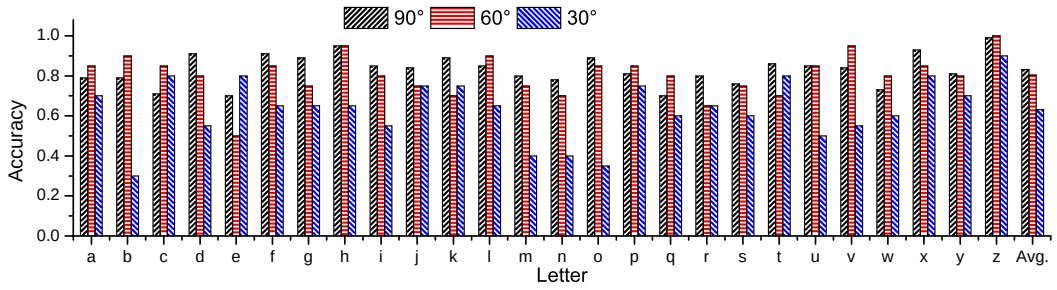


(b) Two-hop Accuracy.

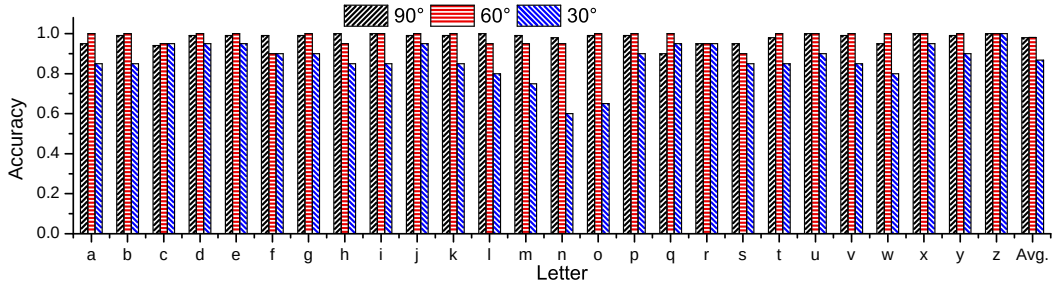


(c) Three-hop Accuracy.

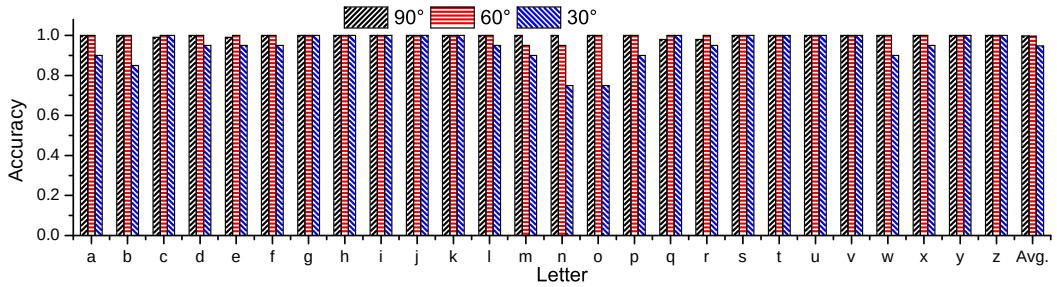
Figure 6.15: Alphabetical Keyboard Inference Accuracy Under Different Light Conditions and Imperfect Reconstruction of the Attack Scenario.



(a) One-hop Accuracy.



(b) Two-hop Accuracy.



(c) Three-hop Accuracy.

Figure 6.16: Alphabetical Keyboard Inference Accuracy for Different Angles Between the Tablet and Camcorders.

b). Different angles between camcorders and the tablet. The performance of VISIBLE is also affected by the angles between the camcorders and the tablet. In previous experiments, the angle between the camcorders and tablet was 90 degree. We changed the angle to 60 and 30 degrees while keeping the distance between the camcorders and tablet unchanged. The experimental result is shown in Fig. 6.16 for each key's inference accuracy by considering one-hop, two-hop, and three-hop neighbors. We can see that in each of three subfigures, 90 and 60 degree angles lead to similar key inference accuracy which is nevertheless better than that of the 30 degree angle. The reason is as follows. Each camcorder has a specific Depth of Field (DOF) that is the distance between the nearest and farthest objects in a scene that appear acceptably sharp in an image. If the angle between the camcorders and the tablet is 90 or 60 degree, all the AOIs are in the DOF of the camcorders so their motions can be clearly recorded. However, if the angle between the camcorders and the tablet is too small, the camcorders cannot contain all the AOIs in their DOF, which leads to blurred AOI images and thus inaccurate estimation of tablet backside motions. If the angle has to be small due to practical constraints, the attacker can use multiple camcorders to record the motions of different AOIs to obtain sharp image of each AOI.

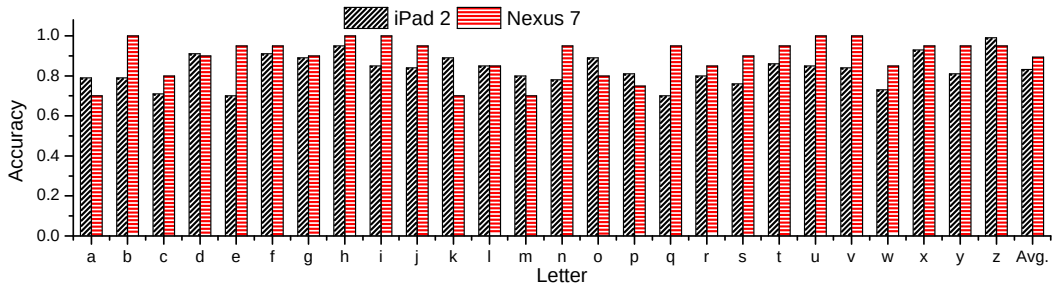
c). Imperfect reconstruction of the attack scenario. As mentioned in Section 6.5.2, to launch a successful key inference attack, the attacker needs to reconstruct the attack scenario based on recorded images. However, the reconstructed layout cannot be exactly the same as the true layout. We therefore evaluate the impact of imperfect reconstruction of the attack scenario. For this experiment, we changed the location of the camcorders randomly by five centimeters and the position of the tablet by three centimeters and then redid the key inference experiment. Fig. 6.15 shows the key inference accuracy when the attack scenario is not perfectly reconstructed. We can see that the key inference accuracy for each key is only slightly lower than that

under perfect reconstruction, which shows the robustness of VISIBLE against small environment change. Note that attack scenario reconstruction is under the full control of the attacker and does not involve the victim. Its accuracy depends only on the quality of the recorded images, and we expect the reconstructed attacker scenario to be accurate in practice. On the other hand, if the environment changes significantly during video recording, e.g., the victim changes position or moves the tablet for more than 10 centimeters, the attacker may need to launch a new round of attack to obtain accurate inference result.

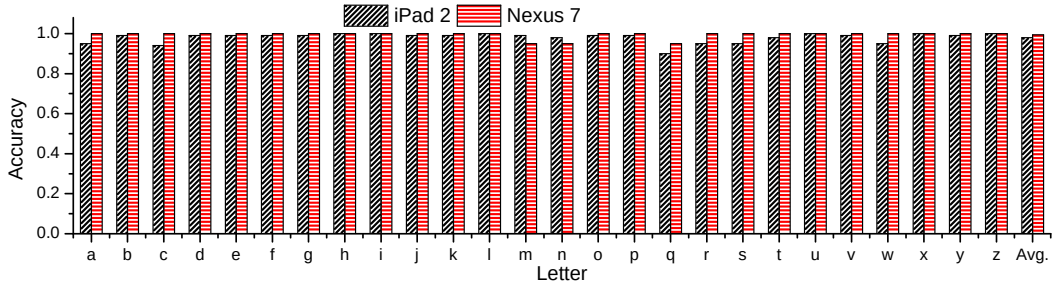
6.6.7 Experiments on a Google Nexus 7 Tablet

To demonstrate the universality of VISIBLE, we also did experiments on a Google Nexus 7 tablet with a 7-inch touchscreen which is smaller than that of an iPad 2. Backside motion estimation on Nexus 7 is easier than that on an iPad 2 tablet for two reasons. First, the size of Nexus 7 is smaller than that of iPad 2, so we were able to video-record the clear tablet backside motion with only one camcorder. Second, the Nexus 7's backside has more texture information (e.g., logo and dots) which enables motion estimation at more parts of the tablet backside.

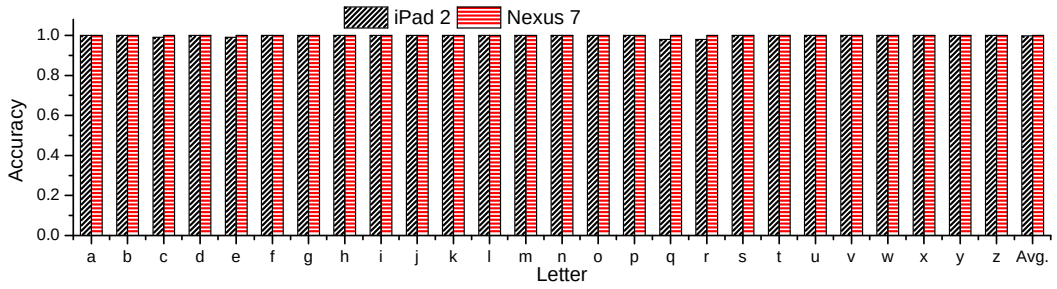
Fig. 6.17 compares the performance of VISIBLE on a Google Nexus 7 tablet with Android 4.4 with that on an iPad 2 tablet with iOS 8. It is easy to see that the key inference accuracy of VISIBLE is similar on both tablets. This means that VISIBLE is applicable to smaller-size tablets as long as there are sufficient areas with texture information on the tablet backside, which holds for almost all tablets. Besides, we can find that the performance on Nexus 7 is slightly better than that on iPad 2. The reason is that the Nexus 7's backside has more texture information for the attacker to extract motion information, while the iPad 2's backside has less texture information (as shown in Fig. 6.4a). As mentioned in Section 6.5.3, AOIs near the edges of the



(a) One-hop Accuracy.



(b) Two-hop Accuracy.



(c) Three-hop Accuracy.

Figure 6.17: Alphabetical Keyboard Inference Accuracy on a Google Nexus 7 Tablet and an iPad 2 Tablet.

tablet backside and separated from each other tend to have larger and more distinctive motions than others, making them more capable of differentiating the motion patterns caused by different keystrokes. Therefore, VISIBLE performs better on the tablets with rich texture information on their backsides.

6.7 Summary, Countermeasures, and Future Work

In this chapter, we propose VISIBLE, a video-assisted key inference attack framework to infer the victim's typing content based on the video recordings of the tablet backside. We adopt complex steerable pyramid decomposition to obtain the subtle motions on the tablet backside and use machine learning techniques to infer the typed keys, words, and sentences. We thoroughly evaluate the performance of VISIBLE via extensive experiments. Our results show that VISIBLE can achieve high key inference accuracy for both PIN and alphabetical soft keyboards and correctly infer the victim's typed words or sentences with very high probability.

There are several possible countermeasures against VISIBLE. The most straightforward defense is to design a large featureless cover to cover the stand or the tablet to prevent the attacker from finding useful AOIs in the recorded video. The second possible defense is to randomize the layouts of the PIN and alphabetical soft keyboards, such that the attacker cannot recover the typed keys even if he can infer the keystroke positions on the touchscreen. This defense may be effective, but it sacrifices the user experience, as the user needs to find every key on a random keyboard layout during every key-typing process. Another possible defense is to have on-board vibrators generate vibrations during the typing process to mask the motions caused by the user's typing process. However, unlike smartphones, most current commercial off-the-shelf tablets are not equipped with on-board vibrators. The ultimate solution is to cover the whole tablet (both the front and back sides). Though most effective,

this solution is inconvenient and might be socially awkward. The investigation of these defenses is left as future work.

To the best of our knowledge, VISIBLE is the first attempt to utilize the backside motion of tablets for keystroke analysis. There are still many issues worth investigation in addition to the countermeasures above. As we mentioned in Section 6.5.2, higher resolutions can help us video-record more texture information details on the tablet backside, and higher frame rates could video-record more motion details overtime. We plan to test VISIBLE with more advanced camcorders with higher resolutions and frame rates. We also seek to investigate the impact of optical and digital zoom and the distance between camcorder and the victim's tablet. In this chapter, we only consider the lower-case letters in the English alphabet. In practice, more contents such as upper-case letters, punctuation, characters, and key combinations might be typed by the victim. Further studies on this issue are challenging but meaningful. Our current study assumes that the victim places the tablet with a holder on a desk, while another common scenario is to hold the tablet by hand. In this case, the motion of the tablet backside is the combination of the motions of the holding hand and the non-holding hand's keystrokes. Keystroke inference in this scenario is much more challenging because we need to cancel the time-varying motion of the holding hand. In VISIBLE, the attacker needs to reconstruct the attack scenario to obtain a training data set to infer the victim's typed inputs. Although feasible, it is not so convenient. A more attractive way is to build a unified and normalized model, which could automatically transfer motions video-recorded in different distances and angles to a unified and normalized distance and angle. This will greatly improve the convenience of launching our proposed attack and deserves further investigations.

Chapter 7

CONCLUSION AND FUTURE WORK

The rapid development of mobile devices and the popularity of various mobile application scenarios make mobile computing a more and more attractive and promising area. In this dissertation, we find a number of challenging security and privacy issues in mobile computing and propose some promising solutions and countermeasures. For D2D communications in mobile computing, we first focus on a privacy issue and demonstrate the a privacy-preserving and efficient way of establishing trust relationship between two mobile devices. Besides, we also consider the security problem of establishing shared secret key between mobile devices. A game-theoretical approach is proposed for stimulating PHY-based cooperative key generation in wireless networks. The incentive-aware cooperative key generation is formulated as a coalitional game and implemented by centralized and distributed protocols for finding a core solution to the coalitional game. With SYNERGY in place, selfish mobile nodes are strongly motivated to collaborate with others in the same coalition to improve their respective key generation rate. The efficacy and efficiency of SYNERGY have been confirmed by extensive simulations. As to mobile crowdsourcing, we first focus on designing the first secure and privacy-preserving object-finding system via mobile crowdsourcing which guarantees object security, mobile crowdsourcing users' privacy, and system efficiency. Detailed simulations confirm that SecureFind can enable very fast and efficient object finding while ensuring the security of the lost object and also the location privacy of the mobile users participating in object finding. We then study the privacy issue in distributed stream monitoring system and present a novel privacy-preserving and communication-efficient distributed stream monitoring

system. Different from previous work on monitoring the function of mean statistic value, our proposed scheme monitors the statistic value at the given percentile rank in a privacy-preserving and communication-efficient fashion. The efficacy and efficiency of our PriStream are confirmed by detailed simulations. Considering the key inference problem in mobile device usage, we demonstrate VISIBLE, a video-assisted key inference attack framework, to infer the victim's typing content based on the video recordings of the tablet backside. VISIBLE adopts complex steerable pyramid decomposition to obtain the subtle motions on the tablet backside and uses machine learning techniques to infer the typed keys, words, and sentences. The performance of VISIBLE is thoroughly evaluated via extensive experiments. Experimental results show that VISIBLE can achieve high key inference accuracy for both PIN and alphabetical soft keyboards and correctly infer the victim's typed words or sentences with very high probability.

This dissertation is far from perfectness. For the future work, this dissertation can be extended in following directions.

First, Chapter 2 studies the problems of unweighted and weighted spatiotemporal matching between two mobile users. A very promising direction is to extend the spatiotemporal matching problem from two mobile users to multiple mobile users while maintaining privacy and efficiency.

Second, Chapter 3 proposes a game-theoretical framework for cooperative shared secret key generation by involving a third mobile user. In order to better improve the key generation rate, we can study the problem of involving multiple users simultaneously as one of future work.

Third, Chapter 4 shows a privacy-preserving object-finding system via mobile crowdsourcing. There are still many open challenges to tackle. For example, in the current design, all the mobile detectors in the target area specified by the object

owner need to participate in object finding. Since some of them may have overlapping coverage, there may be significant room for reducing the communication and computation overhead. One possible solution is to let the service provider select the minimum number of mobile detectors that can jointly cover the target area. This solution, however, requires the service provider to know more accurate locations of mobile detectors. Such tradeoff between system efficiency and location privacy deserves careful investigation. In addition, our current design assumes that mobile detectors are honest-but-curious. There may be dishonest mobile detectors who report fake search results to earn reward without actually performing the object search. How to catch and then punish such dishonest mobile detectors is nontrivial and may conflict with the location-privacy requirement of mobile detectors. How to solve these problems can be further studied.

Fourth, Chapter 5 solves the problem of designing and implementing a privacy-preserving and communication-efficient distributed stream monitoring system. However, the scheme only works for one dimensional data. In practice, the data of distributed stream monitoring system might be in multiple dimensions. How to guarantee privacy and maintain efficiency for data in all dimensions is one of our future work.

Fifth, to the best of our knowledge, VISIBLE proposed in Chapter 6 is the first attempt to utilize the backside motion of tablets for keystroke analysis. There are still many issues worth investigation in addition to the countermeasures above. As we mentioned in Section 6.5.2, higher resolutions can help us video-record more texture information details on the tablet backside, and higher frame rates could video-record more motion details overtime. We plan to test VISIBLE with more advanced camcorders with higher resolutions and frame rates. We also seek to investigate the impact of optical and digital zoom and the distance between camcorder and the vic-

tim's tablet. In this chapter, we only consider the lower-case letters in the English alphabet. In practice, more contents such as upper-case letters, punctuation, characters, and key combinations might be typed by the victim. Further studies on this issue are challenging but meaningful. Our current study assumes that the victim places the tablet with a holder on a desk, while another common scenario is to hold the tablet by hand. In this case, the motion of the tablet backside is the combination of the motions of the holding hand and the non-holding hand's keystrokes. Keystroke inference in this scenario is much more challenging because we need to cancel the time-varying motion of the holding hand. In VISIBLE, the attacker needs to reconstruct the attack scenario to obtain a training data set to infer the victim's typed inputs. Although feasible, it is not so convenient. A more attractive way is to build a unified and normalized model, which could automatically transfer motions video-recorded in different distances and angles to a unified and normalized distance and angle. This will greatly improve the convenience of launching our proposed attack and deserves further investigations.

BIBLIOGRAPHY

[1] "Lost children fast facts", http://www.cnn.com/2013/10/22/us/lost-children-fast-facts/.

[2] "US phone theft", http://www.micro-trax.com/statistics/.

[3] "Tile", http://www.thetileapp.com/.

[4] "StickNFind", https://www.sticknfind.com/.

[5] "BlueBee", http://www.indiegogo.com/projects/bluebee-a-lost-and-found-in- your-pocket.

[6] "Population density of Austin", http://zipatlas.com/us/tx/austin/zip-code-comparison/population-density.htm.

[7] "Population density of Portland", http://zipatlas.com/us/or/portland/zip-code-comparison/population-density.htm.

[8] "Bluetooth low energy", https://en.wikipedia.org/wiki/Bluetooth_low_energy.

[9] "Battery energy consumption", http://www.bbc.com/future/story/20130227-what-is-killing- smartphones.

[10] "Cisco visual networking index: Global mobile data traffic forecast update, 20162021 white paper", http://www.cisco.com/c/en/us/solutions/collateral/service-provider/visual-networking-index-vni/mobile-white-paper-c11-520862.html.

[11] "corn-cob dictionary", Http://www.mieliestronk.com/wordlist.html.

[12] "Enron email dataset", Https://www.cs.cmu.edu/~./enron/.

[13] "Gartner: Device shipments break 2.4b units in 2014, tablets to overtake pc sales in 2015", Http://techcrunch.com/2014/07/06/gartner-device-shipments-break-2-4b-units-in-2014-tablets-to-overtake-pc-sales-in-2015/.

[14] "Parakweet lab's email intent data set", Https://github.com/ParakweetLabs/EmailIntentDataSet.

[15] G. Acs and C. Castelluccia, "I have a dream! (differentially private smart metering)", in "Information Hiding", pp. 118–132 (2011).

[16] E. Adelson, C. Anderson, J. Bergen, P. Burt and J. Ogden, "Pyramid methods in image processing", in RCA Engineer **29**, 6, 33–41 (1984).

[17] S. Agrawal, S. Deb, K. Naidu and R. Rastogi, "Efficient detection of distributed constraint violations", in "ICDE'07", pp. 1320–1324 (Delhi, India, 2007).

[18] R. Ahlswede and I. Csiszar, "Common randomness in information theory and cryptography. i. secret sharing", IEEE Transactions on Information Theory **39**, 4, 1121–1132 (1993).

[19] B. Alomair, A. Clark, J. Cuellar and R. Poovendran, "Scalable RFID systems: a privacy-preserving protocol with constant-time identification", in "DSN'10", pp. 1–10 (2010).

[20] C. Arackaparambil, J. Brody and A. Chakrabarti, "Functional monitoring without monotonicity", in "ICALP'09", (Rhodes, Greece, 2009).

[21] M. Arb, M. Bader, M. Kuhn and R. Wattenhofer, "VENETA: Serverless friend-of-friend detection in mobile social networking", in "WIMOB'08", pp. 184–189 (Avignon, France, 2008).

[22] G. Asharov, Y. Lindell, T. Schneider and M. Zohner, "More efficient oblivious transfer and extensions for faster secure computation", in "ACM CCS'13", (2013).

[23] B. Azimi-Sadjadi, A. Kiayias, A. Mercado and B. Yener, "Robust key generation from signal envelopes in wireless networks", in "CCS'07", (Alexandria, Virginia, USA, 2007).

[24] M. Backes, T. Chen, M. Duermuth, H. Lensch and M. Welk, "Tempest in a teapot: Compromising reflections revisited", in "S&P'09", (Oakland, CA, USA, 2009).

[25] M. Backes, M. Dürmuth and D. Unruh, "Compromising reflections-or-how to read lcd monitors around the corner", in "S&P'08", (Oakland, CA, USA, 2008).

[26] D. Balzarotti, M. Cova and G. Vigna, "Clearshot: Eavesdropping on keyboard input from video", in "S&P'08", (Oakland, CA, USA, 2008).

[27] O. Banos, R. Garcia, J. Holgado, M.Damas, H. Pomares, I. Rojas, A. Saez and C. Villalonga, "mhealthdroid: a novel framework for agile development of mobile health applications", in "IWAAL'14", (Belfast, Northern Ireland, 2014).

[28] H. Bao and R. Lu, "A new differentially private data aggregation with fault tolerance for smart grid communications", Internet of Things Journal, IEEE **2**, 3, 248–258 (2015).

[29] Y. Berger, A. Wool and A. Yeredor, "Dictionary attacks using keyboard acoustic emanations", in "CCS'06", (Alexandria, Virginia, USA, 2006).

[30] R. Bergman, H. Nachlieli and G. Ruckenstein, "Detection of textured areas in images using a disorganization indicator based on component counts", HP Laboratories Israel Technical Report (2007).

[31] B. Bloom, "Space/time trade-offs in hash coding with allowable errors", Comm. ACM **13**, 7, 422–426 (1970).

[32] P. Bose, H. Guo, E. Kranakis, A. Maheshwari, P. Morin, J. Morrison, M. Smid and Y. Tang, "On the false-positive rate of bloom filters", Inf. Process. Lett. (2008).

[33] A. Broder and M. Mitzenmacher, "Network applications of bloom filters: A survey", pp. 636–646 (2002).

[34] P. Brown, P. deSouza, R. Mercer, V. Pietra and J. Lai, "Class-based n-gram models of natural language", Comput. Linguist. **18**, 4, 467–479 (1992).

[35] L. Cai and H. Chen, "Touchlogger: Inferring keystrokes on touch screen from smartphone motion", in "HotSec'11", (Berkeley, CA, USA, 2011).

[36] T.-H. Chan, E. Shi and D. Song, "Privacy-preserving stream aggregation with fault tolerance", in "FC'12", pp. 200–214 (2012).

[37] C.-C. Chang and C.-J. Lin, "LIBSVM: A library for support vector machines", ACM Transactions on Intelligent Systems and Technology **2**, 3, 27:1–27:27, software available at `http://www.csie.ntu.edu.tw/~cjlin/libsvm` (2011).

[38] R. Chen, A. Reznichenko, P. Francis and J. Gehrke, "Towards statistical queries over distributed private user data", in "NSDI'12", (San Jose, CA, 2012).

[39] X. Chen, B. Proulx, X. Gong and J. Zhang, "Social trust and social reciprocity based cooperative d2d communications", in "MobiHoc'13", (Bangalore, India, 2013).

[40] Y. Chen, J. Sun, R. Zhang and Y. Zhang, "Your song your way: Rhythm-based two-factor authentication for multi-touch mobile devices", in "IEEE INFOCOM'15", pp. 2686–2694 (Hong Kong, China, 2015).

[41] Y. Chen, J. Sun, X. Jin, T. Li, R. Zhang and Y. Zhang, "Your face your heart: Secure mobile face authentication with photoplethysmograms", in "IEEE INFOCOM'17", (Atlanta, USA, 2017).

[42] H. Chernoff and E. L. Lehmann, "The use of maximum likelihood estimates in χ^2 tests for goodness of fit", Ann. Math. Statist. **25**, 3, 423–630 (1954).

[43] K. Christensen, A. Roginsky and M. Jimeno, "A new analysis of the false positive rate of a bloom filter", Inf. Process. Lett. (2010).

[44] G. Cormode and M. Garofalakis, "Approximate continuous querying over distributed streams", ACM Transactions on Database Systems **33**, 2, 9:1–9:39 (2008).

[45] T. Cover and J. Thomas, *Elements of information theory* (Wiley-Interscience, New York, NY, USA, 1991).

[46] E. Cristofaro and G. Tsudik, "Practical private set intersection protocols with linear complexity", in "FC'10", vol. 6052, pp. 143–159 (Tenerife, Canary Islands, Spain, 2010).

[47] A. Davis, M. Rubinstein, N. Wadhwa, G. Mysore, F. Durand and W. Freeman, "The visual microphone: Passive recovery of sound from video", ACM Transactions on Graphics **33**, 4, 79:1–79:10 (2014).

[48] P. Dillinger and P. Manolios, "Bloom filters in probabilistic verification", in "FMCAD'04", (Austin, TX, USA, 2004).

[49] W. Dong, V. Dave, L. Qiu and Y. Zhang, "Secure friend discovery in mobile social networks", in "INFOCOM'11", (Shanghai, China, 2011).

[50] C. Dwork, "Differential privacy", in "ICALP'06", pp. 1–12 (S. Servolo, Venice, Italy, 2006).

[51] C. Dwork, F. McSherry, K. Nissim and A. Smith, "Calibrating noise to sensitivity in private data analysis", in "TCC'06", pp. 265–284 (New York, NY, 2006).

[52] C. Dwork, M. Naor, T. Pitassi and G. Rothblum, "Differential privacy under continual observation", in "STOC'10", (Cambridge, Massachusetts, USA, 2010).

[53] F. Eigner, A. Kate, M. Maffei, F. Pampaloni and I. Pryvalov, "Differentially private data aggregation with optimal utility", in "ACSAC'14", pp. 316–325 (New Orleans, Louisiana, USA, 2014).

[54] X. Fan and G. Gong, "Securing nfc with elliptic curve cryptography – challenges and solutions", in "RFIDSec'13 Asia", (Guangzhou, China, 2013).

[55] D. Fleet and A. Jepson, "Computation of component image velocity from local phase information", International Journal of Computer Vision **5**, 1, 77–104 (1990).

[56] M. Freedman, K. Nissim and B. Pinkas, "Efficient private matching and set intersection", in "EUROCRYPT'04", pp. 1–19 (Interlaken, Switzerland, 2004).

[57] A. Friedman, I. Sharfman, D. Keren and A. Schuster, "Privacy-preserving distributed stream monitoring", in "NDSS'14", (San Diego, CA, 2014).

[58] M. Garofalakis, D. Keren and V. Samoladas, "Sketch-based geometric monitoring of distributed stream queries", Journal Proceeding VLDB Endowment **6**, 10, 937–948 (2013).

[59] T. Gautama and M. V. Hulle, "A phase-based approch to the estimation of the optical flow field using spatial filtering", IEEE Transactions on Neural Networks **13**, 5, 1127–1136 (2002).

[60] O. Goldreich, *The Foundations of Cryptography*, vol. 2 (Cambridge University Press, 2004).

[61] S. Gollakota and D. Katabi, "Physical layer wireless security made fast and channel independent", in "INFOCOM'11", (Shanghai, China, 2011).

[62] S. Guha, K. Plarre, D. Lissner, S. Mitra, B. Krishna, P. Dutta and S. Kumar, "Autowitness: Locating and tracking stolen property while tolerating gps and radio outages", in "SenSys'10", (Zurich, Switzerland, 2010).

[63] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann and I. Witten, "The weka data mining software: An update", SIGKDD Explorations Newsletter **11**, 1, 10–18 (2009).

[64] S. He, J. Chen, Y. Sun, D. Yau and N. K. Yip, "On optimal information capture by energy-constrained mobile sensors", Vehicular Technology, IEEE Transactions on **59**, 5, 2472–2484 (2010).

[65] W. He, X. Liu, H. Nguyen, K. Nahrstedt and T. F. Abdelzaher, "PDA: Privacy-preserving data aggregation in wireless sensor networks", in "IEEE INFOCOM'07", pp. 2045–2053 (Anchorage, Alaska, USA, 2007).

[66] J. Huang, F. Qian, A. Gerber, M. Mao, S. Sen and O. Spatscheck, "A close examination of performance and power characteristics of 4G LTE networks", in "MobiSys'12", (Low Wood Bay, Lake District, UK, 2012).

[67] D. Hush and C. Wood, "Analysis of tree algorithm for RFID arbitration", in "IEEE ISIT'98", pp. 107–107 (Cambridge, MA, 1998).

[68] S. Jana, S. Premnath, M. Clark, S. Kasera, N. Patwari and S. Krishnamurthy, "On the effectiveness of secret key extraction from wireless signal strength in real environments", in "MobiCom'09", (Beijing, China, 2009).

[69] M. Jawurek and F. Kerschbaum, "Fault-tolerant privacy-preserving statistics", in "Privacy Enhancing Technologies", vol. 7384, pp. 221–238 (2012).

[70] M. Joye and B. Libert, "A scalable scheme for privacy-preserving aggregation of time-series data", in "Financial Cryptography and Data Security", vol. 7859, pp. 111–125 (2013).

[71] T. Jung, X. Mao, X.-Y. Li, S.-J. Tang, W. Gong and L. Zhang, "Privacy-preserving data aggregation without secure channel: Multivariate polynomial evaluation", in "ICNP'13", (Turin, Italy, 2013).

[72] R. Keralapura, G. Cormode and J. Ramamirtham, "Communication-efficient distributed monitoring of thresholded counts", in "SIGMOD'06", (Chicago, IL, USA, 2006).

[73] D. Keren, G. Sagy, A. Abboud, D. Ben-David, A. Schuster, I. Sharfman and A. Deligiannakis, "Geometric monitoring of heterogeneous streams", IEEE Trans. Knowl. Data Eng. **26**, 8, 1890–1903 (2014).

[74] A. Kirsch and M. Mitzenmacher, "Less hashing, same performance: Building a better Bloom filter", in "ESA'06", (Zurich, Switzerland, 2006).

[75] L. Kissner and D. Song, "Privacy-preserving set operations", in "CRYPTO'05", pp. 241–257 (Santa Barbara, CA, 2005).

[76] B. Klimt and Y. Yang, "The enron corpus: A new dataset for email classification research", in "Machine learning: ECML 2004", pp. 217–226 (Springer, 2004).

[77] M. Kodialam and T. Nandagopal, "Fast and reliable estimation schemes in RFID systems", in "ACM MOBICOM'06", pp. 322–333 (Los Angeles, CA, 2006).

[78] L. Lai, Y. Liang and W. Du, "Phy-based cooperative key generation in wireless networks", in "Communication, Control, and Computing (Allerton), 2011 49th Annual Allerton Conference on", (Allerton House, UIUC, Illinois, USA, 2011).

[79] L. Lai, Y. Liang and W. Du, "Cooperative key generation in wireless networks", IEEE Journal on Selected Areas in Communications **30**, 8, 1578–1588 (2012).

[80] C. Law, K. Lee and K. Siu, "Efficient memoryless protocol for tag identification", in "ACM DIAL-M'00", pp. 75–84 (Boston, MA, 2000).

[81] A. Lazerson, I. Sharfman, D. Keren, A. Schuster, M. Garofalakis and V. Samoladas, "Monitoring distributed streams using convex decompositions", Journal Proceeding VLDB Endowment **8**, 5, 545–556 (2015).

[82] L. Lei, Z. Zhong, C. Lin and X. Shen, "Operator controlled device-to-device communications in let-advanced networks", IEEE Wireless Communications **19**, 3, 96–104 (2012).

[83] L. Li, X. Zhao and G. Xue, "Unobservable re-authentication for smartphones", in "NDSS'13", (San Diego, USA, 2013).

[84] M. Li, N. Cao, S. Yu and W. Lou, "FindU: Privacy-preserving personal profile matching in mobile social networks", in "INFOCOM'11", (Shanghai, China, 2011).

[85] Q. Li and G. Cao, "Efficient and privacy-preserving data aggregation in mobile sensing", in "ICNP'12", pp. 1–10 (2012).

[86] Q. Li and G. Cao, "Efficient and privacy-preserving data aggregation in mobile sensing", in "ICNP'12", (Austin, TX, USA, 2012).

[87] Q. Li and G. Cao, "Efficient privacy-preserving stream aggregation in mobile sensing with low aggregation error", in "Privacy Enhancing Technologies", vol. 7981, pp. 60–81 (2013).

[88] T. Li, S. Chen and Y. Ling, "Identifying the missing tags in a large RFID system", in "ACM Mobihoc'10", pp. 1–10 (Chicago, IL, 2010).

[89] T. Li, W. Luo, Z. Mo and S. Chen, "Privacy-preserving rfid authentication based on cryptographical encoding", in "INFOCOM'12", (Orlando, USA, 2012).

[90] T. Li, Y. Chen, J. Sun, X. Jin and Y. Zhang, "iLock: Immediate and automatic locking of mobile devices against data theft", in "CCS'16", (Vienna, Austria, 2016).

[91] X. Liang, X. Li, K. Zhang, R. Lu, X. Lin and X. Shen, "Fully anonymous profile matching in mobile social networks", IEEE Journal on Selected Areas in Communications **31**, 9, 641–655 (2013).

[92] X. Liao, S. Uluagac and R. Beyah, "S-match: Verifiable privacy-preserving profile matching for mobile social services", in "DSN'14", pp. 287–298 (2014).

[93] Z. Lin, D. Kune and N. Hopper, "Efficient private proximity testing with GSM location sketches", in "FC'12", (Bonaire, 2012).

[94] H. Liu, S. Saroiu, A. Wolman and H. Raj, "Software abstractions for trusted sensors", in "MobiSys'12", pp. 365–378 (Low Wood Bay, Lake District, UK, 2012).

[95] H. Liu, J. Yang, Y. Wang and Y. Chen, "Collaborative secret key extraction leveraging received signal strength in mobile wireless networks", in "IEEE INFOCOM'12", (Orlando, FL, 2012).

[96] J. Liu, S. Zhang, N. Kato, H. Ujikawa and K. Suzuki, "Device-to-device communications for enhancing quality of experience in software defined multi-tier lte-a networks", IEEE Network **29**, 4, 46–52 (2015).

[97] L. Lu, J. Han, R. Xiao and Y. Liu, "Action: Breaking the privacy barrier for rfid systems", in "INFOCOM'09", (Rio de Janeiro, Brazil, 2009).

[98] R. Lu, X. Lin, X. Liang and X. Shen, "A secure handshake scheme with symptoms-matching for mhealthcare social network", Mobile Networks and Applications pp. 1–12 (2010).

[99] S. Lu, J. Yuan, S. Yu and M. Li, "ASK-BAN: authenticated secret key extraction utilizing channel characteristics for body area networks", in "ACM WiSec'13", (Budapest, Hungary, 2013).

[100] W. Luo, S. Chen, T. Li and S. Chen, "Efficient missing tag detection in RFID systems", in "INFOCOM'11", (Shanghai, China, 2011).

[101] W. Luo, S. Chen, T. Li and Y. Qiao, "Probabilistic missing-tag detection and energy-time tradeoff in large-scale rfid systems", in "MobiHoc'12", (Hilton Head, USA, 2012).

[102] W. Luo, S. Chen, Y. Qiao and T. Li, "Missing-tag detection and energy-time tradeoff in large-scale RFID systems with unreliable channels", IEEE/ACM Trans. Netw. **22**, 4, 1079–1091 (2014).

[103] F. Maggi, A. Volpatto, S. Gasparini, G. Boracchi and S. Zanero, "A fast eavesdropping attack against touchscreens", in "IAS'11", (Melaka, Malaysia, 2011).

[104] J. Manweiler, R. Scudellari, Z. Cancio and L. Cox, "We saw each other on the subway: secure, anonymous proximity-based missed connections", in "HotMobile'09", (Santa Cruz, California, 2009).

[105] J. Manweiler, R. Scudellari and L. Cox, "SMILE: encounter-based trust for mobile social services", in "CCS'09", pp. 246–255 (Chicago, Illinois, USA, 2009).

[106] P. Marquardt, A. Verma, H. Carter and P. Traynor, "(sp)iphone: Decoding vibrations from nearby keyboards using mobile phone accelerometers", in "CCS'11", (Chicago, Illinois, USA, 2011).

[107] S. Mathur, W. Trappe, N. Mandayam, C. Ye and A. Reznik, "Radio-telepathy: extracting a secret key from an unauthenticated wireless channel", in "MobiCom'08", pp. 128–139 (San Francisco, California, USA, 2008).

[108] F. McSherry and K. Talwar, "Mechanism design via differential privacy", in "FOCS'07", pp. 94–103 (Washington, DC, USA, 2007).

[109] E. Miluzzo, A. Varshavsky, S. Balakrishnan and R. Choudhury, "Tapprints: your finger taps have fingerprints", in "MobiSys'12", pp. 323–336 (Low Wood Bay, Lake District, UK, 2012).

[110] R. Myerson, *Game theory: analysis of conflict* (Harward University Press, 1997).

[111] J. Myung and W. Lee, "Adaptive splitting protocols for RFID tag collision arbitration", in "ACM Mobihoc'06", pp. 202–213 (Florence, Italy, 2006).

[112] M. Nagy, E. D. Cristofaro, A. Dmitrienko, N. Asokan and A.-R. Sadeghi, "Do i know you?: Efficient and privacy-preserving common friend-finder protocols and applications", in "The 29th Annual Computer Security Applications Conference", pp. 159–168 (2013).

[113] S. Narain, A. Sanatinia and G. Noubir, "Single-stroke language-agnostic keylogging using stereo-microphones and domain specific machine learning", in "WiSec'14", (Oxford, United Kingdom, 2014).

[114] A. Narayanan, N. Thiagarajan, M. Lakhani, M. Hamburg and D. Boneh, "Location privacy via private proximity testing", in "NDSS'11", (San Diego, CA, 2011).

[115] A. Nemmaluri, M. Corner and P. Shenoy, "Sherlock: automatically locating objects for humans", in "MobiSys'08", pp. 187–198 (Breckenridge, CL, 2008).

[116] H. Nishiyama, M. Ito and N. Kato, "Relay-by-smartphone: realizing multihop device-to-device communications", IEEE Communications Magazine **52**, 4, 56–65 (2014).

[117] B. Niu, X. Zhu, T. Zhang, H. Chi and H. Li, "P-Match: Priority-aware friend discovery for proximity-based mobile social networks", in "IEEE MASS'13", pp. 351–355 (2013).

[118] C. Olston, J. Jiang and J. Widom, "Adaptive filters for continuous queries over distributed data streams", in "SIGMOD'03", (San Diego, CA, USA, 2003).

[119] E. Owusu, J. Han, S. Das, A. Perrig and J. Zhang, "Accessory: Password inference using accelerometers on smartphones", in "HotMobile'12", (San Diego, CA, USA, 2012).

[120] D. Ping, X. Sun and B. Mao, "Textlogger: Inferring longer inputs on touch screen using motion sensors", in "WiSec'15", (New York, USA, 2015).

[121] J. Portilla and E. Simoncelli, "A parametric texture model based on joint statistics of complex wavelet coefficients", International Journal of Computer Vision **40**, 1, 49–70 (2000).

[122] L. Pournajaf, L. Xiong, V. Sunderam and S. Goryczka, "Spatial task assignment for crowd sensing with cloaked locations", in "MDM'14", (Brisbane, Australia, 2014).

[123] R. Raguram, A. White, D. Goswami, F. Monrose and J.-M. Frahm, "ispy: Automatic reconstruction of typed input from compromising reflections", in "CCS'11", (Chicago, IL, USA, 2011).

[124] V. Rastogi and S. Nath, "Differentially private aggregation of distributed time-series with transformation and encryption", in "ACM SIGMOD", pp. 735–746 (Indianapolis, Indiana, 2010).

[125] A. Sayeed and A. Perrig, "Secure wireless communications: Secret keys through multipath", in "ICASSP'08", (2008).

[126] V. Shah and V. Wong, "Cardinality estimation in RFID systems with multiple readers", Wireless Communications, IEEE Transactions on **10**, 5, 1458–1469 (2011).

[127] M. Shahzad, A. Liu and A. Samuel, "Secure unlocking of mobile touch screen devices by simple gestures: You can see it but you can not do it", in "MobiCom'13", pp. 39–50 (Miami, USA, 2013).

[128] I. Sharfman, A. Schuster and D. Keren, "A geometric approach to monitoring threshold functions over distributed data streams", in "SIGMOD'06", (Chicago, IL, USA, 2006).

[129] E. Shi, T.-H. H. Chan, E. G. Rieffel, R. Chow and D. Song, "Privacy-preserving aggregation of time-series data stream", in "NDSS'11", (San Diego, CA, 2011).

[130] J. Shi, R. Zhang, Y. Liu and Y. Zhang, "PriSense: privacy-preserving data aggregation in people-centric urban sensing systems", in "INFOCOM'10", (San Diego, CA, 2010).

[131] D.-H. Shin, S. He and J. Zhang, "Joint sensing task and subband allocation for large-scale spectrum profiling", in "INFOCOM'15", (Hongkong, 2015).

[132] D. Shukla, R. Kumar, A. Serwadda and V. Phoha, "Beware, your hands reveal your secrets!", in "CCS'14", (Scottsdale, Arizona, USA, 2014).

[133] L. Simon and R. Anderson, "Pin skimmer: Inferring pins through the camera and microphone", in "SPSM'13", (Berlin, Germany, 2013).

[134] E. Simoncelli and W. Freeman, "The steerable pyramid: A flexible architecture for multi-scale derivative computation", in "ICIP'95", (Washington, DC, 1995).

[135] J. Sun, X. Chen, J. Zhang, Y. Zhang and J. Zhang, "SYNERGY: A game-theoretical approach for cooperative key generation in wireless networks", in "IEEE INFOCOM'14", (Toronto, Canada, 2014).

[136] J. Sun, X. Chen, Y. Zhang and J. Zhang, "SYNERGY: A game-theoretical approach for cooperative key generation in wireless networks", Technical report, Arizona State University, `http://cnsg.asu.edu/files/sun-INFOCOM14.pdf` (2013).

[137] J. Sun, R. Zhang, X. Jin and Y. Zhang, "Securefind: Secure and privacy-preserving object finding via mobile crowdsourcing", Wireless Communications, IEEE Transactions on **PP**, 99, 1–1 (2015).

[138] J. Sun, R. Zhang, J. Zhang and Y. Zhang, "TouchIn: Sightless two-factor authentication on multi-touch mobile devices", in "IEEE CNS'14", (San Francisco, CA, 2014).

[139] J. Sun, R. Zhang and Y. Zhang, "Privacy-preserving spatiotemporal matching", in "IEEE INFOCOM'13", (Turin, Italy, 2013).

[140] J. Sun, X. Jin, Y. Chen, J. Zhang, R. Zhang and Y. Zhang, "VISIBLE: Video-assisted keystroke inference from tablet backside motion", in "NDSS'16", (San Diego, USA, 2016).

[141] J. Sun, R. Zhang, J. Zhang and Y. Zhang, "PriStream: Privacy-preserving distributed stream monitoring of thresholded percentile statistics", in "IEEE INFOCOM'16", (San Francisco, USA, 2016).

[142] J. Sun, R. Zhang and Y. Zhang, "Privacy-preserving spatiotemporal matching for secure device-to-device communications", IEEE Internet of Things Journal **3**, 6, 1048–1060 (2016).

[143] C. Tan, B. Sheng and Q. Li, "How to monitor for missing RFID tags", in "ICDCS'08", pp. 295–302 (Beijing, China, 2008).

[144] C. Tan, B. Sheng and Q. Li, "Efficient techniques for monitoring missing RFID tags", IEEE Transactions on Wireless Communication (2010).

[145] H. Tan, S. Jha, D. Ostry, J. Zic and V. Sivaraman, "Secure multi-hop network programming with multiple one-way key chains", in "WiSec'08", pp. 183–193 (2008).

[146] A. Thapa, M. Li, S. Salinas and P. Li, "Asymmetric social proximity based private matching protocols for online social networks", IEEE Transactions on Parallel and Distributed Systems **26**, 6, 1547–1559 (2015).

[147] H. To, G. Ghinita and C. Shahabi, "A framework for protecting worker location privacy in spatial crowdsourcing", in "VLDB'14", (Hangzhou, China, 2014).

[148] A. Torralba and A. Oliva, "Depth estimation from image structure", (2002).

[149] N. Wadhwa, M. Rubinstein, F. Durand and W. T. Freeman, "Phase-based video motion processing", ACM Transactions on Graphics **32**, 4, 80:1–80:10 (2013).

[150] R. Wilson, D. Tse and R. Scholtz, "Channel identification: Secret sharing using reciprocity in ultrawideband channels", IEEE Trans. on Information Forensics and Security **2**, 3, 364–375 (2007).

[151] J. Won, C. Ma, D. Yau and N. Rao, "Proactive fault-tolerant aggregation protocol for privacy-assured smart metering", in "INFOCOM'14", pp. 2804–2812 (2014).

[152] Y. Xu, J. Heinly, A. White, F. Monrose and J.-M. Frahm, "Seeing double: Reconstructing obscured typed input from repeated compromising reflections", in "CCS'13", (Berlin, Germany, 2013).

[153] Z. Xu, K. Bai and S. Zhu, "Taplogger: Inferring user inputs on smartphone touchscreens using on-board motion sensors", in "WiSec'12", (Tucson, AZ, USA, 2012).

[154] D. Yang, G. Xue, X. Fang and J. Tang, "Crowdsourcing to smartphones: Incentive mechanism design for mobile phone sensing", in "MobiCom'12", (Istanbul, Turkey, 2012).

[155] L. Yang, J. Han, Y. Qi and Y. Liu, "Identification-free batch authentication for RFID tags", in "ICNP'10", (Kyoto, Japan, 2010).

[156] Z. Yang, B. Zhang, J. Dai, A. Champion, D. Xuan and D. Li, "E-SmallTalker: A distributed mobile system for social networking in physical proximity", in "ICDCS'10", pp. 468–477 (Genoa, Italy, 2010).

[157] Q. Yao, Y. Qi, J. Han, J. Zhao, X. Li and Y. Liu, "Randomizing rfid private authentication", in "PerCom'09", (Galveston, TX, 2009).

[158] Q. Ye, H. Wang and J. Pieprzyk, "Distributed private matching and set operations", in "ISPEC'08", vol. 4991, pp. 347–360 (Sydney, Australia, 2008).

[159] Q. Yue, Z. Ling, X. Fu, B. Liu, K. Ren and W. Zhao, "Blind recognition of touched keys on mobile devices", in "CCS'14", (Scottsdale, Arizona, USA, 2014).

[160] K. Zeng, D. Wu, A. Chan and P. Mohapatra, "Exploiting multiple-antenna diversity for shared secret key generation in wireless networks", in "Proceedings of the 29th conference on Information communications", INFOCOM'10 (San Diego, California, USA, 2010).

[161] L. Zhang, X. Li and Y. Liu, "Message in a sealed bottle: Privacy preserving friending in social networks", in "The 33rd International Conference on Distributed Computing Systems", (2013).

[162] L. Zhang, J. Zhang and X. Tang, "Assigned tree slotted aloha RFID tag anti-collision protocols", Wireless Communications, IEEE Transactions on **12**, 11, 5493–5505 (2013).

[163] R. Zhang, Y. Liu, Y. Zhang and J. Sun, "Fast identification of the missing tags in a large rfid system", in "IEEE SECON'11", (Salt Lake City, Utah, 2011).

[164] R. Zhang, J. Shi, Y. Zhang and C. Zhang, "Privacy-preserving profile matching for proximity-based mobile social networking", IEEE Journal on Selected Areas in Communications Special Issue on Emerging Technologies **31**, 9, 268–278 (2013).

[165] R. Zhang, J. Sun, Y. Zhang and X. Huang, "Jamming-resilient secure neighbor discovery in mobile ad hoc networks", Wireless Communications, IEEE Transactions on **PP**, 99, 1–1 (2015).

[166] R. Zhang, J. Sun, Y. Zhang and C. Zhang, "Secure spatial top-k query processing via untrusted location-based service providers", Dependable and Secure Computing, IEEE Transactions on **12**, 1, 111–124 (2015).

[167] R. Zhang, Y. Zhang, J. Sun and G. Yan, "Fine-grained private matching for proximity-based mobile social networking", in "IEEE INFOCOM'12", (Orlando, FL, 2012).

[168] D. Zhao, X.-Y. Li and H. Ma, "How to crowdsource tasks truthfully without sacrificing utility: Online incentive mechanisms with budget constraint", in "INFOCOM'14", (Toronto, Canada, 2014).

[169] Y. Zhao and J. Wu, "B-SUB: A practical bloom-filter-based publish-subscribe system for human networks", in "ICDCS '10", (2010).

[170] Y. Zheng and M. Li, "P-MTI: physical-layer missing tag identification via compressive sensing", in "INFOCOM'13", pp. 917–925 (Turin, Italy, 2013).

[171] F. Zhou, C. Chen, D. Jin, C. Huang and H. Min, "Evaluating and optimizing power consumption of anti-collision protocols for applications in RFID systems", in "ACM ISLPED'04", pp. 357–362 (Newport, CA, 2004).

[172] H. Zhou, L. Huie and L. Lai, "Key generation in two-way relay wireless channels", in "CISS'13", (Baltimore, MD, USA, 2013).

[173] H. Zhu, S. Du, M. Li and Z. Gao, "Fairness-aware and privacy-preserving friend matching protocol in mobile social networks", IEEE Transactions on Emerging Topics in Computing **1**, 1, 192–200 (2013).

[174] T. Zhu, Q. Ma, S. Zhang and Y. Liu, "Context-free attacks using keyboard acoustic emanations", in "CCS'14", (Scottsdale, Arizona, USA, 2014).

[175] L. Zhuang, F. Zhou and J. Tygar, "Keyboard acoustic emanations revisited", in "CCS'05", (Alexandria, VA, USA, 2005).