OPPORTUNISTIC COMPUTING IN WIRELESS NETWORKS

DISSERTATION

Presented in Partial Fulfillment of the Requirements for the Degree Doctor of Philosophy in the Graduate School of The Ohio State University

By

Zhimin Yang, B.E., M.E., M.S.

Graduate Program in Computer Science and Engineering

The Ohio State University

2010

Dissertation Committee:

Dong Xuan, Adviser

Ten-Hwang Lai

Feng Qin
Copyright by
Zhimin Yang
2010
Opportunistic computing is the exploitation of opportunities as they arise to provide computing services that meet application requirements. It maximally utilizes available resources to improve such services subject to application constraints. When applied to different applications, we have different concrete instances of opportunistic computing. In this dissertation, we focus on its utilization in the context of wireless networking applications. Particularly, we study four instances of opportunistic computing in wireless networks: opportunistic social networking, opportunistic localization, opportunistic encryption, and opportunistic authentication.

First, we study opportunistic social networking with mobile phones. We develop E-SmallTalker, a novel distributed mobile communication system that aims to facilitate more effective social networking among strangers in physical proximity. We propose a privacy-preserving opportunistic commonality discovery protocol that utilizes iterative Bloom filters to encode user information. We eliminate unnecessary user interactions by exploiting Bluetooth service attributes to publish encoded user data.

Second, we study opportunistic localization in wireless sensor networks. We propose an “anti-sensor network” system to localize an adversary’s sensors in a non-cooperative environment where these sensors try to evade or deter localization. The system relies on a set of monitors opportunistically observing intermittent wireless
signals emitted by sensors in the protected area. With these observations, we estimate sensor locations with an opportunistic localization algorithm. We also design improved algorithms to handle additional countermeasures that sensors can employ such as message encryption and non-uniform transmission power levels.

Third, we study opportunistic encryption in wireless LANs. We identify and analyze security issues in the current 802.11i security standard. We illustrate the severe consequences resulting from the lack of frame authentication with an exemplary denial-of-service attack. We propose a new key-establishment algorithm for opportunistic encryption in the link layer. Our algorithm makes minimal changes to the existing 802.11 protocol.

Fourth, we study opportunistic authentication in wireless web applications. We propose opportunistically utilizing users’ trusted devices such as mobile phones to aid web authentication in wireless networks. We design and implement a prototype one-time-password (OTP) authentication system that works seamlessly in heterogeneous environments. We propose a one-time-password reference service that allows an OTP-token to be opportunistically used in multiple web applications. We also propose a new connection-aware one-time-password algorithm to thwart man-in-the-middle attacks by using connection information.
Dedicated To

My Family

For Love and Support
ACKNOWLEDGMENTS

First of all, I would like to thank my advisor, Dr. Xuan, for his superb guidance and inspirations throughout this academic experience. I really enjoyed all the insightful conversations which made this dissertation possible. Dr. Xuan showed me how to rigorously define and refine problems, develop solutions, and organize research papers. He’s knowledgeable, patient and always has insightful suggestions and comments. Dr. Xuan broadened my scope and his diligence and passion for high quality research will always encourage me to achieve more. Both my wife and I thank Dr. Xuan from the bottom of our hearts for the kind support and help when our family was in the difficult and unexpected situations. All the wonderful Thanksgiving, Christmas, and Chinese New Year parties at Dr. Xuan’s home are joyful memories which can never be forgettable. I can’t say enough to thank him.

I am also extremely grateful to Dr. Ten H. Lai, Dr. Feng Qin, Dr. Eylem Ekici, and Dr. Timothy Long for serving in my candidacy exam and/or dissertation committees and their constructive feedback in this research.

I would also like to acknowledge my labmates, Dr. Xun Wang, Dr. Sriram Chellappan, Dr. Wenjun Gu, Dr. Xiaole Bai, Adam C. Champion, Boxuan Gu, Jiangpeng Dai, Xudong Ni and Jin Teng for the stimulating discussions. You make my life in OSU so enjoyable. I wish you the best in your career.
I would like to express my gratitude to all those who have advised, helped, and supported me during the past years.

Finally, I am forever indebted to my family for their unconditional love and support throughout my life. My lovely son, David, is the every reason for me to go ahead.
VITA

April 14, 1972 ............................... Born – Harbin, China

1995 ................................. Bachelor in Engineering
Harbin Institute of Technology
Harbin, China

1997 ................................. Master in Engineering
Harbin Institute of Technology
Harbin, China

2009 ................................. Master of Science
The Ohio State University

2005 – Present ........................ Graduate Teaching Associate / Graduate Research Associate
The Ohio State University

PUBLICATIONS

Research Publications


FIELDS OF STUDY

Major Field: Computer Science and Engineering

Studies in:

Computer Networking
- Prof. Anish Arora
- Prof. Eylem Ekici
- Prof. Ten-H. Lai
- Prof. David Lee
- Prof. Ming T. Liu
- Prof. Prasun Sinha
- Prof. Dong Xuan

Software Systems
- Prof. Paul Sivilotti
- Prof. Igor Malkiman

Parallel and Distributed Systems
- Prof. P. Sadayappan
- Prof. Gagan Agrawal
- Prof. Feng Qin
# TABLE OF CONTENTS

<table>
<thead>
<tr>
<th>Chapters:</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>1. INTRODUCTION</strong></td>
<td>1</td>
</tr>
<tr>
<td>1.1 Motivation</td>
<td>1</td>
</tr>
<tr>
<td>1.2 Contributions of This Dissertation</td>
<td>6</td>
</tr>
<tr>
<td>1.3 Organization of This Dissertation</td>
<td>9</td>
</tr>
<tr>
<td><strong>2. OPPORTUNISTIC SOCIAL NETWORKING WITH MOBILE PHONES</strong></td>
<td>11</td>
</tr>
<tr>
<td>2.1 Motivation</td>
<td>12</td>
</tr>
<tr>
<td>2.2 Related Work</td>
<td>17</td>
</tr>
<tr>
<td>2.3 Opportunistic Social Networking: Privacy Preserving Commonality</td>
<td>22</td>
</tr>
<tr>
<td>2.3.1 Commonality Discovery Problem</td>
<td>22</td>
</tr>
<tr>
<td>2.3.2 Opportunistic Commonality Discovery Protocol</td>
<td>25</td>
</tr>
<tr>
<td>2.3.3 Opportunistic Commonality Verification Protocol</td>
<td>29</td>
</tr>
<tr>
<td>2.4 E-SmallTalker System Design, Implementation and Evaluation</td>
<td>32</td>
</tr>
<tr>
<td>2.4.1 System Architecture</td>
<td>32</td>
</tr>
<tr>
<td>2.4.2 System Implementation</td>
<td>39</td>
</tr>
</tbody>
</table>

Abstract ................................................................. ii

Dedication ............................................................... iv

Acknowledgments ......................................................... v

Vita .......................................................... vii

List of Figures ......................................................... xiii
## LIST OF FIGURES

<table>
<thead>
<tr>
<th>Figure</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.1 Pre-computed Bloom filter size (in bytes) with regard to data set size $(N)$ and desired false positive rate $(F)$</td>
<td>29</td>
</tr>
<tr>
<td>2.2 An Example of Alice’s Extended vCard from Bob</td>
<td>30</td>
</tr>
<tr>
<td>2.3 E-SmallTalker System Architecture</td>
<td>32</td>
</tr>
<tr>
<td>2.4 Information Exchange without Bluetooth Connection</td>
<td>38</td>
</tr>
<tr>
<td>2.5 Discovery Time with 60s Bluetooth Searching Interval</td>
<td>42</td>
</tr>
<tr>
<td>2.6 Discovery Time with 120s Bluetooth Searching Interval</td>
<td>42</td>
</tr>
<tr>
<td>2.7 Successful Discovery Rates under 60s/120s Bluetooth Search Intervals</td>
<td>43</td>
</tr>
<tr>
<td>2.8 Discovery Time vs. Distance</td>
<td>44</td>
</tr>
<tr>
<td>2.9 Successful Discovery Rate vs. Distance</td>
<td>44</td>
</tr>
<tr>
<td>2.10 Energy Cost in Device Discovery and Service Discovery</td>
<td>46</td>
</tr>
<tr>
<td>2.11 Energy Cost in Device Discovery with Different Bluetooth Search Intervals</td>
<td>47</td>
</tr>
<tr>
<td>3.1 An Example Anti-Sensor Network</td>
<td>56</td>
</tr>
<tr>
<td>3.2 Batch Localization Algorithm</td>
<td>66</td>
</tr>
<tr>
<td>3.3 NLS Location Estimation Algorithm</td>
<td>67</td>
</tr>
</tbody>
</table>
5.2 Opportunistic OTP Web Authentication Work Flow. . . . . . . . . . . 141
1.1 Motivation

With wireless networking becoming commonplace in offices, homes and businesses, wireless network security and user privacy have received a great amount of attention in recent years. Clearly, they are very important. Unlike traditional wired networks where a device is almost always accessible by other devices, connectivity between pairs of devices can be intermittent in wireless networks, which results in opportunistic contact. In wireless networks, many factors affect such connectivity. Sometimes there is a signal, sometimes not. Sometimes the signal is strong, sometimes it is weak. The problem of intermittent connectivity is further exacerbated if the devices are mobile and there is no prior knowledge of their location, time of contact, and communication channel. Opportunistic contact between pairs of devices is critical for computation in wireless networks.

Besides intermittency, heterogeneity is another source of opportunity. Potentially, there are many kinds of devices that are likely to come in contact with each other opportunistically, such as mobile phones, handheld and notebook computers, wireless sensors, cameras, and RFID-enabled objects [1]. Clearly, these devices have diverse
communicational and computational capabilities. Improving computation results by taking advantage of these diverse capabilities is an important research issue.

In order to utilize the full potential of opportunistic contacts, new networking and computing paradigms are necessary. In the last few years, there have been significant research efforts toward specific instances of opportunistic computing such as opportunistic data forwarding [2] and opportunistic encryption [3–6].

While opportunistic data forwarding [2] exploits opportunistic contacts to primarily develop message or content forwarding applications, opportunistic encryption has different meanings in the literature. It can refer to any system that, when connecting to another system, attempts to encrypt the communications channel; otherwise, it falls back to unencrypted communications [5]. It is the act of setting up a secure channel without verifying the identity of the other host [4]. The objective of opportunistic encryption is to allow encryption without any pre-arrangement specific to the pair of systems involved [3]. Opportunistic encryption can also refer to the framework that uses channel opportunities (acceptable signal to noise ratio) to maximize the throughput subject to desired security constraints [6].

Conti and Kumar [1] are among the first to use the term “opportunistic computing” in recent years. According to them, opportunistic computing initiatives are still in their infancy. There is no clear definition of the term in their paper or in the literature. Here we give our definition by summarizing previous work.

Opportunistic computing is the exploitation of opportunities as they arise to provide computing services that meet application requirements [1]. It maximally utilizes available network resources to improve such services subject to application constraints [7].
In this dissertation, we focus on using opportunistic computing to improve computation in wireless network applications. While applying it to improve wireless network privacy and security is a burgeoning and promising research topic, it does have limitations. Opportunistic computing might fail if no opportunities arise or if the opportunities are not fully exploited. For example, opportunistic encryption can be used to combat passive wiretapping. It does not provide a strong level of security as authentication may be difficult to establish and secure communications may not be mandatory. Therefore, some researchers call opportunistic encryption “Better Than Nothing Security.”

We wish to explore the salient features of opportunistic computing. It can be adapted to applications to satisfy particular domain constraints. It is simple, efficient, lightweight, and easy to implement, which removes a significant impediment to the mass adoption of wireless security. In general, we do not expect opportunistic computing to provide very strong security, although it may do so in some applications. We do expect that it will increase security when exploiting opportunities. We think it of as a “Better Than Not Taking The Opportunities Security.”

In this dissertation, we study four instances of opportunistic computing in wireless networks: (1) opportunistic social networking; (2) opportunistic localization; (3) opportunistic encryption, and (4) opportunistic authentication.

(1). *Opportunistic Social Networking with Mobile Phones*: In social networks, people may encounter each other opportunistically in physical proximity. They may start a face-to-face conversation and gain familiarity with each other. Thus, they expand their social networks by exploiting opportunistic interactions. These face-to-face interactions in physical proximity are crucial to people’s social networking
and they are irreplaceable. Physical proximity provides immediate feedback via non-verbal communication that is essential to forging strong relationships. However, due to the *social gap* between people, they do not fully utilize opportunities for face-to-face interaction in physical proximity. Small talk is an important social lubricant that helps people, especially strangers, initiate conversations and make friends with each other in physical proximity. However, due to difficulties in quickly identifying significant topics of common interest, real-world small talk tends to be superficial. In this dissertation, we wish to transform mobile phones into a tool that facilitates social networking in physical proximity. Toward this end, we propose *E-SmallTalker*, a mobile phone-based purely distributed communication system that encourages people in physical proximity to communicate with each other. The core of E-SmallTalker is a set of privacy-preserving opportunistic commonality discovery protocols customized for Bluetooth communications. E-SmallTalker automatically generates short, user-controlled “small talk” messages on users’ phones based on their common interests, friends, and experiences.

(2). *Opportunistic Localization in Wireless Sensor Networks*: Wireless sensor networks have been largely used in security-related applications where sensor nodes detect and report events that may pose a threat to the network owner. To protect an important area from surveillance by an adversary’s sensor nodes, one possible solution is to localize and disable the adversary’s sensors. Following this direction, we propose an “anti-sensor network” system to achieve the goal. The core of the anti-sensor network is localization of an adversary’s sensor nodes. Prior research on sensor network localization focused on cooperative localization, where the localization system and the sensors to be localized belong to the same party. On the contrary, in our
system, the localization problem has a non-cooperative, even hostile nature. Sensor nodes may try to evade or deter localization by communicating less information, hiding identities via encryption, and varying transmission power. Therefore, the anti-sensor network system must fully exploit observable signals emitted by sensors during their intermittent wireless communications. We design a set of opportunistic localization algorithms that achieve this.

(3). Opportunistic Encryption in Wireless Local Networks: In wireless networks, situations arise in which everyone is granted access to the network yet data confidentiality is still required. Two salient examples are open-access wireless local area networks (WLANs) in Internet cafés or airports and Bluetooth information sharing systems in which pre-shared secrets do not exist and cannot be setup easily. In these wireless networks, a wireless frame encryption key needs to be established between two strangers to provide data confidentiality. This is a “chicken and egg” problem. To setup an encryption key, a user must first be authenticated. To authenticate a user, an existing shared secret is required. Due to the limitations of wireless techniques, existing key establishment protocols in the literature that are based on public-key cryptography cannot be directly utilized.

In the dissertation, we study opportunistic encryption in open wireless LANs. In opportunistic encryption, the key issue is setting up a wireless frame encryption key between communication parties given the constraints of wireless technology. We propose an opportunistic encryption key establishment algorithm to replace the open system authentication algorithm defined in 802.11. Since it takes the place of an “authentication” algorithm, but really does not perform any authentication functions, we call it a “dummy authentication” algorithm. No user authentication is performed
at all. One application of dummy authentication is a situation like that of an open-access WLAN when everyone is granted access to the wireless network yet the data confidentiality must still be achieved.

(4). *Opportunistic Authentication in Wireless Web Applications*: Due to the open nature of wireless networks and the problem of rogue access points (APs), wireless users have greater risk of being attacked by a man in the middle (MITM) than in wired networks. We propose opportunistically utilizing users’ trusted devices such as mobile phones to aid web authentication in wireless networks and prevent MITM attacks. We design an opportunistic connection-aware one-time-password (OTP) web authentication system that enables seamless web authentication in heterogeneous environments. Our system includes soft OTP tokens on mobile phones, regardless of their Bluetooth capabilities, as opposed to traditional hardware OTP tokens.

1.2 Contributions of This Dissertation

This dissertation aims to improve privacy and security in wireless network applications by utilizing opportunistic computing. We make the following contributions:

(1). *Opportunistic Social Networking with Mobile Phones*: We have developed *E-SmallTalker*, a novel distributed mobile phone communication system that aims to facilitate more effective social networking among strangers in physical proximity. E-SmallTalker recommends short, meaningful topics based on users’ commonalities, such as mutual friends and shared interests.

Our system makes no assumptions about data services such as Internet access. It exchanges user information between two phones and performs matching locally. It uses Bluetooth for communication and the Bluetooth Service Discovery
Protocol (SDP) to search for nearby E-SmallTalker users. The SDP service attributes are extended for a new purpose: i.e. to publish encoded user data for discovering potential small talk topics. Our approach does not require user interference to establish a Bluetooth connection.

We propose a privacy-preserving opportunistic commonality discovery protocol that utilizes iterative Bloom filters to encode user information. Data is encoded in a bit string to address the size limitation of Bluetooth SDP attributes. The initial Bloom filters are refined in a few rounds until the desired low false positive rate is achieved.

We implemented the system using Java ME, which is supported on a wide range of mobile devices including smartphones as well as low-end phones. In addition, we build on the Bluetooth protocol but we do not modify the protocol stack. As a result, the system easily runs on most commercial-off-the-shelf (COTS) mobile devices that are shipped with Bluetooth and Java.

(2). *Opportunistic Localization in Wireless Sensor Networks*: We propose an opportunistic localization system to localize sensor nodes aiming to protect an important area from surveillance by an adversary’s sensor nodes. The non-cooperative, even hostile nature distinguishes our work from existing cooperative localization in sensor networks. Since one important application of our system is to disable the adversary’s sensor network, we call our system an “anti-sensor network” system. The system relies on a set of monitor nodes opportunistically observing intermittent wireless transmissions in the protected area.
Monitors use antenna arrays to measure the direction of arrival and the received signal strength of the signals emitted by sensors. With these opportunistic observations, sensor locations are estimated through collective processing. We designed a set of opportunistic localization algorithms to handle additional counter-measures that can be employed by sensors, including message encryption and non-uniform transmission power levels. The proposed opportunistic localization algorithms are designed based on analytical performance measures. Extensive simulations are conducted to demonstrate the effectiveness of our algorithms.

(3). *Opportunistic Encryption in Wireless Local Networks*: We identify and analyze security issues in the current 802.11i security standard. In general, we find that wireless link frames are unauthenticated and unencrypted. This includes management frames and null and QoS null *data* frames in all types of networks as well as data frames that transfer 802.1x/EAP authentication messages in WPA-Enterprise networks. We identify the potential security vulnerabilities in the current applications of null data frames in 802.11 WLANs. We illustrate the severe consequences resulting from the lack of frame authentication by an exemplary denial of service attack.

We propose a solution to patch the current 802.11i standard and address these issues with opportunistic encryption. We propose a new opportunistic encryption key-establishment algorithm that makes minimal changes to the existing 802.11 protocol. In the algorithm, we apply the public-key cryptography-based key-establishment technique to the 802.11 MAC protocol. Our solution has the
advantages of simplicity, compatibility with the standard with few modifications, and low overhead in computation, memory usage, and channel utilization. A device can support both the old open-system authentication and our new “dummy authentication” opportunistic encryption, which protects existing investments in 802.11 networks. Our solution can provide link-layer data confidentiality in open-access wireless networks, separate session encryption keys for different users, and authentication and encryption of important frames such as management and null data frames.

(4). *Opportunistic Authentication in Wireless Web Applications*: We propose to opportunistically utilize users’ trusted devices such as mobile phones to aid web authentication in wireless networks. We design and implement a prototype one-time-password (OTP) authentication system that works seamlessly in heterogeneous environments. Our new system is user-oriented and based on the concept of a one-time-password reference service, which allows an OTP-token to be used in multiple web authentications. We also propose a new connection-aware one-time-password algorithm, which allows us to thwart man-in-the-middle (MITM) attacks by using more factors than a traditional token to calculate an OTP. Particularly, we use the connection and session information as additional factors. We demonstrated the practicality of our system with a prototype implementation.

1.3 Organization of This Dissertation

The rest of the dissertation is organized as follows. In Chapter 2, we present our work on opportunistic social networking with mobile phones. In Chapter 3, we present our work on opportunistic localization in wireless sensor networks. In Chapter 4, we
present our work on opportunistic encryption in wireless local networks. In Chapter 5, we present our work on opportunistic authentication in wireless web applications. We conclude this dissertation in Chapter 6.
CHAPTER 2

OPPORTUNISTIC SOCIAL NETWORKING WITH MOBILE PHONES

In social networks, people come into contact with each other opportunistically. They may start a face-to-face conversation and take the chance to know each other. Small talk is an important social lubricant that helps people, especially strangers, initiate conversations and make friends with each other in physical proximity. However, due to difficulties in quickly identifying significant topics of common interest, real-world small talk tends to be superficial. The mass popularity of mobile phones can help improve the effectiveness of small talk. In this chapter, we first present a privacy preserving opportunistic commonality discovery protocol that automatically finds out common friends or interests. Then we present E-SmallTalker, a distributed mobile communications system that facilitates social networking in physical proximity using the proposed protocol. The system automatically discovers and suggests topics such as common interests for more significant conversations. We build on Bluetooth Service Discovery Protocol (SDP) to exchange potential topics by customizing service attributes to publish non-service-related information without establishing a connection. We propose a novel iterative Bloom filter that encodes topics to fit in SDP attributes and achieves a low false positive rate.
2.1 Motivation

Face-to-face interaction plays an irreplaceable role in our daily lives, especially for social networking purposes. Compared to other forms of social interaction that are separated by time and space boundaries, face-to-face interaction in physical proximity facilitates non-verbal communication. In a face-to-face meeting, for example, people can easily make eye contact and discern others’ moods, personalities, and surroundings. These non-verbal cues provide immediate and valuable feedback that helps people adjust their topics of conversation, body language, and communication manners accordingly.

Apparently, not all people are equally skillful in harnessing what physical proximity can offer to its fullest extent. A well-known barrier is the so-called social gap. When people interact with strangers or unfamiliar parties, they tend to feel self-conscious and reluctant to communicate.

Small talk is a widely-used everyday technique for shortening the social gap by initiating conversations about readily observable topics such as the weather. However, the effectiveness of small talk is limited if it only covers superficial weather-like topics. Indeed, without assistance, it is generally difficult for ordinary people to identify more significant common topics with strangers in face-to-face social settings.

The mass popularity of mobile phones could potentially help improve the practice of small talk. Today, mobile phones alone are used on a daily basis by over three billion people worldwide, i.e., half of the world’s population. These phones always go with their owners and record much information that could be leveraged for more meaningful social interactions, such as mutual friends, common interests, visited places, and even the fact that two users have already met. This chapter aims to use these widely
available phones to enhance the effectiveness of small talk for social networking in physical proximity among strangers. Specifically, we aim to develop a mobile phone-based system for such networking activities.

*The Key Challenge:* The key challenge for such a mobile social networking system is that it must reach a critical mass of users to be useful. That is, the system would provide little value to a user unless a large percentage of other people with whom he would like to interact are also using the system.

A system that aims to enhance small talk’s effectiveness should be able to automatically identify common topics among strangers. One straightforward approach to achieving this goal is to use a central server-based infrastructure. In this approach, a central server stores all users’ information and provides common topics based on matching results. A user’s client application on his mobile phone needs to report his geo-location or send IDs of phones nearby to the server via data services (e.g., Internet or SMS) so that the server can discover commonalities among users. Then the matching results are retrieved by or pushed to the client application.

The centralized approach is problematic for two reasons: (1) Not all mobile phones have data services and in many developing regions they are unavailable. Furthermore, not all users are willing to pay to use a centralized service. (2) Not all users are willing to report and store their sensitive personal information such as geo-location on a central server. Even though one may be willing to report his own information, he may not have others’ permission to report their information, i.e., their phone IDs. Moreover, a central server can be a performance bottleneck and a single point of failure and has the risk of being compromised. Consequently, a centralized system for small talk will unlikely reach a critical mass of users.
Our Contributions: This chapter presents E-SmallTalker, a novel distributed mobile communication system that aims to facilitate more effective social networking among strangers in physical proximity. Our system makes no assumptions about data services such as Internet access. It exchanges user information between two phones and performs matching locally. It uses Bluetooth for communication and is implemented using Java ME. Hence it can be deployed on most commercial off-the-shelf (COTS) mobile phones that are shipped with Bluetooth and Java.

Beyond addressing the key challenge of reaching critical mass of users, this work makes novel intellectual contributions mainly in the following two areas:

- We build on the Bluetooth Service Discovery Protocol (SDP) to search for nearby E-SmallTalker users. We extend the SDP service attribute values for a new purpose, i.e., to publish encoded user data for discovering potential small talk topics. Our approach does not require user interference to establish a Bluetooth connection.

- We propose a novel privacy preserving opportunistic commonality discovery protocol which utilizes iterative Bloom filters to encode user information. By using one way hash functions, data is encoded in a bit string to address the size limit of Bluetooth SDP attributes. The initial Bloom filters are refined in a few rounds until the desired low false positive rate is achieved.

In the following, we explain the rationales behind our approach and contributions:

First, Bluetooth is the most suitable communication technology for our purposes. There are mainly four types of communication technologies available on mobile
phones: cellular network, IrDA, Wi-Fi, and Bluetooth. Communicating data via cellular networks is costly and often unreliable in typical social settings, e.g., inside a building. Infrared Data Association (IrDA) is limited to line-of-sight communication within a very short distance (e.g., 1 meter), which may be considered as intrusive between strangers. Wi-Fi is only available on relatively high-end mobile phones. In contrast, Bluetooth is available on nearly all mobile phones and its communication range is 10 meters on class II devices. Hence we choose Bluetooth as our communication technology.

However, in order to develop a Bluetooth application on mobile phones, we need to overcome several obstacles. For security reasons, a mobile phone will ask for user permission to initiate or accept a Bluetooth connection as well as a passcode for pairing. Hence, an application that relies on Bluetooth connections to transmit data requires explicit user interactions. This requirement not only requires to “babysit” the system but also is too intrusive for strangers. Therefore, we need to find a way for two phones to exchange information without establishing a Bluetooth connection.

We achieve this by using Bluetooth’s Service Discovery Protocol (SDP) to publish/exchange information. In SDP, each service is represented by a service record that is identified by a 128-bit Universally Unique Identifier. All information about a service that is maintained by an SDP server (on a phone) is contained within a single service record, which consists of a list of service attributes. Each service attribute describes a single characteristic of a service (e.g., its name, type, parameters, protocols used) and consists of an attribute ID and the corresponding attribute value. The attribute value is a variable length field, which our system utilizes to publish encoded user information.
However, SDP can only publish limited information, the size of which varies depending on brands and models of mobile phones. For example, in our experiments, one phone can publish up to 10 user defined attributes, each of which has a maximum of 128 bytes of data. In our opportunistic commonality discovery protocol, we use Bloom filters to encode and “compress” user information in order to accommodate it into SDP’s attribute values. To further reduce the size of exchanged information, we propose a novel Bloom filter technique that iteratively refines Bloom filters in several rounds to achieve a desired low false positive rate given SDP’s constraint. The Bloom filters are published via SDP to discover common topics. A device determines common topics by testing if its topics are in another device’s Bloom filter. As a result, the system incurs limited transmission and computation. We also use Bloom filters to provide privacy against eavesdroppers, as Bloom filters use one-way hashing to encode user information. This makes it impossible to reconstruct the information in a filter without performing an exhaustive search of the input space.

We implemented the proposed system using Java ME, which is supported on a wide range of mobile phones including smartphones as well as low-end phones. In addition, we build our system on the Bluetooth protocol but we do not modify the protocol stack. As a result, the system easily runs on most Bluetooth-enabled COTS mobile phones. We performed experiments on real-world mobile phones. The results show that E-SmallTalker achieves our design goals.

A Typical Usage Scenario: To illustrate the usage of our E-SmallTalker system, consider the following scenario. Suppose that two strangers, Alice and Bob, encounter each other at an airport. They are both interested in several movies. As they are strangers, they only make small talk about the weather, which is clearly superficial.
If they run E-SmallTalker on their mobile phones, the system encodes their movie interests into Bloom filters, which are published as service attribute values in SDP’s service record. The system automatically exchanges Bloom filters, performs information matching locally on the phones, and then informs them of their common interest in movies. They can then easily initiate a meaningful conversation about movies. The entire procedure does not need a central server and Internet access and is transparent to both users until each of their phones finds their common interest.

To the best of our knowledge, E-SmallTalker is the first distributed mobile system for social networking between strangers, with two supporting techniques: a new way of utilizing Bluetooth SDP and a novel opportunistic commonality discovery protocol based on our new iterative Bloom filters.

2.2 Related Work

In this section, we first review related work focusing on social networking applications (SNAs) on mobile phones. Then we give a background on Bloom filters and Bluetooth SDP. A considerable body of work has contributed to SNAs. Some SNAs are centralized, whereas other SNAs are distributed. We will discuss them in the following.

2.2.1 Centralized Social Networking Applications

The SNAs in this category are primarily Internet-based. They store user data including social networks on a (conceptually) central server and allow users to find friends and share data via SNA clients running on mobile phones.

Social Serendipity [8] is a typical example of the centralized SNA. It adopts the approach we mentioned before to store and report user information on a central
server. Specifically, Social Serendipity maps Bluetooth MAC addresses to user profiles on other social networking websites. To facilitate face-to-face interactions between nearby strangers, it retrieves their mobile devices’ Bluetooth MAC addresses and uses them to retrieve the strangers’ profiles on the server for similarity matching. Finally, their work uses SMS for device-server communication.

A number of other centralized SNAs aim to enhance awareness and interaction between friends when they are in physical proximity. In general, these SNAs obtain a user’s current geographical location and notify his nearby friends. This gives friends knowledge of each other’s whereabouts, which facilitates opportunistic interactions. Other representative applications in this category include PeopleTones [9], Hummingbird [10], Just-for-Us [11], MobiLuck [12], P3 Systems [13], Micro-Blog [14], and Loopt [15].

In general, centralized SNAs have the following limitations: (1) the server may not always be reachable; (2) communications between the server and devices (via SMS, Wi-Fi, or 2G/3G) may be costly, unreliable and even unavailable; and (3) user privacy may be compromised, e.g., by saving location and other personal data on a third-party server. By comparison, our system uses short-range communication technologies such as Bluetooth to provide reliable service operation without these limitations.

2.2.2 Distributed Social Networking Applications

The SNAs in this category enable mobile devices to directly communicate with each other without requiring a third party. For example, Social Net [16] logs nearby
users’ Bluetooth addresses to infer users’ interaction patterns over time. Nokia Sensor [17] allows users to detect others in the vicinity via Bluetooth; once a connection is established, two devices can exchange information. PeopleNet [18] focuses on multicasting messages or queries to a selected group of devices connected by a mobile ad hoc network based on Bluetooth or Wi-Fi. Nokia Sensor and PeopleNet require establishment of a Bluetooth connection, which requires user intervention and only occurs between trusted parties. These applications focus on providing users service when they are already connected via Bluetooth.

Point&Connect [19] facilitates device pairing between two users in physical proximity by having one user point his device to another user’s device. When multiple devices are nearby, acoustic cues are used by all devices to determine the device towards which the initiating device is pointing. Our work focuses on initiating meaningful small talk by identifying common topics between two strangers. While Point&Connect has a entirely different focus, the two parties involved in Point&Connect are already known to each other and thus they do not need small talk to build a relationship.

2.2.3 **Background - Bloom Filter**

The Bloom filter [20] is a time- and space-efficient probabilistic data structure for testing whether an element is a member of a set. A Bloom filter is a vector of $m$ bits, each of which is initially set to ‘0’. When adding a new element to the Bloom filter, we compute the element over $k$ independent hash functions to generate $k$ hash values as the indices to the vector. The corresponding $k$ entries are set to ‘1’. To insert a set of $n$ elements, this procedure is repeated $n$ times until all the elements are encoded.
in the Bloom filter. During the procedure, if a bit is already set, we leave it as ‘1’. To query an element against a given Bloom filter, the $k$ hashing indices are computed: The element is a member of the set only if all the $k$ corresponding bits are ‘1’ in the vector.

As a probabilistic data structure, Bloom filters are subject to false positives, i.e., they may mistakenly confirm the membership of a given element in lookup. It is possible that two elements are mapped to the same $k$ positions in the bit vector. The false positive rate is defined by the probability that all the corresponding $k$ bits for any given element are ‘1’ in the Bloom filter although it is not really a member of the represented set. Assuming that a hash function selects each position in a Bloom filter with equal probability, then the quantitative measurement of false positive rate $f$ is defined by the following formula:

$$f = (1 - (1 - 1/m)^{kn})^k \approx (1 - e^{-kn/m})^k$$

(2.1)

There is a tradeoff between the computation time, the Bloom filter size, and the false positive rate, which is directly controlled by the three parameters, $m$, $n$ and $k$.

We extend Bloom filters for mobile social networking. Our goal is to discover common friends and/or interests between people with minimum computation and transmission costs. We propose an iterative commonality discovery protocol that refines an initial Bloom filter until it reaches a specified low false positive rate.

Several other extensions to the basic Bloom filter exist in the literature [21–26] that are orthogonal to ours. For example, [21] computes $k$ hash values with only two hash functions; [22] proposes the counting Bloom filter; [23] compresses Bloom filters; [24] supports concise representation and approximate membership queries of dynamic sets; [25] adapts the number of hash functions by query frequency and membership
likelihood of an element to improve the false positive rate; [26] adapts the number of hash functions by the number of appearances of each element, realizing the same functionality of the counting Bloom filter with the same size of the basic Bloom filter.

2.2.4 Background - Bluetooth

There are several steps to use Bluetooth for data transmission: (1) search for nearby devices; (2) discover the services they provide; and (3) pair two devices and establish a connection to use a discovered service. For security reasons, a mobile phone will ask for user permission to initiate or accept a Bluetooth connection as well as a passcode for pairing. Hence an application that relies on Bluetooth connections to transmit data needs user interferences.

Bluetooth consists of a set of protocols that constitute a protocol stack, including the mandatory Service Discovery Protocol (SDP). SDP provides a means for applications to discover which services are available and to determine the attributes of those available services.

The service discovery process consists of several stages. First, information about available devices is collected using the inquiry procedure. Then, a peer-to-peer Bluetooth link (L2CAP) is established to one device. Finally, SDP is utilized for searching desired services using a request/response scheme over the L2CAP transport protocol. SDP itself is a simple protocol that involves communication between an SDP server and a client.

In SDP, each service is represented by a service record that is identified by a 128-bit Universally Unique Identifier (UUID). All information about a service that is maintained by an SDP server is contained within a single service record, which consists
of a list of service attributes. Each service attribute describes a single characteristic of a service (e.g., its name, type, parameters, protocols used) and consists of an attribute ID and the corresponding attribute value. An attribute ID is a 16-bit unsigned integer that distinguishes each service attribute from other service attributes within a service record. The attribute ID also identifies the semantics of the associated attribute value. The attribute value is a variable length field, which can be explored by applications to publish custom data. A client may issue an SDP request to retrieve information from a service record maintained by the SDP server on another device. However, there is a limit on the size of a service record, which may vary on different mobile devices.

2.3 Opportunistic Social Networking: Privacy Preserving Commonality Discovery and Verification

2.3.1 Commonality Discovery Problem

Small talk topics include common friends/contacts and shared interests. Here we abstract all of those as topics or items. The problem can be formulated as follows. Consider a dynamic set $U = \{u_1, u_2, ..., u_N\}$ of $N$ potential communication partners in which each user $u_i$ has a set $Set_{U_i} = \{a_{i,1}, a_{i,2}, \ldots, a_{i,n_i}\}$ of $n_i$ data items. Each $a_{i,j}$ ($1 \leq i \leq N$, $1 \leq j \leq n_i$) is a topic of interest. Any two users $u_i$ and $u_k$ are strangers, which means that they do not share a secret key, and any other user can eavesdrop on the communications between them. Each user $u_i$ wants to discover two things: (1) the identical items in $Set_{U_i}$ and $Set_{U_k}$ ($1 \leq i, k \leq N, i \neq k$); and (2) the corresponding user $u_k$ if there are identical items between $Set_{U_i}$ and $Set_{U_k}$ with the following objectives and constraints:
1. The communication overhead in bytes between $u_i$ and $u_k$ should be small.

2. No items in $SetU_i$ or $SetU_k$ should be disclosed to a passive third party.

3. The honest users $u_i$ and $u_k$ cannot know the other’s items which are not in his own data set.

Furthermore, users $u_i$ and $u_k$ may want to verify/authenticate some or all of the common items/friends in $SetU_i$ and $SetU_k$ before they start a conversation or exchange more sensitive private information, such as the users’ names, or photos.

If users $u_i$ and $u_k$ have some known common friends, we can design an algorithm that generates an encryption key using the identified common friends. Thus, users $u_i$ and $u_k$ are able to establish a secured communication link, in which some sensitive information can be exchanged to facilitate friends verification. Therefore, we separate the original problem into two sub problems. The first sub problem is to discover commonalities between users through a unsecured communication channel. The second sub problem is to verify the common friends which are the results of the first sub problem.

There are a couple ways to discover common topics. For example, (1) A naïve way is to establish a Bluetooth connection between two users’ devices $u_i$ and $u_k$ and transfer the two data sets together to compute their intersection, $SetU_i \cap SetU_k$. However, this requires user interaction to set up Bluetooth connection and transfer a large message. (2) Users $u_i$ and $u_k$ can use the service of a trusted third party. First, they send their item lists to that third party, which discovers the common items, and then they retrieve them from the third party. This method is simple. However, such a third party may not exist. In the real world, many users do not trust a central
service provider due to its terms of use, which can be changed at any time and for any reason without notice and it does not explain how the service provider and its partners will use the personal information. It also requires an Internet connection, which does not always exist, since many mobile phone users do not have a data plan. Most importantly, this solution violates constraint 2. (3) Users $u_i$ and $u_k$ set up a shared session key using a public key scheme, exchange the items to be shared, and match them. The disadvantages of this scheme is that it violates constraint 3. (4) Users $u_i$ and $u_k$ can use public key and homomorphic encryption schemes [27, 28] based private matching and oblivious transfer protocols [29] to extract set intersection. The disadvantages of this type of schemes are the extensive computation and communication overhead on mobile phones, which violates constraint 1.

Alternatively, we encode the data sets in Bloom filters and use the Bluetooth SDP to transmit the Bloom filters. Then we compute their intersection. This approach is non-intrusive and more efficient. Nevertheless, there are two challenges that must be addressed: First, the size of a Bloom filter $m$ grows linearly with the size of a data set $n$ if we want to guarantee a given false positive rate $f$. Second, the size of a Bloom filter is effectively bounded by the maximum size of the SDP service record, which varies on different mobile phones. According to the false positive rate formula, a preferred false positive rate cannot be guaranteed when the data set is large. A suitable compromise must be sought. Several other extensions to the basic Bloom filter exist in the literature [21–26] that are orthogonal to ours. However, the results of these works cannot be readily applied to mobile phones because Bluetooth SDP can only exchange limited information.
We discuss our solution in the rest of this section, which is mainly an iterative protocol.

### 2.3.2 Opportunistic Commonality Discovery Protocol

We aim to achieve the desired false positive rate \( f \) with a minimum total amount of transmission. We devised a multi-round protocol: in each round, a Bloom filter of some false positive rate is published and a subset of common data items is computed. This smaller subset is then encoded with another new Bloom filter to get a low overall false positive rate. After several rounds, eventually the commonalities are reported with the desired false positive rate \( f \).

We specify the algorithm as follows. For simplicity, it assumes that there are only two parties, A and B. Let the interesting data sets of A and B be \( SetU_A \) and \( SetU_B \), respectively. It is not difficult to extend the specification to allow for multiple parties.

**Step (1).** Initially (round 0, no communications), both devices encode their own data sets, \( SetU^0_A = SetU_A \) and \( SetU^0_B = SetU_B \), in two static Bloom filters, \( BF^0_A \) and \( BF^0_B \), respectively, and publish them using a special static attribute ID in the SDP service record.

**Step (2).** In the first round, A first retrieves \( BF^0_B \) via Bluetooth SDP and then checks the membership of each data item in \( SetU^0_A \) against \( BF^0_B \) to obtain a matching set \( SetU^1_A \subseteq SetU^0_A \). Next, A encodes \( SetU^1_A \) into a new dynamic Bloom filter \( BF^1_A \). Finally, A publishes \( BF^1_A \) and the current round number using a new dynamic attribute ID calculated from B’s Bluetooth ID making it specific to B. Symmetrically B takes the same action.
Step (3). In the following \((r+1)\)-th round, A first retrieves \(BF_B^r\) that is specially
generated for A. Step (2) is repeated similarly to generate a new matching set
\(SetU_A^{r+1}\) and a new Bloom filter \(BF_A^{r+1}\) specially for B published with the same
attribute ID as in step (2). This process is repeated until the new matching
set is empty, or the new matching set is the same as that of the last round or
the desired false positive rate is reached. A dynamic attribute is removed from
SDP service record when a predefined lifetime is reached after the end of above
process.

In each round, we replace an old hash function with a new independent hash
function generated by the technique described in [21]. Suppose for any two items,
there exists at least one generated hash function which distinguishes them. In this
way, we can iteratively eliminate the case in which two items have the same set of
hash values. Thus, for any two honest parties A and B, the resulting set \(SetU_A^r\) in
round \(r \geq 1\) between A and B converges to \(SetU_A \cap SetU_B\) as \(r \rightarrow \infty\).

**Theorem 1.** For any two honest parties A and B, the resulting set \(SetU_A^r\) in round
\(r \geq 1\) between A and B converges to \(SetU_A \cap SetU_B\) when \(r\) goes to infinity.

Between two strangers A and B, it is reasonable to assume that the intersection
of their data sets is a proper subset of either original set and the intersection is much
smaller. As the data set size \(n\) decreases, so does the Bloom filter size \(m\) when the
false positive rate \(f\) and number of hash functions \(k\) are fixed. If we keep the Bloom
filter size \(m\) and \(k\) constant, then \(f\) decreases when \(n\) decreases. Therefore, in each
round when a new Bloom filter is constructed, we can either dynamically decrease the
filter size \(m\) according to the current \(n\) and resulting \(f\), or decrease \(f\) dramatically by
keeping the filter size $m$. In practice, we cannot let the round number $r$ go to infinity. We want to keep it within a reasonable small number, thus the resulting common set is a approximation of the real intersection $\text{Set}U_A \cap \text{Set}U_B$.

**Minimizing Discovery Costs:** In the above protocol, let $n_A^0$ be the size of the initial data set (e.g., contacts) $\text{Set}U_A^0$ for any user $A$. Let $n^i_A$ be the size of $A$’s matching set $\text{Set}U_A^i$ in the $i^{th}$ round; let $m^i_A$ be the length of $BF_A^i$, which is the Bloom filter published in the $i^{th}$ round of user $A$; let $f_A^i$ be the false positive rate associated with $BF_A^i$. We define $MTU_A$ as the maximum effective size of any attribute value in the Bluetooth SDP service record (that is used to carry a Bloom filter), and $MID_A$ as the maximum number of attribute IDs that can be used to publish Bloom filters. Let $t_d$ be the average time needed for searching devices in the physical proximity, and $t_s$ be the average time for retrieving one Bloom filter from a known device. Assume that $k$, the number of hash functions used for computing a Bloom filter, remains the same in our protocol. Suppose that the protocol terminates at the $r^{th}$ round which is unknown and takes time $T$ in total. We have the following constraint optimization problem:

$$\min\left(\sum_{i=0}^{r-1} m^i_A + \sum_{i=0}^{r-1} m^i_B\right)$$  \hspace{1cm} (2.2)

which is subject to the following constraints:

$$f = \prod_{i=0}^{r-1} f_x^i \leq f_0; x \in \{A, B\}$$  \hspace{1cm} (2.3)

$$0 < m_x^i \leq MTU_x; x \in \{A, B\}$$  \hspace{1cm} (2.4)

$$r \leq MID_x; x \in \{A, B\}$$  \hspace{1cm} (2.5)

$$t_d + r \times t_s = T \leq T_0;$$  \hspace{1cm} (2.6)
In the above, constraint (2.3) denotes that the overall false positive rate $f$, computed by the product of all $f_i$, is less than a given target value $f_0$; constraint (2.4) denotes that the length of every Bloom filter is less than the maximum possible value; constraint (2.5) denotes that the number of Bloom filters published is less than the limit, assuming that each Bloom filter takes one attribute ID; and constraint (2.6) denotes that the total time used in device inquiry and Bloom filter exchange is less than a permissible value $T_0$.

The Benefits of Our Protocol: Our multi-round protocol essentially proposes an iterative Bloom filter (IBF) scheme that can significantly reduce the size requirements of Bloom filters and hence improve system efficiency without sacrificing precision of the false positive rate. Now we illustrate the performance benefits of our protocol with the following example. Suppose that a user has 500 contacts, a phone number is expressed as 10 digits, and the average length of first names and last names is 10 characters. (The actual length of both names may be much longer.) In a naïve approach in which we publish a list of phone numbers and names, we need $500 \cdot (10 + 10) = 10000$ bytes. Figure 2.1 gives a table for calculating the Bloom filter size with different numbers of contacts and false positive rates. To guarantee a 0.01% false positive rate, we need a Bloom filter with 1401 bytes. However, using our protocol in 2-round, we can reduce this size requirement. Given the same contact list, we can set the false positive rate in each round to 1% (with the overall false positive rate 0.01%). In the first round, we need 600 bytes for the first Bloom filter. We obtain an initial common set with $500 \cdot 1\% = 5$ false positives. In the second round, we need only build a Bloom filter to address this reduced set while keeping the false positive rate at 1%. Regardless of whether there are true common elements, the second Bloom
Figure 2.1: Pre-computed Bloom filter size (in bytes) with regard to data set size (N) and desired false positive rate (F).

<table>
<thead>
<tr>
<th>N/F</th>
<th>5%</th>
<th>2%</th>
<th>1.5%</th>
<th>1%</th>
<th>0.1%</th>
<th>0.01%</th>
<th>0.001%</th>
<th>0.0001%</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>5</td>
<td>6</td>
<td>6</td>
<td>7</td>
<td>10</td>
<td>15</td>
<td>21</td>
<td>30</td>
</tr>
<tr>
<td>10</td>
<td>9</td>
<td>11</td>
<td>12</td>
<td>13</td>
<td>19</td>
<td>29</td>
<td>41</td>
<td>59</td>
</tr>
<tr>
<td>25</td>
<td>21</td>
<td>26</td>
<td>28</td>
<td>31</td>
<td>47</td>
<td>71</td>
<td>103</td>
<td>147</td>
</tr>
<tr>
<td>50</td>
<td>42</td>
<td>52</td>
<td>56</td>
<td>61</td>
<td>94</td>
<td>141</td>
<td>205</td>
<td>283</td>
</tr>
<tr>
<td>100</td>
<td>83</td>
<td>104</td>
<td>110</td>
<td>120</td>
<td>188</td>
<td>281</td>
<td>408</td>
<td>566</td>
</tr>
<tr>
<td>150</td>
<td>125</td>
<td>155</td>
<td>166</td>
<td>180</td>
<td>282</td>
<td>421</td>
<td>612</td>
<td>878</td>
</tr>
<tr>
<td>200</td>
<td>165</td>
<td>207</td>
<td>220</td>
<td>240</td>
<td>376</td>
<td>561</td>
<td>816</td>
<td>1170</td>
</tr>
<tr>
<td>250</td>
<td>208</td>
<td>258</td>
<td>275</td>
<td>300</td>
<td>470</td>
<td>701</td>
<td>1020</td>
<td>1463</td>
</tr>
<tr>
<td>300</td>
<td>249</td>
<td>310</td>
<td>330</td>
<td>360</td>
<td>563</td>
<td>841</td>
<td>1224</td>
<td>1755</td>
</tr>
<tr>
<td>350</td>
<td>291</td>
<td>362</td>
<td>366</td>
<td>420</td>
<td>657</td>
<td>981</td>
<td>1428</td>
<td>2048</td>
</tr>
<tr>
<td>400</td>
<td>332</td>
<td>413</td>
<td>440</td>
<td>480</td>
<td>751</td>
<td>1121</td>
<td>1632</td>
<td>2340</td>
</tr>
<tr>
<td>450</td>
<td>374</td>
<td>465</td>
<td>495</td>
<td>540</td>
<td>845</td>
<td>1261</td>
<td>1836</td>
<td>2633</td>
</tr>
<tr>
<td>500</td>
<td>416</td>
<td>516</td>
<td>550</td>
<td>600</td>
<td>939</td>
<td>1401</td>
<td>2040</td>
<td>2925</td>
</tr>
<tr>
<td>550</td>
<td>457</td>
<td>568</td>
<td>605</td>
<td>660</td>
<td>1032</td>
<td>1541</td>
<td>2244</td>
<td>3217</td>
</tr>
<tr>
<td>600</td>
<td>498</td>
<td>619</td>
<td>660</td>
<td>720</td>
<td>1126</td>
<td>1681</td>
<td>2448</td>
<td>3510</td>
</tr>
<tr>
<td>650</td>
<td>540</td>
<td>671</td>
<td>715</td>
<td>780</td>
<td>1220</td>
<td>1821</td>
<td>2652</td>
<td>3802</td>
</tr>
<tr>
<td>700</td>
<td>581</td>
<td>723</td>
<td>770</td>
<td>840</td>
<td>1314</td>
<td>1962</td>
<td>2856</td>
<td>4095</td>
</tr>
<tr>
<td>750</td>
<td>623</td>
<td>774</td>
<td>825</td>
<td>900</td>
<td>1408</td>
<td>2102</td>
<td>3060</td>
<td>4387</td>
</tr>
<tr>
<td>800</td>
<td>666</td>
<td>826</td>
<td>880</td>
<td>960</td>
<td>1501</td>
<td>2242</td>
<td>3284</td>
<td>4690</td>
</tr>
<tr>
<td>850</td>
<td>708</td>
<td>877</td>
<td>935</td>
<td>1020</td>
<td>1595</td>
<td>2388</td>
<td>3496</td>
<td>4972</td>
</tr>
<tr>
<td>900</td>
<td>750</td>
<td>929</td>
<td>990</td>
<td>1080</td>
<td>1689</td>
<td>2522</td>
<td>3672</td>
<td>5255</td>
</tr>
<tr>
<td>950</td>
<td>792</td>
<td>980</td>
<td>1045</td>
<td>1140</td>
<td>1783</td>
<td>2656</td>
<td>3876</td>
<td>5557</td>
</tr>
<tr>
<td>1000</td>
<td>833</td>
<td>1032</td>
<td>1100</td>
<td>1100</td>
<td>1877</td>
<td>2802</td>
<td>4079</td>
<td>5348</td>
</tr>
</tbody>
</table>

filter’s overhead that accounts for the 5 false positives is 7 bytes. Therefore the worst-case overhead is 607 bytes, which is almost 60% saving compared to the 1401 bytes required by the basic Bloom filter. More savings are expected when the number of contacts grows larger and the number of rounds increase. This example is illustrative enough to show the merits of our protocol over the basic Bloom filter.

### 2.3.3 Opportunistic Commonality Verification Protocol

We also designed a method that makes the topics verifiable. Note that not all topics can be verified. For example, a user can claim sports especially swimming is his most favorite hobby in spite of the fact that he cannot swim at all and he hates...
swim. On the contrary, friendship or contact-ship can be verified. We extend the vCard [30] (a file format standard for electronic business cards) to include dedicated information and digital signatures as shown in figure 2.2. For example, in Figure 2.2, Alice and Bon are friends. “Bob Smith” is the card issuer; Alice is the vCard holder; and “12:34:45:78:90” is the Bluetooth address of Alice’s phone. We assume that extended vCards are exchanged using Bluetooth enabled mobile phones when friends meet. An extended vCard issuer application on the phone first obtains the other parties Bluetooth address, fills the relationship field, generates a digital signature and transfers the resulted vCard to the card holder.

Given the contact information is encoded in vCard, we specify the friendship verification protocol as follows. For simplicity, it assumes that there are only two parties, A and B. Let the approximate common data sets obtained by A and B via the previous opportunistic commonality discovery algorithm be $Set_{UB}$ and $Set_{BU}$. It is not difficult to extend the specification to allow for multiple parties.
Step(1) A and B find the top \( k \) items respectively in the set \( Set_{AB} \) and \( Set_{BA} \) in alphabetical order.

Step(2) A and B form two new strings \( S_{AB} \) and \( S_{BA} \) respectively by concatenating the top \( k \) items in sequence. i.e. \( S_{AB} = S_{Item_{1}} | | S_{Item_{2}} | | \cdots | | S_{Item_{k}} \); and \( S_{BA} = S_{Item'_{1}} | | S_{Item'_{2}} | | \cdots | | S_{Item'_{k}} \). Where \( | | \) represents string concatenation operation; \( S_{Item_{i}} \) is a string representation of \( Item_{i} \). For example, if \( Item_{i} \) is a vCard from a friend “Bob Smith” with the mobile phone number “16146880066”, then \( S_{Item_{i}} = “BobSmith@16146880066” \). \( k \) is a number agreed by A and B, which can be set up during the last step of the opportunistic commonality discovery protocol.

Step(3) A and B generate 8-hexadecimal pin numbers \( P_{AB} \) and \( P_{BA} \) respectively by using SHA1 hash function, and the strings \( S_{AB} \) or \( S_{BA} \). One possible way is as follows. First, the 20 bytes SHA1 hash of \( S_{AB} \) is arranged into the form of byte 0 to byte 19. Then let the index be the low-order 4 bits of byte 19, which has a range of 0-15. Next A takes 4 consecutive bytes from the hash starting from the index. Finally this 4-byte binary code is represented as 8 hexadecimal digits.

Step(4) A and B try to do Bluetooth pairing using the two pin numbers. If the pin numbers match, then an encrypted communication link can be setup. Otherwise, stop.

Step(5) A and B exchange their extended vCards in the approximate common set via the encrypted Bluetooth link.
Step(6) If A is able to verify the digital signatures of B’s extended vCards, A can release/send his private information (e.g. name and photo) to B. Otherwise, A close the connection and stop. Similarly, B take the same action.

2.4 E-SmallTalker System Design, Implementation and Evaluation

To demonstrate the effective of our protocol, we developed a system called E-SmallTalker, which suggests meaningful small talk topics between co-located strangers. In this section, we first present the system architecture and then discuss the implementation detail. Afterwards, we give the system’s evaluation result.

2.4.1 System Architecture

Figure 2.3 shows the E-SmallTalker system architecture, which includes the following four software components:
**Context Data Store:** This component stores user data that contributes to small talk and metadata that controls system operation. Small talk between two people is often highly *situated*: topics may cover mutual friends and hobbies, schools both attended, places both visited, and historical facts such as their meeting last year in the same conference. System metadata includes the Bloom filter parameters and user preferences such as what data may and may not be published (in Bloom filters to strangers). For the scope of this chapter we simplify this part and assume that friends’ information is imported from the phone contact database and that the user’s interests are represented in a limited vocabulary of keywords.

**Context Encoding & Matching:** This component encodes the context data to be published by the Context Exchange component. The data is encoded using Bloom filters with user-configured parameters. To save computation, the Bloom filters are computed and cached in the Context Data Store when the parameters are changed or the encoded data are updated. The Bloom filters are retrieved by the Context Exchange to find matching elements when a query is received.

**Context Exchange:** This component includes a Bluetooth server (BTS) and a Bluetooth client (BTC). The BTS creates the service record with Bloom filters as service attribute values and publishes the service record via the Bluetooth SDP server. The BTC first performs an inquiry over the Bluetooth radio to retrieve the MAC address of any device in the physical proximity at the time of inquiry; then it discovers whether or not the device is running E-SmallTalker. If this is the case, it retrieves the Bloom filters that the device publishes.

**User Interface (UI):** This component provides interfaces for a user to configure small talk policies and rules. For example, the user can specify which contacts to
encode in his Bloom filter; what personal profile data to publish, e.g., name, age, phone number, gender, and interests; Bloom filter parameters to encode his own information; and conditions by which a received Bloom filter will be ignored. The user can also specify how he is notified of matching information: text display, ringtone, vibration, or speech. For the scope of this chapter, we also simplify this part and assume that the Bloom filter encodes all friends’ names plus mobile phone numbers.

The four components work together to help users initiate small talk. The corresponding workflow is as follows: First, our E-SmallTalker system uses the Context Data Store to record user-customized profile information. Then, the Context Encoding & Matching component compresses and encodes information from the Context Data Store into Bloom filters. When two users are in Bluetooth communication range, their Context Exchange components automatically exchange their Bloom filters to find their similarities via Bluetooth SDP. In different social scenarios, users can easily set up corresponding settings via E-SmallTalker’s User Interface component. This chapter focuses on the two most critical components: Context Exchange and Context Encoding & Matching.

*Context Encoding & Matching:* To discover possible matches based on Bluetooth SDP, we use our new Bloom filters based opportunistic commonality discovery protocol. The goal is to minimize system overheads and error rate under the constraint that Bluetooth SDP can only exchange limited information.

In our implementation, the length of a Bloom filter in each round should be less than or equal to 128 bytes, which is the maximum effective size of any attribute value in the Bluetooth SDP service record, and the number of rounds should be less than 10, which is the maximum number of attribute IDs that can be used to publish Bloom
Algorithm 1 The 2-Round Discovery Protocol for User $u_0$

1: [Round_1:]
2: Initialize Bloom filter $BF_0$;
3: for each item $a_{0,j}$ in $SetU_0$
4: calculate $k$ hash indices of item $a_{0,j}$;
5: set the corresponding bit of $BF_0$ to 1;
6: end for
7: publish $BF_0$ with a static attribute ID;
8:
9: [Round_2:]
10: acquire the dynamic device set $U$;
11: for each user $i$ in set $U \setminus \{u_0\}$
12: retrieve the Bloom filter $BF_i$ from $u_i$;
13: for each item $a_{0,j}$ in $SetU_0$
14: if $BF_i.contains(a_{0,j})$ then
15: add $a_{0,j}$ to common set $CS_i$;
16: end if
17: end for
18: if $CS_i.size > 0$ then
19: initiate a new Bloom filter $BF'_0$;
20: encode $CS_i$ into $BF'_0$ with different set of $k$ hash indices;
21: publish $BF'_0$ with a dynamic attribute ID;
22: end if
23: end for
24: for each user $i$ with $CS_i.size > 0$
25: retrieve the new Bloom filter $BF'_i$ from $u_i$;
26: for each item $a_{0,j}$ in $CS_i$
27: if $BF'_i.contains(a_{0,j})$ then
28: add $a_{0,j}$ to common set $CS'_i$;
29: end if
30: report common set $CS'_i$ with user $u_i$;
31: end for

filters. We found that the size of the initial data set (e.g., contacts) $SetU^0_A$ for any user $A$ is a few hundred, the protocol converges after a few rounds, and two rounds are normally enough for strangers. Hence it is effectively a 2-round protocol.

Algorithm 1 specifies our 2-round protocol for publishing user data and discovering matches. The same algorithm is executed on every device in question. In the first round, we initiate Bloom filter $BF_0$ and insert all the user data items into $BF_0$. Then
the resulting $BF_0$ is published. In the second round, the Bloom filter are retrieved from all nearby devices. If matches are found with the Bloom filter $BF_i$ from any user $u_i$, we store the data in the corresponding set $CS_i$. To achieve a preferred false positive rate, we build a new Bloom filter $BF'_0$. The calculation exactly follows the above analysis of the constraint optimization problem for the 2-round protocol. Then, $BF'_0$ is published. Finally, by steps 25–30, we collect the set of common interests $CS'_i$ with target user $u_i$ and report it to the local user.

**Context Exchange:** A common way of using Bluetooth to exchange information involves several steps: (1) search for nearby devices; (2) discover the services they provide; and (3) pair two devices and establish a connection to use a discovered service. In step (3), a mobile phone will ask for user permission to accept Bluetooth connections as well as a passcode for pairing. This is common practice for most mobile phone manufactures and service providers. Hence, an application that relies on Bluetooth connections to transmit data will wait for or interrupt the users. If a phone need to be communicated with multiple phones, multiple pop up windows waiting for user interactions hassle the user a lot. For users of our system, this is too intrusive and irritating. We need to find a way for two phones to communicate with each other without establishing a Bluetooth connection. We achieve this by using Bluetooth’s Service Discovery Protocol (SDP) to exchange encoded user information.

The trick is using Bluetooth service attribute values to publish Bloom filters. SDP provides a means for applications to discover which services are available and to determine the attributes of those services. In SDP, each service is represented by a service record that is identified by a 128-bit Universally Unique Identifier (UUID). All information about a service that is maintained by an SDP server is contained
within a single service record, which consists of a list of service attributes. Each service attribute describes a single characteristic of a service (e.g., its name, type, parameters, protocols used) and consists of an attribute ID and the corresponding attribute value. The attribute value is a variable length field, which is explored by our system to publish custom data.

Our system takes advantage of this structure. We create a service record by starting a virtual service with a known UUID and a list of attributes. We use attribute values to publish information encoded in Bloom filters. We call it a “virtual service” because it provides no service in the traditional sense that a client can consume or to which a client can connect. It exists only to publish attributes. By updating the service record to the SDP servers, two Bluetooth devices can exchange information without setting up a Bluetooth connection.

Unlike Wi-Fi, there is no broadcast channel or beacon signals in Bluetooth due to the fact that it uses a frequency-hopping spread spectrum radio technology. In order to do communication, a Bluetooth slave device must follow the master’s hop pattern which cannot be generated without knowing the master’s address/clock values. Therefore, our system adopts a pull model, not a push model. A device is not broadcasting its service; rather, it is publishing its service, waiting for another device to discover it and retrieve the information.

An alternative approach to exchange information without setting up a Bluetooth connection is to use the Bluetooth device name, which consists of a maximum of 248 bytes of text data. Bluetooth SDP can publish more information than a single name. The size of custom data (the number of attributes times the length of attribute values) in a service record varies on different mobile devices. For example, in our experiment
a successful communication between a Sony-Ericsson W810i phone and a Nokia N82 can exchange up to 10 attributes, each of which has a maximum of 128 bytes of data. Two Nokia N82s can exchange more attributes and more bytes of data in each attribute. In addition, SDP provides more flexible and fine-grained control as we are able to change an attribute value individually.

The Context Exchange component in Figure 2.4 shows an overview of Bluetooth-SDP-based communication in our system. The Bluetooth server publishes Bloom filters through the phone’s SDP server. The Bluetooth client acquires the other phone’s Bloom filters by sending a SDP request to its SDP server.

To find common interests or friends, our E-SmallTalker system first encodes all contact information or the keywords for interest into a Bloom filter. Such a system can retrieve another system’s Bloom filters via the context exchanging component we discuss here. If two E-SmallTalker systems share a common contact or a common interest, it can be found by using Algorithm 1, which is described earlier.
2.4.2 System Implementation

We choose Java ME as our prototype development environment because it is supported on most mobile phones on the market. We have implemented the E-SmallTalker system and tested it on several brands of mobile phones, including Sony Ericsson (W810i) and Nokia (5610xm, 6650, N70, N75, N82). It is important for a social networking system like E-SmallTalker to run across a wide variety of phones: it is useful in practice only when a critical mass of users are using it.

We implemented the system with the Eclipse SDK and the Sun Java Wireless Toolkit for Connected Limited Device Configuration (CLDC) based on the Mobile Information Device Profile (MIDP) specification. We used the Java APIs for Bluetooth described in the JSR-82 interface for developing service discovery applications. We imported some Java code from the XSiena (eXtended Scalable Internet Event Notification Architecture) project for the hash functions used in our Bloom filters. The size of the deployed executable is about 127KB.

When a user starts the application for the first time, it asks him to configure his personal profile (e.g., name, telephone number, interests, past experience) and the system settings (e.g., automatic or manual discovery, time period between each discovery). The system imports the contacts from the address book and lets the user choose which ones will be published in the Bloom filter. Then the system performs the following major operations: (1) generating and publishing the Bloom filters through Bluetooth SDP according to the user’s configurations, (2) retrieving the Bloom filters from nearby mobile phones, (3) matching the common interests or contacts, and (4) prompting the user when there are matches.
2.4.3 System Performance Evaluation

Experiments Setup

There are two classes of metrics to evaluate E-SmallTalker’s performance: system performance and social performance. System performance includes discovery performance and power consumption. Social performance mainly includes the likelihood that strangers in physical proximity share common interests. In order to evaluate social performance, we need to run a massive field test, which is very costly. As a first step of performance evaluation, we focus on system performance. System performance has a critical impact on whether people accept our system, which is the key challenge of our design. We plan to run a massive field test to measure social performance in our future work.

One key system performance is discovery performance. To evaluate the discovery performance in E-SmallTalker, we focus on two detailed evaluation metrics: (1) Successful discovery rate. This is defined as the percentage of successful discoveries among all attempts to search among nearby users. Given an E-SmallTalker user $A$, assuming that he shares common interests with $n$ nearby users, a successful discovery is one that reports all the matches among user $A$ and the $n$ nearby users. (2) Commonality discovery time. This is the period from the time of starting a search to the time of finding someone with common interest. This can be considered as an end-to-end delay.

Another key system performance is power consumption. Since Bluetooth communication dominates E-SmallTalker’s power consumption, we measure power consumption in the two major communication parts: Bluetooth device discovery and Bloom
filter retrieval (i.e., Bluetooth service discovery). We also measure power consumption when E-SmallTalker is always on.

In our experiments, we use 6 mobile phones (a Sony Ericsson W810i, a Nokia 5610xm, a Nokia 6650, and 3 Nokia N82s). Each experiment is repeated 10 times to average the results. We conduct a survey of 35 college students that reveals that they have, on average, 143 contacts in their mobile phones. From this result, we set the number of contacts per phone to \( n = 150 \) in the experiments. In addition, we used seven hash functions \( (k = 7) \) for all Bloom filter insertion and lookup operations.

Since Bluetooth’s technical details influence our experiments, we conduct experiments considering the following three factors: (1) the Bluetooth search interval, i.e., the time between two consecutive searches for Bluetooth devices and services in the application, (2) the number of nearby devices, and (3) the distance between two devices, which is 4 meters by default.

**Successful Discovery Rate and Discovery Time**

We first conduct experiments to measure how the Bluetooth search interval and the number of users (devices) impact discovery performance.

Intuitively, if the Bluetooth search interval is too long, the user may miss some potential opportunities for social interactions. On the other hand, if the interval is too short, the chance of running into collisions increases significantly and causes discovery failures, especially when multiple mobile phones are in physical proximity. This is caused by current limitations of the Bluetooth protocol as will be discussed later. For similar reasons, the more devices in physical proximity, the higher the chance of running into collisions and discovery failures.
In this group of experiments, we put the phones in close proximity (about 4 meters apart if possible). The Bluetooth search intervals were set to 60 or 120 seconds and the number of phones was iteratively increased from 2 to 6.

With regard to the number of phones under different searching intervals, Figures 2.5 and 2.6 show the minimum, average and maximum discovery time, and Figure 2.7 shows the successful discovery rate.

Figure 2.5: Discovery Time with 60s Bluetooth Searching Interval.

Figure 2.6: Discovery Time with 120s Bluetooth Searching Interval.
Figure 2.7: Successful Discovery Rates under 60s/120s Bluetooth Search Intervals.

From the experimental data, by fixing the number of devices, the Bluetooth searching interval does not show any significant impact on discovery time. The average discovery time when the interval is 60 s is only slightly longer than the case of 120 s. This can be explained by the fact that the discovery time is actually dominated by the Bluetooth device discovery time, which is on the order of magnitude of 10 s [31]. On the other hand, it is clear from the data that as the number of devices increases, the discovery time increases. Furthermore, both the number of devices and the search intervals show a clear impact on the successful discovery rate. The success rate drops as the interval decreases and the number of devices increases.

In the next group of experiments, we seek to understand how the distance between devices affects the common interest discovery performance. We place two Nokia N82 phones on the floor 1 meter apart, run the experiment 10 times, and measure the average discovery time and successful discovery rate. Then we repeat the experiments by increasing the distance 1 meter at a time until the two phones are 10 meters apart.
Figure 2.8 shows the minimum, average and maximum discovery times versus distance between two phones. Figure 2.9 presents the success rate versus distance. In summary, the minimum, average and maximum discovery times in our experiments are 13.39, 20.04 and 58.11 seconds respectively among E-SmallTalker on all the data we collected. The overall success rate is 90%. From the data, there is no clear trend of how the distance affects the average discovery time or the success rate, although the max discovery time becomes flicky when the distance increases from 2 meters to
5 meters. This result demonstrates that E-SmallTalker’s performance is quite stable within 10 meters. However, when two phones are placed more than 10 meters apart, which is the nominal communication range for a standard Bluetooth device, they cannot find each other.

According to the Bluetooth specification [31], “the inquiry substate may have to last for 10.24 seconds unless the inquirer collects enough responses and determines to abort the inquiry substate earlier.” In a noisy environment, there is no guarantee of successful inquiry. In such situations, the inquiry time may far exceed the default time of 10.24 seconds. In the inquiry substate, a Bluetooth device will not respond to another device. As a consequence, if two mobile phones start searching for devices at approximately the same time, there is a collision and they cannot obtain each other’s service information in a timely manner. With an increased number of devices or a reduced Bluetooth searching interval, the probability of collision increases, which leads to a lower successful discovery rate and a longer discovery time.

**Power Consumption**

To study trends regarding power consumption, we place two Nokia N82 phones 4 meters apart, both fully charged. A Symbian application, the Nokia Energy Profiler (called NEP or Juice), is used to read the battery voltages, currents, and energy consumptions once per second.

First, we measure one N82’s energy cost for Bluetooth device discovery and service discovery (for the purpose of Bloom filter retrieval), respectively. Figure 2.10 shows the accumulated energy consumption for 1000 rounds of device discovery and service discovery. From the data we can see that device discovery consumes significantly more energy than service discovery. Hence, the service discovery part in our 2-round
protocol has little impact on the overall power efficiency of the application, which is dominated by the device discovery part.

Then we monitor the power consumption in device discovery with nonstop (i.e., continuous), 60 s, and 120 s Bluetooth search intervals. Figure 2.11 shows the results of a fully charged Nokia N82 for 8 hours. It shows the clear trend that the energy consumption decreases when the Bluetooth search interval increases.

Finally, we record the run time of E-SmallTalker on a fully charged N82 phone (1050 mAh) until the battery is exhausted. In our experiment, we let the phone run in standby mode while turning on the cellular and Bluetooth radios. When the search interval is set at 60 s, the phone runs slightly longer than 29 hours. Using the same settings on the cellular and Bluetooth radios, we redo the experiment without running E-SmallTalker. In this case, the phone runs 32 hours until the battery is depleted. This result is encouraging since a user will likely only need to run E-SmallTalker when necessary, e.g., in specific social settings.
2.5 Discussions

Security Considerations: Our design assumes that users are willing to share personal information at some level with strangers without their awareness. Our design can provide privacy against eavesdroppers (passive attacks) due to the fact that Bloom filters use one-way hashing to encode the topics, making it very difficult to reconstruct the list of topics in a filter without performing an exhaustive search of the topic-value space. To provide privacy against active attackers, we can utilize research results in the area of privacy-preserving set intersection, such as the cryptography protocols in [29]. However, their protocols incur extensive computation and communication costs, which do not fit in our situation. Our design is a trade-off between privacy and performance given the constraints imposed by Bluetooth. We also assume that the communicating parties are in physical proximity, i.e., within 10 meters, which is the nominal communication range for class II Bluetooth devices. Several possible attacks may compromise the users’ privacy:

Figure 2.11: Energy Cost in Device Discovery with Different Bluetooth Search Intervals.
An attacker can publish a Bloom filter consisting of a high percentage of random 1’s such that everyone in the vicinity would recognize him as having common friends. Although this attack leads our system provide false topics for the honest user, it does help the user start a conversation. To defend against this attack, we can modify E-SmallTalker to check that the percentage of 1’s in any Bloom filter circulated in our system is below a certain threshold. We can also use the friendship verification protocol described in section 2.3.3 to test the resulting common friends. Even though there is no extended vCard stored in the phone, we can still use the first four steps to test whether a common pin code can be agreed.

An attacker may launch a man-in-the-middle attack by intercepting and forwarding other people’s Bloom filters. On one hand, since the purpose of our system is just to provide topics for facilitating a conversation among strangers, the information being transferred is unlikely very sensitive. On the other hand, the middleman has no context information to decode the bits in an intercepted Bloom filter and it is not easy to infer any useful information.

An attacker may also copy contact information from a telephone directory and store it in bulk to his mobile phone. As a result, many users might recognize his as having common friends. However, this kind of attack is not very effective given the large number of contacts imported from the phone book. For example, suppose that the false positive rate is 0.1% and the number of contacts in the phone book is 300,000. Then the number of matches is $300,000 \cdot 0.1\% = 300$, which is very high considering that a person normally has only a few hundred contacts. Furthermore, this type of attack can be detected because of its unusually high percentage of 1’s caused by the size limit of Bloom filters imposed by Bluetooth SDP. As in the first
case, we can modify E-SmallTalker either to ignore such Bloom filters or to verify the
friendship.

**User Experience:** Our current implementation works the best when there are only
two users in proximity. At the system level, the discovery protocol can identify all
devices with matching interests. However, partly due to Bluetooth limitations, our
system cannot yet tell who owns which devices. To identify this, one possible solution
is to exchange pictures or some descriptions (such as shirt color) about the users after
common topics are found and some privacy policies are met. Other complementary
techniques such as Point&Connect [19] can be leveraged for one to point his device to
the device of an interesting person. We plan to extend the system such that the user
can tell specifically with which persons in a crowd he can talk about certain topics.

E-SmallTalker can also be extended to collect a rich variety of context information
such as the location and the time by which two users meet, who else is around when
they meet, and the topics about which they conversed. It is also possible to retrieve
public information about the users, e.g., from their personal web sites or other social
networking services. Such information can make small talk topics more interesting.

Furthermore, it is important to reduce the total discovery time for a better user
experience. By our experiments, the Bluetooth device discovery time accounts for
over 90% of the time cost of our application. We plan to extend the system by
reducing the Bluetooth device discovery time. We point to recent results that aim to
accelerate the Bluetooth device discovery, such as Scott et al. [32] and Woodings et
al. [33]. These results can be leveraged to further improve performance of our system.
2.6 Summary

In this chapter, we presented E-SmallTalker, a mobile phone-based distributed system for social networking in physical proximity among strangers. Our system suggested common topics for users to initiate significant conversations. It leveraged Bluetooth SDP to exchange these topics without establishing a connection. We customized service attributes to publish non-service-related information. We proposed a novel opportunistic commonality discovery protocol based on our new iterative Bloom filters that encode topics to fit in SDP attributes to achieve a low false positive rate. Our approach was efficient in computation and communication. We have implemented the system and evaluated its performance in real-world phones. Our experiments and analysis illustrated its promise for social interactions in physical proximity.
CHAPTER 3

OPPORTUNISTIC LOCALIZATION IN WIRELESS SENSOR NETWORKS

In this chapter, we propose an “opportunistic localization system” to localize sensor nodes. Since one particular important application of the opportunistic localization system is to protect an important area from being under surveillance by an adversary’s sensor nodes, we call it an “anti-sensor network system” and use the two terms interchangeably in this chapter. The major components of the system are a set of observing points (monitors) deployed in the area of importance. The observers take opportunities to localize sensor positions using antenna arrays to measure direction of arrival (DoA) and received signal strength of the signals emitted by sensors. Once sensors are localized, additional measures, which are out the scope of this dissertation, are taken to physically remove or disable localized sensors. We designed a set of opportunistic localization algorithms to handle additional counter-measures that can be employed by sensors, including message encryption and non-uniform transmission power levels. The proposed opportunistic localization algorithms are designed based on analytical performance measures, and evaluated through simulations.
3.1 Motivation

Wireless Sensor Networks (WSNs) have been envisioned to deliver in-situ observations from inaccessible and inhospitable areas [34]. The majority of existing WSN applications are security related applications where sensor nodes detect and report intruders and other events that may pose a threat to a given asset. A logical consequence of such deployment scenarios is the use of a counter-acting system to prevent sensor networks from fulfilling their missions. Such counter-acting systems would be deployed by the owners and operators of a target asset.

In this chapter, we introduce the concept of the anti-sensor network system which aims to prevent the operation of a sensor network in a given area. We also design new opportunistic localization algorithms to support operations in the anti-sensor network system. The basic idea behind our proposed opportunistic localization system is the observation of the sensor communication activity to find sensor node locations. We envision a set of fixed observer nodes, referred to as monitors, intercepting the communication of wireless sensor nodes. Based on the observations of communication events at multiple monitors, the system tries to find the location of the sensor nodes in the area to be protected. Monitors estimate the distance to the signal source using received signal strength, and the direction of arrival (DoA) using their antenna arrays. Once sensor locations are determined, additional measures are taken to physically remove or disable localized sensors.

We also envision that sensor nodes trying to avoid being detected by the opportunistic localization system. Two counter-measures that can be used by sensors are the encryption of transmitted information to hide their IDs, and the variation of
transmission power to make RSS-based distance estimations unreliable. The countering functions and measures of the two systems create a challenging environment for the sensor network to maintain its presence, and for the localization network system to localize adversary sensor nodes. The opportunistic localization procedures proposed in this work go beyond the standard localization methods presented in the past. Since the signal sources are hidden due to encryption, the use of multiple samples from the same source depends on the correct mapping of received signals to their actual sources. We propose two opportunistic localization algorithms to accomplish this mapping and to improve the localization performance.

To the best of our knowledge, the proposed anti-sensor network is unique in that it aims to protect a given asset against sensor network-based observations. There are many potential application areas of an anti-sensor network system, ranging from protection of sensitive civilian infrastructures such as airports to the obvious military applications. The proposed opportunistic localization system architecture primarily depends on processing the received signals at observation points. While the core of the proposed methods partially overlaps with the sensor localization work reported earlier [35,36], a non-cooperative/hostile set of signal sources has not been considered in the past. To mitigate the problems arising from the non-cooperative nature of the signal sources and detectors, we design robust and resilient algorithms for location detection and signal correlation. Another important property of our proposed work is that it allows for graceful degradation of the system performance under these adverse conditions.
3.2 Related Work

Localization in WSNs have been studied extensively in the past to determine the locations of sensor nodes [37]. The use of GPS devices in low cost sensor nodes is not feasible due to cost constraints. Instead, many localization systems rely on a smaller number of higher capability landmark nodes equipped with GPS devices that emit beacons for other sensor nodes to determine their locations. The resulting solutions, referred to as *measurement-based methods*, estimate distance and/or direction of arrival of beacon signals to determine sensor locations. The distance information can be obtained through time of arrival, time difference of arrival, or received signal strength measurements. To determine the position of a sensor node in a two-dimensional coordinate system, at least three measurements from non-collinear reference points are needed. Examples of such localization methods include Cricket [36], AhLOS [38], APS [35] and, RADAR [39]. *Connectivity-based techniques* only use proximity information to derive the location of sensors. Sensor nodes only know what nodes are nearby through ordinary message exchanges, but not how far away these neighbors are or in what direction they lie. These methods, often also called range-free techniques, compute sensor locations iteratively. Examples of connectivity-based techniques include centroid algorithm [40], DV-HOP [41], MDSMAP [42], and APIT [43]. Although these and other localization methods only consider the positioning in a cooperative (or at least non-hostile) situation, these methods inspire us to solve our new problem.

Another group of work related to our proposed system is about the physical security of WSNs. In [44], the authors identified and modeled blind physical attacks, where they studied the issue of deployment of sensors in a sensor network to meet lifetime requirement. They have also focused on the search-based physical attacks
in [45]. To protect against sensor tampering, one defense involves tamper-proofing the node’s physical package [46]. While these proposals acknowledge the possibility of attacks on sensor networks, they focus on how to protect sensors’ physical security via improved hardware or how to prolong the operation time of a sensor network.

### 3.3 The Opportunistic Localization System Architecture

The primary aim of an anti-sensor network is to protect a given observation area from being monitored by sensor nodes. To this end, an anti-sensor network must take opportunistic to identify the presence and the location of possible sensors in an observation area in a fast and accurate manner. The general operation scenario can be regarded as a clash of two opposing systems: Sensors try to observe a protected area and collect information from the field, and the anti-sensor network tries to prevent sensors from doing so. At the same time, sensors use counter-measures to avoid (or at least delay) being detected. As in any adversarial interaction, opposing systems may utilize multiple measures and counter-measures. Within the scope of this study, we consider and analyze two easily implementable yet effective counter-measures as outlined in Section 3.3.2.

#### 3.3.1 System Description

The anti-sensor network is comprised of a number of fixed monitors located at known positions. We assume that the monitors do not have any resource constraints in terms of energy and processing power, and are capable of receiving messages transmitted by any potential sensor in their observation area. Monitors are also capable of communicating among themselves using separate frequencies over dedicated channels. Hence, any information collected at any monitor can be shared with other monitors.
reliably and at very low delays. The sensor network, on the other hand, is composed of an unknown number of sensor nodes equipped with wireless communication interfaces. Sensors are located at unknown locations in an area observed by the anti-sensor network. Although limited in power resources, sensor nodes are assumed to be able to perform basic message encryption and perform other elementary computational operations.

Monitors detect sensor locations based on the communication activity of sensors. We assume that monitors can detect and receive signals from all sensor nodes in the observation area. The observations are converted to two basic estimations, namely, distance to signal source, and the Direction of Arrival (DoA). Monitors rely on received signal strength (RSS) to estimate the distance to the signal source [39, 47]. It is also assumed that monitors are equipped with antenna arrays used to estimate the

1Since sensor nodes do not cooperate with the monitors, distance estimations cannot be done using methods such as time difference of arrival.
DoA of signals [48]. These estimations inherently contain errors. Through collaborative processing of individual observations, the resulting error in location estimation is minimized. Throughout the paper, we assume that the distance and DoA estimation errors have zero means. An example network is depicted in Figure 3.1. In this figure, a rectangular area is protected by four monitors located at the corners of the area. Upon detecting signals emitted from a sensor node, monitors produce estimates about the distance to the signal source using received signal strength and the DoA of the signal using their antenna arrays.

### 3.3.2 Sensor Counter-Measures

To counter the detection efforts of monitors, sensors may launch counter-measures. In this work, we consider two simple yet effective counter-measures: Message encryption and location camouflaging through transmission power changes. The resulting scenarios are outlined in the following. The details of monitor actions under these scenarios are outlined in Section 3.4.

*Message Encryption:* Sensors try to avoid being detected by encrypting their messages. Message encryption prevents the monitors from identifying message sources directly as no explicitly ID can be extracted from the transmitted message. However, it is still possible to classify an encrypted message received by multiple monitors as belonging to the same source. For this purpose, we utilize the reception time stamps at the monitors and the encrypted bit sequence.

*Power Level Variation:* In addition to message encryption, sensors can also try to hide their location through transmission power variation. This additional defense on sensors’ side involves transmission of encrypted data packets at various transmission
powers. The direct consequence of this measure is that the received signal strength cannot be used to estimate the distance to the signal source. Hence, sensor location estimation needs to be performed only based on the DoA estimation.

### 3.4 Opportunistic Localization Algorithms

The localization of sensor nodes is a challenging task as sensors do not cooperate with the monitors. Under these adverse conditions, the already challenging sensor localization becomes even harder to accomplish. To this end, we propose to use localization algorithms augmented with signal classification. In this section, we outline the basic location estimation methodology used in our procedures and analyze achievable performance levels. Then we introduce counter-measure mitigation techniques to localize sensors.

#### 3.4.1 Basic Opportunistic Localization Methodology

In this section, we introduce a mathematical representation of the localization problem and achievable performance limits. First, we introduce a representation where multiple independent observations can be used for location estimation. Then, we show how the accuracy performance can be improved through the use of multiple observations with the same set of observers. Motivated by these observations, we then discuss how the number of observations can be increased under message encryption.

Let the anti-sensor system be comprised of $m$ monitors located at $(x_i, y_i), i = 1, \ldots, m$. Let each monitor observe the DoA and the received signal strength of a signal emitted by a sensor located at $(x, y)$. Under ideal conditions, each monitor $i$ should determine the direction of arrival (DoA) of the signal $\theta_i(x, y)$ and the distance
to the emitter sensor $r_i(x,y)$ with no error:

$$r_i(x,y) = \sqrt{(x-x_i)^2 + (y-y_i)^2} \quad (3.1)$$

$$\theta_i(x,y) = \arctan \frac{y-y_i}{x-x_i}, \quad i = 1, \cdots, m. \quad (3.2)$$

However, a monitor $i$ can only estimate the DoA $\hat{\theta}_i$ and the distance to the sensor $\hat{r}_i$ imprecisely due to measurement errors and errors introduced by the signal propagation. Let distance and DoA estimation errors be denoted by $\Delta r_i$ and $\Delta \theta_i$:

$$\Delta r_i = \hat{r}_i - \sqrt{(x-x_i)^2 + (y-y_i)^2} \quad (3.3)$$

$$\Delta \theta_i = \hat{\theta}_i - \arctan \frac{y-y_i}{x-x_i}, \quad i = 1, \cdots, m. \quad (3.4)$$

Let the error in location estimation be denoted as $\Psi = [\Delta x \Delta y]^T$. Using Taylor series expansion, we can obtain a linearized estimate for $\Delta r_i$ and $\Delta \theta_i$, $i = 1, \cdots, m$, expressed in terms of the error in position $\Psi = [\Delta x \Delta y]^T$.

$$\Delta r_i = \frac{\partial r_i(x,y)}{\partial x} \Delta x + \frac{\partial r_i(x,y)}{\partial y} \Delta y \quad (3.5)$$

$$\Delta \theta_i = \frac{\partial \theta_i(x,y)}{\partial x} \Delta x + \frac{\partial \theta_i(x,y)}{\partial y} \Delta y \quad (3.6)$$

Defining terms $a_i$ and $b_i$ for $i = 1, \cdots, m$ as

$$a_i = \frac{\partial r_i}{\partial x} = \frac{x-x_i}{r_i}, \quad a_{m+i} = \frac{\partial \theta_i}{\partial x} = \frac{-(y-y_i)}{r_i^2}, \quad (3.7)$$

$$b_i = \frac{\partial r_i}{\partial y} = \frac{y-y_i}{r_i}, \quad a_{m+i} = \frac{\partial \theta_i}{\partial y} = \frac{x-x_i}{r_i^2}, \quad (3.8)$$

and considering the observations of all monitors $i = 1, \cdots, m$, the resulting system of equations can be expressed as

$$\Phi = A \Psi, \quad (3.9)$$

where $\Phi = [\Delta r_1 \cdots \Delta r_m \Delta \theta_1 \cdots \Delta \theta_m]^T$, and
\[ A = \begin{bmatrix} a_1 & b_1 \\ \vdots & \vdots \\ a_{2m} & b_{2m} \end{bmatrix}. \]  

(3.10)

If estimation error of all monitors \( \Phi \) is known, then the estimation error in location \( \Psi \) can be computed as follows:

\[ \Psi = A^+ \Phi, \]  

(3.11)

where \( A^+ = (A^T A)^{-1} A^T \) is the pseudo inverse of \( A \).

With this introduction, we now show that multiple observations obtained from a given sensor by a fixed set of monitors is an effective way of reducing the localization uncertainty. To represent localization uncertainty, we use a covariance-based metric. The covariance matrix \( \Sigma_\Phi \) of the observation error vector \( \Phi \) is given by

\[ \Sigma_\Phi = \mathbb{E} \left[ (\Phi - \mathbb{E}[\Phi]) (\Phi - \mathbb{E}[\Phi])^T \right], \]  

(3.12)

where \( \mathbb{E}[\cdot] \) is the expected value operator. If the observation errors have zero mean, i.e., \( \mathbb{E}[\Phi] = 0_{2m \times 1} \), then \( \Sigma_\Phi \) can be written as \( \Sigma_\Phi = \mathbb{E}[\Phi \Phi^T] \). It is reasonable to assume that the measurements obtained from different monitors are independent of each other. Under the zero-mean distribution assumption of measurement errors, \( \Sigma_\Phi \) has non-zero elements only in the main diagonal, i.e., \( \Sigma_\Phi = \text{diag}(\sigma_1^2, \cdots, \sigma_m^2, \sigma_{m+1}^2, \cdots, \sigma_{2m}^2) \), where \( \sigma_i^2 \) is the variance of distance estimations at monitor \( i \), and \( \sigma_{m+i}^2 \) is the variance of the DoA estimation at monitor \( i, i = 1, \cdots, m \).

The covariance matrix \( \Sigma_\Psi \) of the location estimation error can be computed as follows:

\[ \Sigma_\Psi = \mathbb{E} [\Psi \Psi^T] = \mathbb{E} \left[ (A^+ \Phi) (A^+ \Phi)^T \right] = A^+ \mathbb{E} [\Phi \Phi^T] (A^+)^T = A^+ \Sigma_\Phi (A^+)^T \]  

(3.13)
The covariance matrix $\Sigma$ is a $2 \times 2$ matrix in the following form:

$$
\Sigma = \begin{bmatrix}
\sigma^2_{\Delta x} & \sigma_{\Delta x \Delta y} \\
\sigma_{\Delta x \Delta y} & \sigma^2_{\Delta y}
\end{bmatrix}.
$$

(3.14)

We use the trace $\text{tr} (\Sigma) = \sigma^2_{\Delta x} + \sigma^2_{\Delta y}$ as a measure of the position estimation uncertainty. Let the sensor deployment area $A$ and the locations of $m$ monitors be given. The maximum uncertainty for the deployment area $A$ is computed as

$$
\text{Tr}_A = \max_{(x,y) \in A} \text{tr} (\Sigma).
$$

(3.15)

$\text{Tr}_A$ represents the worst case uncertainty obtained for a given anti-sensor network for the observation area $A$. This uncertainty can be reduced by adding more resources to the estimation (i.e., employing more monitors) or reducing the distance and DoA estimations with higher precision devices. If these options are not viable, an alternative approach is to increase the number of observations used to calculate the sensor locations. The following theorem shows the relationship of multiple observations and the localization uncertainty.

**Theorem 2.** Let $\text{tr}(\Sigma)$ be used as the uncertainty measure of the sensor localization procedure. Given a fixed set of monitor positions, the uncertainty reduces to $\frac{1}{k} \text{tr}(\Sigma)$ if $k$ statistically independent observations are obtained from the same signal source and processed jointly.

**Proof.** The pseudo-inverse matrix $A^+$ can be written as follows:

$$
A^+ = \frac{1}{h} \begin{bmatrix}
c_1 & \cdots & c_{2m} \\
d_1 & \cdots & d_{2m}
\end{bmatrix}_{2 \times 2m},
$$

where

(3.16)
\[ h = \left( \sum_{i=1}^{2m} a_i^2 \right) \left( \sum_{i=1}^{2m} b_i^2 \right) - \left( \sum_{i=1}^{2m} a_i b_i \right)^2, \]  
(3.17)

\[ c_i = a_i \sum_{l=1}^{2m} b_l^2 - b_i \sum_{l=1}^{2m} a_l b_l, \]  
(3.18)

\[ d_i = b_i \sum_{l=1}^{2m} a_l^2 - a_i \sum_{l=1}^{2m} a_l b_l, \]  
(3.19)

and \( a_i \) and \( b_i \) are defined in Equations 3.7 and 3.8, respectively. Then, \( \Sigma_\Psi \) can be expressed as given in Eq. 3.21.

\[
\Sigma_\Psi = \frac{1}{h} \left[ \begin{array}{ccc}
& c_1 & \cdots & c_{2m} \\
d_1 & & & \\
& \vdots & \ddots & \vdots \\
d_{2m} & & & 
\end{array} \right] \left[ \begin{array}{ccc}
\sigma_1^2 & & \\
& \ddots & \ddots & \ddots \\
& & \ddots & \\
& & & \sigma_{2m}^2 
\end{array} \right] \frac{1}{h} \left[ \begin{array}{ccc}
c_1 & d_1 \\
& \vdots & \ddots & \ddots \\
& & \vdots & \\
c_{2m} & d_{2m} 
\end{array} \right] 
\]  
(3.20)

\[
\Sigma_\Psi = \frac{1}{h^2} \left[ \sum_{i=1}^{2m} \sigma_i^2 c_i^2 \sigma_i^2 c_i d_i \sum_{i=1}^{2m} \sigma_i^2 c_i d_i \right] 
\]  
(3.21)

Similarly, let \( A_k \) be defined as a \( 2km \times 2 \) transformation matrix formed by \( k \) observations by \( m \) monitors the same way as in Eq. 3.10:

\[
A_k^T = \left[ \begin{array}{ccc}
a_1' & \cdots & a_{2km}' \\
b_1' & \cdots & b_{2km}' 
\end{array} \right]. 
\]  
(3.22)

A monitor can obtain independent samples originating from the same source if the samples are taken sufficiently apart from each other in time. If a set of monitors receive \( k \) statistically independent signals from the same source, \( a'_{2jm+i} = a_i \) and \( b'_{2jm+i} = b_i \) for \( i = 1, \ldots, 2m \) and \( j = 1, \ldots, k-1 \). Hence, the resulting transformation matrix \( A_k \) has the form \( A_k = [A \mid \cdots \mid A]_{2 \times 2km}^T \), where \( A \) is the matrix given in Eq. 3.10. Consequently, the pseudo-inverse \( A_k^+ \) of the matrix \( A_k \) can be written as follows:

\[
A_k^+ = \frac{1}{h'} \left[ \begin{array}{ccc}
c_1' & \cdots & c_{2km}' \\
d_1' & \cdots & d_{2km}' 
\end{array} \right]. 
\]  
(3.23)
where $h' = k^2 h$, $c_{2jm+i}' = kc_i$, and $d_{2jm+i}' = kd_i$ for $i = 1, \ldots, 2m$ and $j = 1, \ldots, k-1$.

Substituting these equalities in Eq. 3.23, we obtain

$$A_k^+ = \frac{1}{kh} \begin{bmatrix} c_1 & \cdots & c_{2m} \\ d_1 & \cdots & d_{2m} \end{bmatrix}.$$ 

The resulting covariance matrix $\Sigma k \Psi$ is computed as

$$\Sigma k \Psi = \frac{1}{k^2 h^2} \left[ \frac{k}{k} \sum_{i=1}^{2m} \sigma_i^2 c_i^2 \right]$$

$$= \frac{1}{kh^2} \left[ \frac{\sum_{i=1}^{2m} \sigma_i^2 c_i^2}{\sum_{i=1}^{2m} \sigma_i^2 c_i d_i} \frac{\sum_{i=1}^{2m} \sigma_i^2 c_i d_i}{\sum_{i=1}^{2m} \sigma_i^2 d_i^2} \right] = \frac{1}{k} \Sigma \Psi.$$ 

Therefore, $\text{tr} (\Sigma k \Psi) = \frac{1}{k} \text{tr} (\Sigma \Psi)$ as claimed.

Theorem 2 presents the gain in uncertainty performance with each additional observation from the same source. If every received signal can be associated with its correct source, this performance improvement can be achieved with an unbiased localization algorithm. This observation motivates us to increase the number of samples to improve the localization uncertainty. However, in the adversarial sensor/anti-sensor network environment envisioned in this paper, sensors protect their identity through message encryption. While a one-time transmission received by multiple monitors can be correlated as outlined in Section 3.4.2, the same procedure cannot associate multiple transmissions with their actual sources. First, we propose to use a single packet transmission to obtain multiple observations. As the resulting estimates may not be sufficiently accurate, we augment this approach with clustering algorithms that aim to map unknown sample points to their respective sources. However, clustering does not provide error-free classification of sample points, which may lead to localization errors and higher uncertainty. In the following, we propose algorithms to improve the localization accuracy under encryption counter-measure.
3.4.2 Localization with Encrypted Messages

When sensors use encryption to hide their identities, monitors need to find alternative ways to associate received signals with their sources. To this end, we propose to use a single packet transmission to extract multiple samples. The resulting estimates are then used by two localization algorithms. In these localization algorithms, observing multiple transmissions from the deployment area, we cluster the individual location estimations as belonging to a set of sources. Then, the clustered observations are jointly processed to yield our final estimation for the signal sources.

Classification of a One-Time Transmission

Under encryption, sensor IDs cannot be extracted from transmitted information packets to connect transmissions with sources. However, monitors can associate a given packet transmission received by multiple monitors to the same source, although the identity of the source would remain unknown. Such transmissions are referred to as one-time transmissions. This association is performed using time stamps and signal signatures.

Let a sensor $SN$ at $(x, y)$ transmit an encrypted packet $p$ at time $t$. Let $p$ be received as $p_i$ a monitor $i, i = 1, \cdots, m$ and time-stamped $t_i$. Due to inaccuracies in clock synchronization, there can be a difference in the time stamps, i.e., $|t_i - t_j| \leq t_\epsilon$, $i \neq j, i, j = 1, \cdots, m$, where $t_\epsilon$ is the maximum clock synchronization difference. If the system operates at very low duty cycles, timestamps may be sufficient to conclude that a set of received packets $p_i$ belongs to the same source $SN$. However, at higher duty cycles, it is possible that $t_\epsilon$ is large enough to allow multiple encrypted messages to be received by a given monitor. In such cases, the encrypted bit sequences need
to be compared at a central location to identify whether or not they originate from the same source. Although costlier, this second approach yields a more accurate correlation between the observations.

**Using One-Time Transmissions for Location Estimation**

We define a *sample* as the set of $m$ distance and/or DoA estimations obtained by different monitors based on a one-time transmission, where $m$ is the number of monitors. When the estimation errors are not trivially small (which is the case in most systems), a location estimation based on a single sample is very inaccurate. However, the performance can be dramatically improved if multiple independent samples can be obtained from the same source as shown in Theorem 2. Since the source of a signal can be determined for a given packet, we can improve the localization performance by taking multiple observations during a single transmission.

Assuming that the transmission rates of sensor nodes is low and the packet transmission time is sufficiently long, it is possible to obtain multiple independent observations (samples) by each monitor during one packet transmission. We define a *sample set* $S_i$ consisting of $N_p$ samples $s^j_i$ gathered during $j^{th}$ packet transmissions, where $i = 1, \cdots, N_p$, denotes the sample number during the packet transmission. Each sample set $S_i$ consists of $m \times N_p$ distance and/or angle estimates. When $S_i$ processed by a location estimation algorithm (such as the non-linear least square (NLS) localization algorithm described in Figure 3.3), the resulting location estimation is referred to as *sample point* $SP_i$. 

65
Batch Localization Algorithm

The sample points may still contain inaccuracies. To increase the number of observations for localization, we need to group sample points according to their probable sources. The first localization algorithm we propose is a Batch Localization algorithm that processes the results of a large set of observations jointly. First, sample sets obtained from one-time transmissions are processed by the non-linear least square (NLS) localization algorithm, producing sample points. Then, sample points are clustered using the Quality Threshold (QT) clustering algorithm [49]. For each cluster of sample points, we re-estimate the final location of the sensor using the NLS algorithm with the sample sets of all sample points in the cluster. The outline of the algorithm is given in Figure 3.2. In our work, we use the maximum variance value computed
over the entire deployment area $\mathcal{A}$ as the cluster diameter limit $R$:

$$
R = \max_k \sqrt{\text{tr} (\Sigma_\Phi)} \tag{3.25}
$$

In this algorithm, $NLS(X)$ is the non-linear least square (NLS) [50] location estimation algorithm outlined in figure 3.3 applied on the sample observation set $X$. NLS algorithm is an interactive procedure to determine approximate solutions of over-determined or inadequately specified systems of non-linear equations. In NLS, the first location estimate can be obtained using only the observations of a single monitor or a combination of observations via multi-lateration or triangulation.

<table>
<thead>
<tr>
<th>Input: Sample Set: $X$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Output: Estimated Location: $EL$</td>
</tr>
<tr>
<td>$NLS(X)$:</td>
</tr>
<tr>
<td>1. Set counter $j:=0$</td>
</tr>
<tr>
<td>2. Compute an initial location estimate $(\hat{x}_0, \hat{y}_0)$</td>
</tr>
<tr>
<td>3. WHILE TRUE</td>
</tr>
<tr>
<td>4. FOR each monitor $i$</td>
</tr>
<tr>
<td>5. $x:=\hat{x}_j$, $y:=\hat{y}_j$ for Equations 3.5 and 3.6</td>
</tr>
<tr>
<td>6. Update $\Phi_j$:</td>
</tr>
<tr>
<td>7. Compute $\Delta r_i$ using Eq. 3.5</td>
</tr>
<tr>
<td>8. Compute $\Delta \theta_i$ using Eq. 3.6</td>
</tr>
<tr>
<td>9. Compute location error $\Psi_j$ using $\Phi_j$ and Eq. 3.11</td>
</tr>
<tr>
<td>10. IF $\Psi_j &gt; \epsilon$</td>
</tr>
<tr>
<td>11. $[\hat{x}<em>{j+1} ; \hat{y}</em>{j+1}]^T := [\hat{x}_j ; \hat{y}_j]^T + \Psi_j$</td>
</tr>
<tr>
<td>12. $j:=j+1$</td>
</tr>
<tr>
<td>13. ELSE</td>
</tr>
<tr>
<td>14. Return $EL:=(\hat{x}_j, \hat{y}_j)$</td>
</tr>
</tbody>
</table>

Figure 3.3: NLS Location Estimation Algorithm

In the batch localization algorithm, the clustering is done using a variation of the Quality Threshold (QT) [49] algorithm given in Figure 3.4. The minimum cluster
size is included in this procedure to eliminate the outliers. In our computations, we
typically use 1% of the number sample sets as the minimum cluster size.

Input: Sample Points: \( SP_i \); Cluster Diameter: \( R \);
Minimum Cluster Size: \( CS_{\text{min}} \)

Output: Number of Clusters: \( N \); Cluster Member Sets: \( C_i \)

\[ QT(SP_i, R): \]
1. Set Cluster Number \( N := 0 \)
2. WHILE \( S_i \neq \emptyset \)
3. Reset temporary cluster sets \( TC_j = \emptyset \)
4. FOR every sample point \( j \) in \( SP_i \)
5. Add all neighboring sample points within \( R \) to \( TC_j \)
6. Determine the largest temporary cluster \( TC_k \)
7. IF \( |TC_k| < CS_{\text{min}} \)
8. RETURN
9. Increment number of clusters: \( N := N + 1 \)
10. Store \( TC_k \) in \( C_N \)
11. Update \( SP_i \) by removing all sample points in \( C_N \):
12. \( SP_i := SP_i \setminus C_N \)

Figure 3.4: QT Algorithm

Hierarchical Localization Algorithm

Another method to localize nodes is to process the obtained sample sets in a hier-
archical manner. In this algorithm, the available sample set is divided into \( M \) groups.
After clustering these groups independently, the resulting estimates are bundled to-
gether to obtain another level of grouping. The benefit of the hierarchical algorithm
is not limited to reduced computational complexity: When sample set groups are
chosen smaller, the errors due to boundary violations are expected to be reduced,
as well. Our proposed hierarchical localization algorithm is outlined in Figure 3.5.
The function CENTROID(\( X \)) in this procedure computes the centroid of all locations

68
contained in the set $X$. Also note that we do not eliminate small groups since the
group sizes tend to be smaller when the sample point numbers are divided by $M$.

| Input: | $N$ Sample Sets: $S_i$; Cluster Diameter: $R$; |
| Number of Subgroups: $M$ |
| Output: | Estimated Number of Sensors: $EN$; |
| Estimated Locations: $EL_j$ |

$\textbf{H-Loc}(s_i,R,M)$:

1. FOR every sample set $S_i$, $i = 1, \cdots, N$
2. Compute sample point $SP_i := NLS(S_i)$
3. Divide $SP_i$ into $M$ groups $GSP_k$, $k = 1, \cdots, M$
4. FOR each group $GSP_k$, $k = 1, \cdots, M$
5. Cluster $GSP_k$ using QT algorithm:
6. Determine the number of clusters:
7. $EN_k := QT(GSP_k,R,0).N$
8. Determine sample point clusters:
9. $C_{kj} := QT(GSP_k,R,0).C_j$, $j = 1, \cdots, EN_k$
10. Determine all sample sets of samples points in $C_{kj}$:
11. $SP_i \in C_{kj} \Rightarrow S_i \in cs_{kj}$
12. Compute $CC_k := NLS(cs_{kj})$
13. Cluster $CC_k$ using QT algorithm:
14. Determine the number of clusters
15. $EN := QT(CC_k,R,0).N$
16. Determine sample point clusters
17. $C_j := QT(CC_k,R,0).C_j$, $j = 1, \cdots, EN$
18. FOR every cluster $C_j$, $j = 1, \cdots, EN$
19. Compute $EL_j := CENTROID(C_j)$

Figure 3.5: Hierarchical Localization Algorithm

3.4.3 Localization with Power Level Variations

In addition to message encryption, sensors can also change their transmission
power levels to affect monitors’ estimations. While the algorithms outlined in Sec-
tion 3.4.2 can be used without modification, the basis of the NLS estimation method
must be changed slightly since distance estimations are unreliable. The estimation error vector \( \Phi \) and the transformation matrix \( A \) are re-defined as \( \Phi' \) and \( A' \) by removing all distance estimation-related terms as shown below:

\[
\Phi' = \begin{bmatrix}
\Delta \theta_1 \\
\vdots \\
\Delta \theta_m
\end{bmatrix}, \quad \text{and} \quad A' = \begin{bmatrix}
a_{m+1} & b_{m+1} \\
\vdots & \vdots \\
a_{2m} & b_{2m}
\end{bmatrix}.
\] (3.26)

After this modification, the new covariance matrix \( \Sigma'_\Phi \) of \( \Phi' \) is given as

\[
\Sigma'_\Phi = \text{diag}(\sigma_{m+1}^2, \ldots, \sigma_{2m}^2),
\] (3.27)

and the covariance matrix of the location estimation error is updated as

\[
\Sigma'_\Psi = A'^+ \Sigma_\Psi(A'^+)^T,
\] (3.28)

where \( A'^+ \) is the pseudo-inverse of \( A' \).

### 3.5 Performance Evaluation

In this section, simulation results reflecting the performance of the proposed anti-sensor localization system are presented. We use the mean and standard variance of the localization error as well as false positive and negative alarm ratios as metrics. Unless otherwise stated, we consider a \( 100m \times 100m \) deployment area with 20 randomly placed sensors and four monitors located in the corners. We assume that all monitors have estimation errors uniformly distributed in the range of \([5, +5]m\) for \( \Delta x \) and \([-0.04, +0.04]\) rad for \( \Delta \theta \). Reported results reflect the average of 200 independent runs. We also assume that packet transmissions of sensors are long enough to be sampled up to five times to produce statistically independent observations. The performance evaluation of the proposed localization algorithms is performed for three cases. The Case 1 involves no encryption or transmission power variations and serves
as our baseline under which the best performance is achieved. Case 2 corresponds to the scenario where sensors encrypt their messages. Finally, in case 3, sensors both encrypt their messages and also change their power levels.

### 3.5.1 Sensitivity Analysis of Localization Algorithms

The sensitivity of B-Loc and H-Loc algorithms is evaluated by changing the number of samples processed to obtain the location information. In this simulation study, we consider $N_p = 4$ with 20 and 50 sensors; and $N_p = 5$ for 20 sensors, and change the number of samples. We observe the false positive alarm ratio (i.e., ratio of the number of nodes falsely identified to exist to the total number of sensors), negative false alarm ratio (i.e., the ratio of the number of sensor nodes not found to the total number of sensors), the mean error, and the standard variance of the error. In Figure 3.6, Figure 3.7, and Figure 3.8 the positive and negative false alarm ratios are plotted against the average number of samples per sensor. The average number of transmissions is determined by dividing the average number of samples per sensor by $N_p$.

In all figures, we can see that the false positive and negative alarm ratios are zero since the IDs are plainly visible to monitors and no clustering is performed. For $N_p = 4$ and 50 sensors, the false positive alarm ratio is very small (not larger than 0.1%) for all cases and localization algorithms as shown in Figure 3.6(a). However, this comes at the expense of increased false negative alarm ratio. When B-Loc algorithm is used, the ratio is as large as 30% for Case 3, which drops to almost half when H-Loc algorithm is used instead, as shown in Figure 3.6(b).
Figure 3.6: Sensitivity Analysis of B-Loc and H-Loc Algorithms – False Alarms – \( N_p = 4, 50 \) Sensors

(a) \( N_p = 4, 50 \) Sensors, False Positive

(b) \( N_p = 4, 50 \) Sensors, False Negative
Figure 3.7: Sensitivity Analysis of B-Loc and H-Loc Algorithms – False Alarms – $N_p = 4, 20$ Sensors

(a) $N_p = 4, 20$ Sensors, False Positive

(b) $N_p = 4, 20$ Sensors, False Negative
Figure 3.8: Sensitivity Analysis of B-Loc and H-Loc Algorithms – False Alarms – $N_p = 5$, 20 Sensors
With $N_p = 4$, when the number of sensors is dropped to 20, the false positive alarm ratio increases to almost 3%, which tapers off quickly when more samples are obtained for Case 2 and 3 and both localization algorithms (Figure 3.7(a)). In general, B-Loc algorithm results in higher false positive alarm ratios. As shown in Figure 3.7(b), Case 2 has a much smaller false negative alarm ratio, than for Case 3 as expected when compared for each algorithm. For both cases, H-Loc algorithm results in a lower false negative alarm ratio in an absolute sense. Considering only the false alarm ratios, we observe a trade-off between algorithms: B-Loc has a better false positive alarm performance and H-Loc has a better false negative alarm performance. Note that both algorithms perform almost the same way when $N_p$ is increased to 5 for 20 sensors. Under this scenario, Case 2 also has a better performance than Case 3, which is expected since the amount of information contained in Case 2 is larger than in Case 3. This latter scenario is depicted in Figures 3.8(a) and 3.8(b).

We also observe the mean and the standard variance of the error for the same scenarios as shown in Figure 3.9, 3.10 and Figure 3.11. In these Figures, we can observe that the means are better estimated with B-Loc algorithms than with H-Loc algorithm for all three scenarios and two cases as shown in Figures 3.9(a), 3.10(a), and 3.11(a). When the number of samples is increase, the error mens stabilize, indicating that additional observations do not improve the average error performance. The standard variance, on the other hand, is minimized for 20 samples for both scenarios with 20 sensors when B-Loc algorithm is used. The standard variance performance of the H-Loc algorithm is consistently worse than the B-Loc algorithm. This also brings out another trade-off: Although H-Loc algorithm reduces the number of negative false alarms, it performs poorly with respect to the localization accuracy.
Figure 3.9: Sensitivity Analysis of B-Loc and H-Loc Algorithms – Detection Performance – \( N_p = 4, 50 \) Sensors
Figure 3.10: Sensitivity Analysis of B-Loc and H-Loc Algorithms – Detection Performance – $N_p = 4, 20$ Sensors

(a) $N_p = 4, 20$ Sensors, Mean

(b) $N_p = 4, 20$ Sensors, Standard Variance
Figure 3.11: Sensitivity Analysis of B-Loc and H-Loc Algorithms – Detection Performance – $N_p = 5, 20$ Sensors

(a) $N_p = 5, 20$ Sensors, Mean

(b) $N_p = 5, 20$ Sensors, Standard Variance
3.5.2 Effect of Number of Monitors and Signal Samples

We have observed the effect of the number of monitors (equally spaced along the perimeter) and the number of samples on the estimation error mean and the standard variance for $N_p = 5$ as shown in Figure 3.12. In these simulations, the H-Loc and B-Loc algorithms are found to have very little difference in both performance metrics. Hence, only B-Loc has been shown in Figure 3.12. As expected, as the number of samples per sensor and the number of monitors increase, the mean error decreases for all cases with diminishing returns. For standard variance, the performance improvement is negligible for Cases 1 and 2 for high number of samples or monitors. In Case 3, the performance becomes slightly worse for very high number of samples due to misclassifications during clustering. This classification error can be explained by the cumulative effect of higher number of insufficient samples distorting the performance of the clustering efficiency of the localization algorithm. Hence, it can be overall stated that the performance behavior of the proposed localization algorithms depend on the initial accuracy of the sample sets. Therefore, it may not be possible to observe the benefits of an always increasing accuracy with higher number of samples.

3.5.3 Effect of Estimation Errors in Monitors

In this experiment, we observe the effect of estimation error ranges on the mean and the standard variance of the localization error for $N_p = 5$. We change the distance estimation error range from $\pm 1$ to $\pm 10m$ and DoA estimation range from $\pm 0.01$ to $\pm 0.1rad$. The results for Case 2 are presented in Figure 3.13. In both figures, we can observe that the B-Loc and H-Loc algorithms once again have very similar performance. In Figure 3.13(a), the mean error distance increases for increasing
ranges of DoA and distance error ranges. The mean error distance increases for Case 1, as well, albeit at a very small rate. It should be noted that this increase for Case 1 is caused by finite (and insufficient) number of samples that do not produce a sample mean of zero as they should. The standard variance error of Figure 3.13(b) also shows a very similar trend for both Case 1 and Case 2.

The same simulation study is also repeated for Case 3. In this case, the changes in distance error range is irrelevant as they are not considered in the estimation procedure. The results are not graphically presented. Our results show that the localization error mean and standard variance increase as the DoA estimation error range increases. We also observe that the selection of H-Loc or B-Loc does not affect the performance significantly in this case, either.

3.6 Summary

In this chapter, the opportunistic localization system is proposed that aims to protect a given asset from being observed by a sensor network. The system relies on fixed set of monitor nodes to observe wireless transmissions in the protected area. With these observations, sensor locations are estimated through collective processing. Once localized, sensors are physically removed from the observation area. To accomplish this, we also develop new opportunistic localization methods to be used in non-cooperative environments. Our simulation results also support our analytical findings and design principles: Longer observation times improve the localization performance and produce accurate location estimations with diminishing returns.
Figure 3.12: Error Mean and Standard Variance vs. Number of Samples and Monitors
Figure 3.13: Error Mean and Standard Variance vs. Error Ranges for Case 2
CHAPTER 4

OPPORTUNISTIC ENCRYPTION IN WIRELESS LOCAL NETWORKS

The current 802.11i standard can provide data confidentiality, integrity and mutual authentication in enterprise Wireless Local Area Networks (WLANs). However, secure communication can only be provided after successful authentication and a robust security network association is established. In general, the wireless link layer is not protected by the current standard in WLANs, which leads to many possible attacks, especially in public open-access wireless networks. We argue that regardless of the type of network under consideration, link-layer protection and data confidentiality are of great importance in wireless applications. In this chapter, we first identify and analyze the security issues ignored by the current 802.11 security standard. Then we propose our solution to patch the current 802.11i standard and address all those issues with opportunistic encryption. One of the key aspects of opportunistic encryption is how to set up an encryption key. We propose a new opportunistic encryption key-establishment algorithm. Since it takes the place of the open authentication algorithm in 802.11 MAC protocol, we call it dummy authentication algorithm. Dummy means no real authentication for a user. In the key-establishment algorithm, we apply public-key cryptography’s key-establishment technique to the 802.11 MAC protocol.
4.1 Motivation

Wireless networks offer organizations and users great benefits such as flexibility, portability and low installation cost. Commercial Wireless Local Area Network (WLAN) products are widely available on the market and they facilitate easy setup and convenient access. Currently, most popular commercial wireless network interface cards and wireless routers support the following security options: disabled, Wired Equivalent Privacy (WEP), Wi-Fi Protected Access-Enterprise (WPA-Enterprise) and WPA-Personal (also known as WPA-Pre-Shared Key or WPA-PSK).

When security is disabled, the network is an open-access network that is open to everyone and requires no user authentication. Due to their simplicity and low management costs, open-access networks are widely used. However, they provide no protection whatsoever and are vulnerable to passive eavesdropping, traffic analysis and user fingerprinting. They are also vulnerable to active attacks such as wireless packet injection. It is very easy to inject malicious code in HTTP traffic or false DNS or DHCP responses for attack purposes such as phishing. We argue that open-access wireless networks should be open only to access, not to attack, and they should also provide data confidentiality.

WEP is used to provide secure data communications over wireless links. However, numerous researchers have shown that there are significant deficiencies in both the authentication and the encryption mechanisms [51–56]. Recent research shows that an active attack on the WEP protocol can recover a 104-bit WEP key in less than 60 seconds [55].
In WPA-PSK networks, the EAP authentication is reduced to a simple pre-shared common password instead of user-specific credentials. As long as a user provides the correct password for the network, he will be granted access to a WLAN.

WPA-Enterprise networks use the Extensible Authentication Protocol (EAP) to authenticate a user before he is allowed access to a network. Previous research in 802.11i (also known as WPA2) [57, 58] has shown that it is a well-designed standard for data confidentiality, integrity and mutual authentication in enterprise networks. However, encrypted data communications can be provided only after a Robust Security Network Association (RSNA) is established. Before an RSNA is established, all wireless messages are sent in the clear. Because the four-way handshake is not protected, i.e., encrypted, the messages can be easily captured. This leads to vulnerabilities in PSK in practice. WPA-PSK with a weak key is subject to dictionary attacks. Although this is not a design flaw of the standard, in reality, most people use a simple password as a pre-shared passphrase. WPA-PSK is also vulnerable to insider attacks, passive eavesdropping, or active attacks. Although each user can have a different passphrase in theory, current products only allow one passphrase to be used by all users.

In general, wireless link frames are not protected. This includes management frames in all four types of networks, both null and QoS null data frames in these networks, and data frames that transfer 802.1x/EAP authentication messages in WPA-Enterprise networks. Without link-layer protection, many attacks can be launched by forging these frames, e.g., Denial-of-Service (DoS) attacks using forged disassociation frames or EAP over LAN (EAPOL) logoff messages and power-saving attacks using forged null data frames.
In order to provide secure WLANs and counter possible attacks, *link-layer protection against spoofing and eavesdropping is needed*. If link-layer protection is provided, none of the aforementioned attacks can occur. For instance, in open-access networks, an attacker can no longer obtain cleartext messages or inject malicious code.

However, providing wireless link-layer protection in WLANs is a challenging task. First, millions of wireless networking products have been released to market and are in use. Any new solution should be compatible with the existing 802.11 standards and require as few modifications as possible. Second, the computational, memory usage and wireless channel utilization overhead should be low, since both the wireless Access Points (APs) and wireless network interface cards are resource-constrained devices. Finally, the solution should resist existing attacks and *not* introduce any new vulnerabilities.

To address the issues in the standard, we use opportunistic encryption to improve WLAN security. While opportunistic encryption has been existing for some time, the way that we set up encryption keys is new. We proposed a new *dummy authentication* key-establishment algorithm in this chapter. Here, *dummy* means *no real authentication for a user*. In dummy authentication key-establishment algorithm, we apply public-key cryptography’s key-establishment technique to the 802.11 MAC protocol. The only modifications to the 802.11i standard are a pairwise master key derivation function and the second stage of the RSNA establishment procedure, in which we replace the open-system authentication algorithm with our new dummy authentication key-establishment algorithm. We use the dummy authentication key-establishment algorithm to set up a session key that is in use for a short time. Then we use this
session key to derive a pairwise master key, which is also defined in the 802.11i standard and used to derive a set of encryption keys for link-layer data protection. Our method complements the standard; it is not a stand-alone protocol. It requires few modifications to the standard, yet it can greatly enhance the security of WLAN in a way that provides data confidentiality without user authentication. Our method is especially useful for public open-access WLAN environments and small-office and home networks.

4.2 Related Work

In this subsection we give a brief overview of related work.

4.2.1 WLAN Security

It is said that the most significant source of security risk in wireless networks is the use of radio waves to transmit data [59]. Radio waves move through the air and can be intercepted by any attacker with the appropriate equipment within a certain range.

The early IEEE 802.11 standard [60] provides two basic security mechanisms for secure access to WLAN networks. Entity authentication, including open-system and shared-key authentication, is used to authenticate the wireless devices. WEP is used to provide secure data communications over wireless links. However, numerous researchers have shown that there are significant deficiencies in both the authentication and the encryption mechanisms [51–56].

The last related IEEE standard, 802.11i [61], which was ratified in 2004, can overcome all existing security issues of WEP. Unlike WEP, 802.11i implements a set of functions called Robust Security Network Association that offers a greater security
level as well as authentication and integrity check functions [59]. WPA2-Personal uses a Pre-Shared Key. The authentication mechanism based on EAP/802.1x/RADIUS in WPA2-Enterprise can provide mutual authentication. New key-management and -establishment algorithms are introduced in the standards to manage security keys dynamically and automatically. Before a secure data transmission session starts, a fresh pairwise key and/or a group key must be generated by the handshake procedures. The Advanced Encryption Standard (AES) with the Counter Mode-Cipher Block Chaining (CBC)-Message Authentication Code (MAC) Protocol (CCMP) provides data confidentiality, data origin authentication, data integrity and replay protection for wireless frames.

Since this new standard is not compatible with exiting WEP wireless devices, Wi-Fi protected access (WPA) has been proposed as a temporary and transitional measure to repair the problems in WEP and offer backward compatibility. By using a keyed cryptographic Message Integrity Code (MIC), an extended Initialization Vector (IV) space and a key mixing function, the Temporal Key Integrity Protocol (TKIP) can provide stronger security in WPA.

4.2.2 Open-Access WLANs

Public wireless networks may still require user accounts and corresponding credentials for authorization, authentication and accounting purposes. In this chapter, we only focus on one aspect of public wireless networks, i.e. data link security in open-access wireless networks.

Open wireless networks should actually be called open-access wireless networks that have no access authorization and authentication. On the one hand, they should
not be totally open wireless networks, in which anyone can eavesdrop or tamper with somebody else’s wireless communication in addition to open access. On the other hand, they are not the Open Access Networks used in telecommunication systems.

Here, *open* means *open access*, i.e., no authentication. This is similar to anonymous access or guest access. All possible users who can access network may do so, perhaps with some restrictions.

In order to set up different encryption keys for each individual party that holds a PSK, Mano [62] proposes an enhanced four-way handshake, which is modified to incorporate the Diffie-Hellman key-exchange protocol. Since the four-way handshake is a required process that follows pairwise master key derivation, this modification affects not only Small Office and Home (SOHO) network but also enterprise wireless systems. Thus it is not desirable. Furthermore, the Diffie-Hellman key-exchange protocol is known to be vulnerable to the man-in-the-middle attack.

Microsoft’s WPS uses a very complex way to deal with public WLANs. A mobile station first uses a null username and no certification to set up a connection with a server by PEAP-TLS guest authentication. Then a new account is created after the customer configures his identification and payment information. Finally, the station reestablishes a secure connection with AP using PEAP with MSCHAPv2 [63]. This method requires an infrastructure consisting of several servers behind the AP and users’ interaction to set up a connection.

*Captive portal* is a web page-based technique that can be used to build open wireless hotspots. When an unauthenticated user wants to access the network, he must open a browser and request a webpage. He is then redirected to a special webpage through a secure HTTP connection. This webpage may require authentication, or
simply display an acceptable use policy and require him to agree to its terms before he can connect to the Internet and use the service. The authentication can be as simple as a username and a password, an OpenID or a certificate. In most cases, an authentication server is needed. If a user does not use HTTP, e.g., he checks emails or uses SSH to access a server, then the portal will block him. Access points operate in open system authentication mode, so the wireless link is unencrypted. Some captive-portal solutions can also work with WPA, but a user must set up an account before he can use his credentials to connect to the network.

4.2.3 Public Key-based Key Establishment

Symmetric-key cryptography requires that the communicating parties share a secret key. Shared secrets can be established between parties in various ways. Configuring the keys in advance, as in WPA-PSK, appears to be the simplest solution, but it is not scalable. Session key-establishment protocols can be used to establish a shared secret between two or more parties through the network on demand. Essentially, there are two methods to establish keys between parties: key transport and key agreement [64].

In the case of key-agreement protocols, all parties contribute a key share, and the parties compute the session key from these key shares. In the case of key-transport protocols, the session key is created by one of the protocol participants, and then it is transported to the intended receivers in a secure way. Other popular key agreement protocols include Secure Remote Password Protocol (SRP) and Menezes-Qu-Vanstone (MQV) protocol.
The most famous key-agreement protocol is the Diffie-Hellman (DH) protocol \cite{65}, which is elegantly simple. Its security is based on the difficulty of computing discrete logarithms. But it does not provide key authentication at all, which leads to MITM attacks. Some variants of the DH protocol can resist these attacks. However, these variants either require a secret shared between communication parties or require a PKI so that a peer can check the other peer’s signature. We did not choose the DH protocol because it is unsuitable for direct use in the MAC protocol. There are several disadvantages. First, generating temporary public/private keys for each session is computationally expensive. Second, man-in-the-middle attacks are possible. Finally, both stations and APs need public/private keys or certificates, which are unnecessary for low-security systems and duplicate functionality in high-security systems such as some enterprise WLANs.

Besides symmetric-key schemes, public-key cryptography can be used to transfer a session key or a key share in a secured way. \cite{64} classifies major key agreement schemes well. The SSL/TLS protocol and SSH protocol are salient examples that use public-key cryptography-based key-agreement protocols.

### 4.2.4 Management Frame Protection

In \cite{66}, Ge and Sampalli present a per-frame authentication scheme to authenticate and protect some 802.11 management frames. With this scheme, every unicast frame received by the wireless station or AP after WEP authentication is first authenticated and then the corresponding management function carried out. This work is based on WEP and no confidentiality is provided.
IEEE 802.11w is a proposed amendment to the IEEE 802.11 standard to increase the security of its management frames. A new Broadcast/Multicast Integrity Protocol (BIP) is proposed in 802.11w to protect against forgery of broadcast/multicast management frames using an Integrity Group Temporal Key (IGTK), which is generated on the AP and delivered to each station as part of the four-way handshake and/or group handshake [67, 68]. The 802.11i CCMP data frame protection scheme is extended to provide data confidentiality, replay protection, and data origin authenticity for robust unicast management frames.

However, the security offered by 802.11w can only take place after EAP authentication and the four-way handshake. Frames that are sent before authentication are not protected. There is still no protection for management frames sent before EAP authentication, and no protection for data frames sent during EAP authentication. There is no intention to protect null data frames even after the four-way handshake. Most importantly, the proposed amendment neglects open network security.

4.3 Security Issues in 802.11i

In this section, we identify and analyze the security issues ignored by the current 802.11i security standard.

4.3.1 No Protection in Open WLANs

Wireless LAN (WLAN) access points are not only deployed by organizations as a part of enterprise wireless networks, but they are also installed in public “hotspots” such as airports, hotels, conference rooms, shopping malls, and Internet cafés for public Internet access. Public WLANs—especially free Wi-Fi hotspots—are widely used worldwide and there is no sign of decreasing deployment.
Two business models are possible for a public WLAN at a hotspot: free access and paid access. No matter which business model is used, the problem is that, at present, most public Wi-Fi hotspots use open-access networks without data encryption.

It is well known that open-access wireless networks have many security issues; they are subject to numerous attacks. Anyone with a wireless device such as a PDA or laptop can sniff or monitor the air traffic. Session hijacking and traffic injection and modification can be easily performed. Several tools that do so are publicly available and a miscreant can launch an attack without any knowledge of network protocols. Examination of existing wireless network attacks shows that the lack of link-layer data encryption is the root of open-access system vulnerabilities.

One way to solve this problem is to utilize the existing 802.11i standard, deploy an authentication server (AS) behind APs, and set up a user account for each possible customer. However, much work is necessary for deployment and chaos can easily result if the network is used for a big conference or in an airport that serves thousands of customers every day.

Actually, Microsoft’s Wireless Provisioning Services (WPS) are designed to simplify, automate, and standardize initial sign-up and subscription renewal so that users do not have to perform different sets of steps for each wireless provider to which they want to connect [63]. However, WPS requires several servers, including an Internet Information Services (IIS) provisioning server, a Remote Authentication Dial-In User Service (RADIUS) server configured with a certificate issued by a certification authority (CA), a Dynamic Host Configuration Protocol (DHCP) server to provide wireless customers with IP addresses, a database server to hold a promotion code database, and, finally, an Active Directory or a Lightweight Directory Access Protocol (LDAP)
database to store customers’ account information. Clearly, service providers’ investment, management, and maintenance costs would be exorbitant.

Small businesses are justifiably reluctant to accept these burdens, not to mention setting up an AS that may cost a thousand dollars, or even the whole WPS infrastructure that requires several servers. In a paid-access business model, the service provider may have an incentive to set up an AS. However, for most small businesses, free access is an advertisement method to attract more customers. They just want to buy a wireless router, connect it to the Internet, turn it on, quickly set it up, and use it. Even for citywide WLANs, one requirement is the provision of free Internet access to every customer with the least possible management.

Another way to provide secure communication in public WLANs is through a Virtual Private Network (VPN), which creates an encrypted communication “tunnel” from the mobile station to a trusted VPN host. The problem is that not everyone has a VPN resource from which he can connect and open wireless networks provide a new platform for miscreants to attack the VPN itself.

4.3.2 Weak Protection of WPA-PSK

WLANs are not only deployed in enterprise networks and public hotspots: many individuals install them in their homes as a convenient extension to wired LANs. According to a recent report by Synergy Research Group, the consumer market represents over half the total WLAN market in recent years.

WPA requires an authentication server, such as RADIUS, to prevent unauthorized users from accessing the network. In situations where an AS is not feasible, such as home and small-business networks, a simpler option also exists: Pre-Shared Keys
(PSKs). Later used as a Pairwise Master Key (PMK), a PSK is a 256-bit value generated by combining the WLAN’s name, or *Service Set Identifier* (SSID), with an 8- to 63-byte passphrase via a predefined function. While the 802.11i standard allows unique keys to be created for each user, in practice, a single key is used for all users. Known as WPA-PSK, this approach authenticates everyone with the same secret passphrase who is connected to the access point (AP).

There is no design flaw in WPA-PSK or the 802.11i standard. But the way it is used and implemented make it vulnerable. People are accustomed to choosing short passphrases as their shared secrets. Wireless products provide no way for each user to choose a different passphrase.

**Vulnerability of WPA-PSK to insider attacks**

By *insider*, we mean anyone who obtains the key through legitimate channels. He may be a staff member in a small business or a tenant in a rented house. Although he is authorized to access the wireless network, he is *not* authorized to sniff others’ communications across the network.

An insider can observe the four-way handshakes in the Pairwise Transient Key (PTK) establishment process. This four-way handshake occurs whenever a station (STA) connects to a WLAN using WPA or WPA2. It also occurs periodically thereafter whenever the AP refreshes transient keys.

Unfortunately, the way in which WPA/WPA2 encryption keys are generated and delivered makes it easy for an outside attacker to try to guess the PSK via a dictionary attack. Once an outsider has the PSK, he can steal service or decrypt data sent by legitimate users on the network. WPA-PSK has no means to prohibit a malicious insider from decrypting the data.
As shown in Figure 4.1, all the information needed to generate the PTK can be obtained from the first two messages. This includes the nonces exchanged between the AP and station, and two MAC addresses. An insider can calculate the PMK from the passphrase and therefore has all necessary information to calculate future PTKs using the following known functions.

\[
PMK = PSK = PBKDF2(passphrase, ssid, ssidLength, 4096, 256) \quad (4.1)
\]

\[
PTK = PRF - X(PMK, “Pairwisekeyexpansion”||
min(AA, SA)||max(AA, SA)||
min(ANonce, SNonce)||max(ANonce, SNonce)), \quad (4.2)
\]
where PBKDF$^2$ (Password-Based Key Derivation Function) is a key derivation function defined in RSA Laboratories’ Public-Key Cryptography Standards (PKCS) series. PRF-$X$ is a Pseudo-Random Function based on a keyed-Hash Message Authentication Code using SHA-1 (HMAC-SHA1) that generates a PTK of size $X$ bits, as defined in 802.11i section 8.5.1.1.

**Vulnerability of WPA-PSK with a weak key**

An outside attacker can try to guess the PSK by capturing and analyzing the four-way handshake messages. Recording handshake messages from an active WLAN is easily accomplished with a wireless capture program like Wireshark, Kismet or Airodump. Those who are impatient can use a frame injector like Airjack [69] or Airoreplay [70] to force legitimate users to reconnect.

From the captured data, a four-way handshake for the named WLAN can be extracted, including all of the values of interest from that handshake: the AP and station’s MAC addresses, the $SNonce$ and $ANonce$ values, and the fourth message’s payload and Message Integrity Code (MIC). Then the attacker examines the supplied dictionary file, trying words as possible passphrases to find the right PSK, which is the value that, when used as a PMK with all of these observed values, generates a matching MIC. This procedure can be automated by programs like aircrack-ng [70], KisMAC [71], and coWPAtty [72].

To make things worse, an attacker can use pre-computed password hashes like rainbow tables [73] to speed up cracking. For example, a 2006 ShmooCon demo showed coWPAtty testing 18,000 passphrases per second using a pre-hashed WPA PSK lookup table. One of the pre-hashed WPA PSK lookup tables posted online represents 170,000 words hashed against the top 1000 most common SSIDs.
The 802.11i standard suggests that in WPA-PSK, a pre-shared key should be very strong. To enforce that, a passphrase should contain at least 20 characters. The vulnerability of WPA-PSK with a weak key to dictionary attacks is not a design flaw of the standard. Still, in reality, most people use a simple password as a pre-shared secret.

4.3.3 No Protection of Wireless Frames

In general, wireless frames are not protected. Even after a Robust Security Network Association (RSNA) is established, certain data frames are still not encrypted.

No Management Frame Authentication & Encryption

Management frames in WLANs are not protected by the current standard. Unlike data traffic, which is encrypted with protocols such as the Temporal Key Identity Protocol (TKIP) or the Advanced Encryption Standard (AES) with Counter Mode-Cipher Block Chaining (CBC)-Message Authentication Code (MAC) Protocol (AES-CCMP), 802.11 management frames are always sent in an unsecured manner. They are always unauthenticated and unencrypted, even when the very highest levels of WLAN security are used. By exploiting these unprotected frames, denial-of-service, impersonation and modification attacks can be launched. An attacker can either flood the access point (AP) with management frames in order to exhaust its resources or send faked management frames, e.g., de-authentication and disassociation frames, to break existing connections. The de-authentication- or disassociation-frame DoS attack can be launched against either the client or the AP by sending forged de-authentication or disassociation frames. These attacks can completely bring down an existing connection and force the station to reestablish a connection. Based on these
two types of attacks, a series of attacks can be launched, including WEP attacks, WPA-PSK offline dictionary attacks and man-in-the-middle (MITM) attacks.

No Null Data Frame Authentication & Encryption

Two special data frames are null data frames and QoS null data frames. Because there is no payload data in the frames, they cannot be protected by the encryption and message authentication schemes that the 802.11i standard provides. However, because they are short and lightweight, they have several uses. For example, in power saving mode, stations use null data frames to indicate whether they are “asleep” or “awake.” During active scanning, stations use null data frames to indicate “asleep” before scanning other channels and “awake” after coming back to the original channel. An attacker can take advantage of these uses and send forged null data frames to steal buffered messages in the AP. Our analysis and experiment results show that null data-based power-saving-mode attacks can achieve greater performance than faked power-saving-poll frame attacks [74].

Null data frames can also be used in virtual jamming attacks, in which an attacker fills a very large Net Allocation Vector (NAV) duration field in the frames to trick honest stations into thinking that the channel is busy.

No Data Frame Authentication & Encryption for EAPOL Messages

Figure 4.2 gives a general overview of the message flow in the EAP/802.1x authentication process among a station (STA), AP, and authentication server (AS). The data frames sent during this process are not protected by the 802.11i standard since an RSNA has not been established at this stage. These frames are used to transmit
EAPOL (EAP over LAN) messages, and by exploiting them, denial-of-service and user-fingerprinting attacks can be launched.

An attacker can either eavesdrop on the traffic to gather sensitive information such as usernames or accounts or launch DoS attacks. Possible EAPOL-related DoS attacks include but are not limited to the following:

- Faked EAPOL-logoff attacks, where an attacker sends forged EAPOL logoff frames to the AP to force a legitimate station out of service;

- Faked EAP-failure attacks, where an attacker spoofs EAP failure frames to disconnect an authenticating session;

- Premature EAP-success attacks, where a rogue AP sends premature EAP success frames to a wireless station and forces it to drop an authenticating session.

In addition to the above issues, the first message of the four-way handshake does not use the MIC field, which means integrity and authenticity cannot be guaranteed. Thus by flooding forged initial messages, an attacker can force the AP and the station to produce inconsistent keys [51, 58, 59].

4.3.4 A Null Data Frame based DoS Attack Example

In Fig. 4.3, we show the general IEEE 802.11 data frame format. Null data frames are a special type of data frame where the Frame Body field is empty. The Frame Control field is also shown in Fig. 4.3, where the Protocol field denotes the version of 802.11 MAC. The Type and Subtype fields denote the type of the frame. The ToDS and FromDS fields denote whether the frame is sent to or from the distributed system respectively, and are set to 1 and 0 respectively in null data frames. The MoreFrag
field denotes whether there is more fragment in the current packet, which is 0 in null data frames. The Retry field denotes whether the frame is a retransmitted one. The PwrMgmt field denotes the power saving state. The station sets this bit to 1 if it plans to switch to sleeping state after this frame, while it sets this bit to 0 if it decides to switch to or stay in active mode. The MoreData field is used to denote whether there are more buffered data. The ProtectedFrame field denotes whether encryption is applied to the frame body. Since null data frame has no frame body, this bit is set to 0. The Order field denotes whether strict frame ordering is applied.

**Attacking Algorithm based on Null Data Frame Spoofing**

The basic idea of null data frame based Denial-of-Service (DoS) attacks is that the attacker spoofs wireless frames to mess up with the intended functionalities of the
genuine frames. Here we take power management as example to study how to launch a DoS by forging null data frame. We know that the AP will buffer incoming packets for a station which is in power saving mode, or sleeping state. By spoofing a victim station in sleeping state, an attacker can fetch the buffered packets at the access point and cause continuous frame losses for the victim station. An intuitive attack is that the attacker keeps flooding fake null data frames with $PwrMgmt$ field set to 0. This flooding based attack is simple to launch, but it involves a large amount of frame injections. This is not cost effective to the attacker, and results in easy detection. We design an attack that injects much fewer frames while achieving the same effect. The pseudocode of the attack is given in Algorithm 2. Specifically, the attacker captures and checks all frames related with the victim station. When the victim station sends a frame informing its intent to switch to sleeping state (lines 4 to 5), the attacker generates one fake null data frame (line 6). The attacker needs to set the MAC addresses of the sender and receiver appropriately (lines 7 to 8), and set the sequence number appropriately (line 9) to prevent immediate detection by the access point via sequence number inconsistency. Then the attacker sets the $PwrMgmt$ field to 0 (line 10) and sends the fake null data frame to access point (line
Algorithm 2 Null Data Frame based DoS Attack

1: Null_Data_Frame_based_DoS_Attack
2: while (TRUE) 
3:   capture a frame \( F \);
4:   if \( F\).sender_MAC == victim_MAC then 
5:     if \( F\).PwrMgmt == 1 then 
6:       generate fake null data frame \( F' \);
7:       \( F'\).sender_MAC == victim_MAC;
8:       \( F'\).receiver_MAC == AccessPoint_MAC;
9:       \( F'\).sequence_number == F.sequence_number + 1;
10:      \( F'\).PwrMgmt == 0;
11:     send fake null data frame \( F' \);
12:     victim_is_slepping == TRUE;
13:   else 
14:     victim_is_slepping == FALSE;
15:   end if 
16:   else if \( F\).receiver_MAC == victim_MAC AND \( F\).type == DATA AND victim_is_slepping == TRUE then 
17:     send fake ACK frame to access point;
18:   end if 
19: end while

11). The attacker also sets a variable \( victim\_is\_sleeping \) to TRUE (line 12) to track the state of the victim station. If the victim station sends a frame informing its intent to switch to active state (line 13), the attacker does nothing except for setting the variable \( victim\_is\_sleeping \) to FALSE (line 14). During the attack, when the access point sends a data frame to the victim station that is in sleeping state (line 16), the attacker sends a fake ACK frame back (line 17).

Experimental Results

In order to determine the feasibility of the above attack and its effectiveness, we implemented the algorithm using C program in linux and conducted extensive experiments on our testbed. Our first set of experiments is to test user experiences when
performing common network activities such as web surfing and file downloading under attack. In our experiments, two types of stations are used with four representative access points, that are, the access point in our campus (Aruba), the access point in our department (Cisco), a commercial wireless router (Netgear) and the software access point installed by us (MadWiFi [75]). We found that under attack, the communications become stalled in all cases. Users cannot open new website and the existing file downloading process stops. This illustrates the effectiveness of the attack.

To further evaluate the quantitative impact of the attack on different applications, we use the Iperf [76] as traffic generator and performance measurement tool. The server is installed with Iperf to generate TCP/UDP traffic, while the stations are installed with Iperf to receive traffic and measure throughput. Each test lasts for 300 seconds. Initially, there is no attack in place. The attack is enabled at the 60th second, lasts for 60 seconds, and is disabled at the 120th second. Again, the attack is enabled at the 180th second, lasts for 60 seconds, and is disabled at the 240th second.

In Fig. 4.4, we illustrate the impact of such attack from the real experiments. TCP and UDP traffic are tested on Intel NIC with power saving mode enabled and MADWiFi access point respectively. We can see in Fig. 4.4(a) that TCP throughput decreases to 0 immediately after attack comes at the 60th second. This is because the attacker keeps deleting the data from the access point. Even worse, the TCP connection is disconnected during the attack, and the throughput remains 0 after attack is disabled at the 120th second. This shows that consistent frame loss caused by the attack could cause TCP disconnection. In Fig. 4.4(b), we can see that UDP throughput degrades significantly during the time the attack is enabled. This is also because of the attacker deleting frames from the access point. However, UDP
traffic is able to resume its normal throughput when the attack is disabled due to the connectionless nature of UDP.

4.4 Wireless Frame Protection via Opportunistic Encryption

The key of the opportunistic encryption is how to take opportunities to set up a session key. We propose a new opportunistic encryption key establishment algorithm to replace the open-system authentication algorithm defined in 802.11 protocols. Since the algorithm does not really authenticate a user, but it takes the place of an authentication algorithm, we call it “dummy authentication” key establishment algorithm. Next we smoothly apply the public-key cryptography based key-establishment technique to the 802.11 MAC protocol in the dummy authentication. The only modifications to the 802.11i standard are a pairwise master key derivation.
function and the second stage of the RSNA establishment procedure, in which we replace the open-system authentication algorithm with our new dummy authentication key-establishment algorithm.

In order for a station to use opportunistic encryption, an AP should indicate that it supports dummy authentication in its beacon frames. Note that an AP also supports open-system authentication as well to provide backward compatibility. Thus, an encrypted link can be set up if a station supports dummy authentication. If a station does not support dummy authentication, then the original open-system authentication should be used. This is exactly what an opportunistic encryption does. In this subsection, the dummy authentication algorithm is introduced first. Then we describe the resulting RSNA establishment procedure. Finally, we explain how to use the common session key obtained from dummy authentication to derive the pairwise master key.

4.4.1 Opportunistic Encryption Key Establishment Algorithm

We assume that each AP has a pair of public-private keys, denoted as \( pk \) and \( sk \), e.g., RSA keys. The public key is contained in a CA-signed certificate or a self-signed certificate. We also assume that each AP has two secret symmetric keys \( k \) and \( k_{\text{pre}} \) that the AP updates regularly.

The dummy authentication key-establishment algorithm is as follows (also shown in Fig. 4.5):

1. A station sends a request \( (\text{ids}) \) to AP, where \( \text{ids} \) is the station’s MAC address.

   In the wireless frame, the authentication algorithm number is set to 2, which
indicates the new dummy authentication. The Authentication Transaction Sequence Number (ATSN) is set to 1.

2. The AP creates a ticket $ticket := (ids, ts, l, hmac)$. Here $ts$ is a timestamp from the AP’s local clock, $l$ defines a validity period for the ticket and $hmac := HMAC(ids, ts, l)$ is the keyed hash of $(ids, ts, l)$ using AP’s secret key $k$. The AP sends $(ida, code, ticket, cert)$ to the station. Here $ida$ is the AP’s MAC address, $cert$ is the AP’s certificate and $code$ is a status code that indicates success or rejection. As before, an AP can reject a client due to MAC address access-control-list violation or for other reasons. The ATSN is set to 2.

3. Optionally, the station verifies the AP’s certificate. If it is valid, then the station stores the ticket along with the AP’s public key. The station generates a random number $rnd$ and a pre-session key $psk$ and encrypts them with the AP’s public key $pk$. It sends $(ids, ticket, rnd, E_{pk}(rnd || psk))$ to the AP. The ATSN is set to 3. The station now can derive the common session key $csk$ using a pseudo-random
function $PRF$ defined in section 8.5.1.1 of the 802.11i standard as follows:

$$csk := PRF(psk, \text{“dummy authentication”}, ts||ida||ids||hash(pk))$$ (4.3)

4. The AP verifies the ticket by checking that: (a) the identifier $ids$ matches the one in the message; (b) the current local time is within the validity period $l$ starting from timestamp $ts$; and (c) the ticket digest code $hamc$ is correct. If all checks pass, the AP considers it as a valid key-establishment request and recovers the pre-session key $psk$ by decrypting its encrypted form using AP’s private key $sk$. Now the AP derives the common session key $csk$ using Function 3, and encrypts the pre-session key $psk$ with it using an existing encryption algorithm such as AES. The AP sends $(ida, code, E_{csk}(psk))$ to the station and considers the station’s dummy authentication as successful. The ATSN is set to 4.

5. If the code is $success$ and the station can recover the pre-session key from the last message, then a common session key is established between the station and the AP. If something goes wrong during the dummy authentication process, then the station restarts it from the beginning. As in the standard, an authenticated station must be de-authenticated before it can authenticate itself again.

After a predefined period timeout, the AP updates its secret symmetric keys by setting $k_{pre} := k$ and generates a new $k$. When the AP checks a ticket, it determines which key the timestamp should use. The key-update period is longer than the largest ticket-validity period. A ticket has several functions:
1. The validity period of a ticket allows the reuse of the ticket in that period. Then steps 1 and 2 in the dummy authentication and key-establishment algorithm can be omitted, which can enhance efficiency.

2. It allows the AP to not allocate storage resources for a station during dummy authentication, i.e., the dummy authentication is stateless on the AP’s side. For example, if we use a scheme similar to TLS, the AP needs to generate a 32-byte random number for each request and remember it in order to verify the request later. If the maximum number of resources is predefined, it will suffer from DoS attacks which are similar to association-number depletion attacks and TCP SYN flooding attacks.

3. It changes an asymmetric-cryptographic operation to a symmetric-cryptographic operation, which reduces computation time. If there is no ticket in the third step, then an attacker can waste valuable AP computation time by flooding the channel with false request messages to be decrypted with its private key.

4. It binds to a specific MAC address and introduces a timestamp as a measure to prevent replay attacks.

4.4.2 Resulting RSNA Establishment Procedure

By replacing open-system authentication with dummy authentication, the resulting RSNA establishment procedure between an AP and a station STA is shown in Figure 4.6(b). It has the following stages:

1. Network and security discovery stage. In this stage, the AP broadcasts a beacon periodically. The station can either discover the network by listening to the
beacon or optionally scan the network actively using probe request messages. The AP should respond to the probe requests.

2. **Authenticating stage.** The current 802.11i standard requires that the open-system authentication algorithm be used. We replace it with dummy authentication.

3. **Associating stage.** Every station is permitted to associate with the AP.

4. **EAP (802.1x) authentication stage.** This stage only exists in enterprise networks. The AP or the station has the option to start 802.1x authentication and exchange EAP messages to verify user/server identities. The details are shown in Figure 4.2.

5. **Four-way handshake and group key handshake stage.** The AP starts a four-way handshake to derive the keys for this session. The handshake must be completed.
before any encrypted data can actually be exchanged between this station and the AP.

6. *Encrypted data communication stage.*

Before using dummy authentication, an open-access network only has stages 1, 2 and 3. The data is not encrypted in stage 6. A WPA-PSK network does not have stage 4 as shown in Figure 4.6(a). Figure 4.6(b) shows the RSNA establishment procedure when dummy authentication is used in both an open-access and a WPA-PSK network. In a WPA-PSK network with dummy authentication, the second stage is replaced with the dummy authentication key-establishment algorithm. The other stages remain the same. In an open-access network with dummy authentication, not only is the authentication algorithm in the second stage changed, but an extra four-way handshake stage is introduced. The communication in stage 6 is encrypted by the keys derived in the four-way handshake in stage 5.

4.4.3 **Pairwise Master Key Derivation**

The dummy authentication key-establishment algorithm can be used in WPA-PSK WLANs as well as in open-access WLANs. The session key obtained in dummy authentication is used to derive PMKs in new open-access public WLANs, to derive PMKs in WPA-PSK networks, and to protect management and null data frames from spoofing. While it seems redundant to use dummy authentication in enterprise networks, there is still the benefit of protecting EAPOL data messages against eavesdropping and spoofing. The lifetime of a session key spans from its establishment to the end of the four-way handshakes during which a set of keys are derived as the standard specifies. After that, the communication uses new keys.
In WPA-PSK WLANs with dummy authentication key establishment, the PMK can be calculated as follows:

\[ PMK := csk \oplus PBKDF2(passphrase, ssid, ssidlength, 4096, 256) \]  \hspace{1cm} (4.4)

where \( \oplus \) denotes the exclusive OR operation and the second part is defined in the 802.11i standard. In open-access WLANs with dummy authentication, the same algorithm is used with the passphrase set to the string “open system”. Having obtained the PMK, the four-way handshake can be performed as shown in Figure 4.1 to confirm the possession of the shared PMK in the AP and a station and to derive a fresh pairwise transient key that protects subsequent data communications between them.

Our method only modifies the open-system authentication procedure and key-derivation algorithm. Certainly the beacon should indicate that the new authentication algorithm is supported. Everything else remains the same as in the standard. It minimally modifies the 802.11i specification, which protects existing investments therein. We can further enhance WLAN security by using this session key to do per-frame authentication, as in the next subsection. However, this must modify the frames that we want to protect.

\section*{4.4.4 Wireless Frame Authentication and Encryption}

After the successful completion of dummy authentication, the AP and the client already share a session key. This session key can be used to authenticate and validate unicast management and null data frames and encrypt the data frames that are transmitted during the EAP authentication procedure. It also helps to prevent unintended disclosure of sensitive system parameters. Using digital signatures signed by the AP’s private key, the broadcast management frames can also be protected.
from forgery. A general frame protection scheme is to encrypt the frame body and attach a message integrity code (MIC) with it that is computed with a cryptographic hash function. The resulting frame has the format:

\[
frame := (MAC \text{ Header}, eBody, pArgs, MIC, FCS) \quad (4.5)
\]

Here, the encrypted frame body \( eBody := E_{tk}(\text{plaintext frame body}, pArgs) \). The message integrity code \( MIC := H_{tk}(\text{plaintext frame body}, pArgs) \), where \( E \) and \( H \) denote some encryption and keyed-hash function, respectively. \( tk \) denotes a temporal key. \( pArgs \) are private arguments or parameters of a given encryption function, such as an initialization vector and TKIP sequence counter (TSC) in TKIP or a positive number (PN) in CCMP. The PN is incremented by one for each new frame. \( pArgs \) can be plaintext, ciphertext or a combination thereof. The \( MIC \) is added to the end of the frame before the Frame Check Sequence (FCS). The station or AP checks the \( MIC \) before accepting a frame. If a frame fails the decryption or authentication check, it is dropped. Any attempt to copy, alter, or replay the frame invalidates either the \( pArgs \) (sequence number) or the \( MIC \). The frame body can be unencrypted if only message authentication is needed.

Management frame authentication

Since the upcoming 802.11w amendment will provide additional security measures for data origin authenticity, replay detection, and robust management frame protection, we do not duplicate that functionality here. We only propose a simple method that uses the existing message protection schemes defined in the standard. Although the 802.11i standard does not define any mechanism to protect management frames, it does define two data confidentiality and integrity protocols: TKIP and CCMP.
We adopt the same encryption protocols used for the data frames to protect unicast management frames. The difference is that the management frame body is used in TKIP or CCMP instead of the data frame body. Since we don’t want to expose the common session key $csk$, we define a temporal key used for those protocols as

$$tk := PRFX(csk, \text{“temporal key”}, null)$$ (4.6)

We use $PRF^{256}$ and $PRF^{128}$ for TKIP and CCMP, respectively. This temporal key is only used until the end of the four-way handshake, when a set of new keys will be derived. Afterwards, these new keys will be used to protect frames. Because of the key’s short term, we do not define the $tk$ renewal procedure.

Broadcast and multicast management frames can be protected by a similar scheme, but the MIC is calculated from the AP’s private key, which is not efficient.

We also point out that a session key in our method is derived earlier than the keys used in the standard, in which the keys are established after EAP authentication, four-way and group handshakes. We can have additional protection of the frames before and during EAP authentication and handshakes, which cannot be provided by the new amendment.

Null data frame authentication

The existing standard does not protect null and QoS null data frames, since there are no payloads in the frames. However, by applying the general protection scheme, we can extend the current standard by modifying TKIP and CCMP to calculate the MIC only. The timestamp in the previous beacon is treated as filed plaintext data, even though it is not in the resulting frame, i.e.,

$$frame := \text{MAC Header, null, } pArgs, H_{tk}(\text{“last timestamp”}, pArgs), FCS$$ (4.7)
In this case, even if there is no data to be encrypted, the MIC is different for
each frame because of the changing timestamp and increased sequence number TCS
or PN. This makes forging and replaying the null data frames useless.

The TKIP MIC only provides weak protection of the integrity of the MPDU data
field. CCMP protects both the integrity of both the MPDU data field and selected
portions of the IEEE 802.11 MPDU header.

**EAP and handshake messages protection**

There are data protection schemes (for example, TKIP and AES-CCMP) in the
802.11i standard, but these protections can only be provided after successful EAP
authentication. We can further extend them to data frames sent before and during the
EAP authentication process using the temporal key $tk$ derived in this subsection. EAP
messages and (four-way) handshake messages can be protected during encryption and
decryption using TKIP or CCMP with the temporal key $tk$. The sender encrypts the
payload, adds the message integrity check code, and sends the resulting frame. The
receiver then checks the originality and integrity of the frame and processes it after
decryption.

4.5 Discussions

*Evaluation:* We compare the message number of our proposed solution with other
wireless network settings. The results are shown in Table 4.1. It easily follows that
open networks have the fewest number of wireless messages, the least AP processing
burden, and the shortest connection setup time. Although public key processing is
performed in the authentication server in WPA-EAP-PEAP authentication, the con-
nection setup time is the longest because this setting has the most wireless messages
Table 4.1: Performance Comparisons

<table>
<thead>
<tr>
<th>Authentication algorithm</th>
<th>Message number</th>
<th>AP processing setup time</th>
</tr>
</thead>
<tbody>
<tr>
<td>Open</td>
<td>2</td>
<td>least</td>
</tr>
<tr>
<td>PSK</td>
<td>2 + 4</td>
<td>2nd least</td>
</tr>
<tr>
<td>Dummy</td>
<td>4+4</td>
<td>most</td>
</tr>
<tr>
<td>EAP-PEAP</td>
<td>&gt; 2 + 4 + 8</td>
<td>middle</td>
</tr>
</tbody>
</table>

to exchange, which takes the greatest amount of time. Dummy authentication requires the AP to perform extra processing. The connection setup time is still less than that of EAP-PEAP authentication.

The dummy authentication process also works without the last message. In this case, it reduces to a three-way handshake. However, if something goes wrong, it will not be discovered until the next message.

Variations: To reduce the number of messages in dummy authentication, the AP’s public key can be sent in probe-response or beacon frames. In that case, only two messages are needed. This process is as follows: (1) The client generates a session key encrypted with the AP’s public key and sends it to AP in a dummy authentication request; (2) The AP returns a response. However, since probe-response frames are sent more often than authentication request/response frames, this method has greater air interface overhead than the proposed method, because a certificate (the payload) is needed in every probe-response frame.

Another variation of the method is to separate the key transfer procedure and the dummy authentication procedure as shown in Figure 4.7(a) and 4.7(b).
In the improved method, we add two more messages—key request and key response—for the public-key transfer procedure. Dummy authentication and key transfer are separated into two procedures to reduce the frequency of sending large certificates, thus improving efficiency and alleviating possible DoS attacks. In the key-response frame, there is a timestamp that indicates when the certificate was last updated. This information and certificate signature can also be included in the beacon frames so that a station can check consistency or obtain the latest information. This variation has the advantage that a station only needs to request the AP’s certificate once and the AP can process the key request in batch mode. This is to delay the key response and broadcast it once for awhile. The disadvantage is that two more messages are needed.

Figure 4.7: Variations of Dummy Authentication.
We can reduce the number of messages in dummy authentication to further improve the efficiency by transmitting some information in (re)association request/response frames as in Figure 4.7(c). Thus only two messages remain for the dummy authentication procedure. This improvement also has some disadvantages, such as more modification (association, re-association request and response frames) to the standard, which cannot prevent forged de-association frames during the dummy authentication process. The resulting improved method includes the public-key transfer shown in Figure 4.7(a) and the dummy authentication and modified association procedure shown in Figure 4.7(c).

Existing protocols, such as the Transport Layer Security (TLS) protocol, are widely used in wired networks as well as in some EAP methods. However, they cannot be used in 802.11 MAC protocols directly, since (1) they are not MAC protocols and require reliable transport-layer services that can only exist after the establishment of a RSNA; (2) the complex handshake procedure with many messages exchanged is inefficient for noisy wireless communication; One major difference between TLS and our dummy authentication is that while TLS is designed to provide mutual authentication, dummy authentication provides no *real* authentication—it is just used to establish a short-term session key instead of proving the identity of the client as in the traditional meaning of authentication.

In our method, all algorithms except public key encryption/decryption, such as TKIP, MIC, and AES, are already implemented in current products. We do not implement negotiation and multiple encryption/hash-function algorithms, so there are fewer messages exchanged and lower complexity than SSL/TLS protocols. Hence
there is less program code and memory usage compared to methods with an authentication server, such as a free RADIUS implemented in an AP. On the AP side, most wireless routers on the market provide firmware upgrade functions. It is easy to upgrade them to support dummy authentication. On the station side, the software driver is more easily updated than the firmware.

*Rogue-AP and MITM Attack:* The proposed method partially solves the rogue AP threat in open public WLANs. If each AP in a public WLAN has a legitimate digital certificate signed by a well-known Certification Authority, a client can verify the AP’s identity, hence protecting it from rogue APs. In case an AP only has a self-signed certificate, the rogue AP threat is a pre-existing vulnerability that is neither alleviated nor enhanced by the presence of dummy authentication.

When a self-signed certificate is used, man-in-the-middle attacks (MITMs) cannot be prevented in an open public WLAN. However, it will not be a problem in enterprise WLANs since mutual authentication is adopted in enterprise WLANs. In WPA-PSK, outsider MITM attacks can be prevented. Insider MITM attacks can also be prevented if the attacker does not have the AP’s private key.

We choose the key-agreement method instead of a key-transport method because a certificate might be self-signed. The simplest form of key transport might be that a station chooses a session key and it sends the encrypted session key to the AP after it obtains the AP’s public key. In this case, an attacker can force the station to send the session key to him and forward it to the real AP. Then the attacker can observe all the traffic with the session key.

*Denial-of-Service Attack:* An attacker may launch a DoS attack by flooding messages. Since the response frame with a public key or certificate is large, an attacker
can saturate a wireless channel by flooding the first message. This can achieve the same effect as probe-request frame flooding attacks. We can alleviate this attack effect by modifying the protocol as described before in this section. An attacker may try to flood the third message hoping to exhaust the AP’s computation and memory resources. In order to do this, he must send frames with a different MAC address and forged tickets. We believe our method can resist this attack, since the AP can verify the ticket using its secret symmetric key instead of its public key. The AP only continues to decrypt the pre-session key after the ticket is validated. The AP will not respond to a second such authentication message because once the AP processes it, the station enters the “authenticated” state. According to the 802.11 MAC protocol, the expected messages at this state are associate request and de-authentication. We can also include digital signatures of the AP’s messages in steps 2 and 4 to prevent them being forged.

SSL/TLS: Public-key cryptography is widely used in wired network protocols and applications such as in the Secure Socket Layer (SSL) and Transport Layer Security (TLS) protocols, Secure Shell (SSH), etc. In WLANs it is also used in some EAP methods, such as in EAP-TTLS (Tunneled Transport Layer Security protocol), EAP-PEAP (Protected Extensible Authentication Protocol) and EAP-FAST (Flexible Authentication via Secure Tunneling). SSL/TLS cannot be used in 802.11 Medium Access Control (MAC) protocols directly, since (1) SSL and TLS are not MAC protocols and require reliable transport-layer services that can only exist after the establishment of a RSNA; (2) the complex handshake procedure with many messages exchanged is inefficient for noisy wireless communication; (3) multiple cipher suites, hash functions and key agreement protocols are unnecessary. One major
difference between SSL/TLS and our dummy authentication is that while these protocols are designed to provide mutual authentication and channel encryption, dummy authentication provides no *real* authentication—it is just used to establish a session key.

To protect existing investments and utilize existing centralized authentication servers (RADIUS), the 802.11i standard does not use a Public Key Infrastructure (PKI) in the first place. This is also because both the AP and the station must have a certificate in order to perform mutual authentication in an enterprise network. Public key deployment, especially at stations, is a huge management problem. Certificate management, including certificate provisioning, storage, renewal, revocation, and configuration, is beyond the scope of the specification. The EAP-TLS, EAP-TTLS and EAP-FAST protocols actually use SSL/TLS during an EAP authentication procedure. However, as they are only authentication methods, they fail to provide more protection for a WLAN than what a PKI can offer.

In our method, only the AP needs a public key. The mutual authentication procedure in enterprise networks is unchanged; it is simply protected. The public key is only used at the beginning for session key generation. All algorithms except RSA, such as TKIP, MIC, and AES, are already implemented in currently available products. We do not implement negotiation and multiple encryption/hash-function algorithms as in SSL/TLS protocols, so there are fewer messages exchanged and lower complexity than these protocols. Hence there is less program code and memory usage compared to methods with an authentication server, such as a free RADIUS implemented in an AP. Thus our approach is suited for resource-constrained APs. On the AP side, most wireless routers on the market provide firmware upgrade functions.
It is easy to upgrade them to support dummy authentication. On the station side, the wireless network card driver is more easily updated than the firmware.

4.6 Summary

In summary, we have two main contributions in this section. First, we deeply analyze the security issues ignored by the current 802.11i standard as well as possible attacks resulting from each issue. We also propose a null-data frame based denial of service attack and show the effectiveness by experiment results. Second, we propose a new dummy authentication key-establishment algorithm to setup a session key for opportunistic encryption that is used to improve WLAN security. We also discuss some variations to reduce the number of messages and improve the efficiency of the new method. Although based on a simple idea, the opportunistic encryption can provide link-layer data encryption in open wireless networks, separate session keys for different users and protection for important frames such as management frames, null data frames, and EAP authentication messages. Our solution has the advantages of simplicity, compatibility with the standard with few modifications, and low overhead in computation, memory usage and channel utilization. A device can support both open-system authentication and dummy authentication, which protects existing investments in 802.11 networks.
CHAPTER 5

OPPORTUNISTIC AUTHENTICATION IN WIRELESS WEB APPLICATIONS

Due to the open nature of wireless transmission and rogue access point problems, wireless users have more risk of being attacked by a Man-In-The-Middle (MITM) than in traditional wired networks. In this chapter, we propose to take the opportunity that a user might have a trusted device such as a mobile phone, to aid an authentication in wireless networks to thwart man-in-the-middle attacks.

5.1 Motivation

HTML form-based web authentication is a widely used authentication scheme that allows the application to directly collect user credentials via an HTML form. In this type of web authentications, users are usually asked to input a username and a password. However, this type of authentication is not secure even when the HTML form is used with the Secure Socket Layer protocol (SSL) or its successor, the Transport Layer Security protocol (TLS). Keyloggers and phishing attacks can extract user identity and sensitive account information.

Since this scheme only authenticates a user by a password, it is known as a single-factor web authentication. For applications that require strong security, such as an
online banking or financial services, two- or multiple-factor authentication schemes are recommended.

The One-Time-Password (OTP) token is a popular form of authentication factor because of its strong security and relative ease to use. Tokens like RSA’s SecureID, Vasco’s Digipass, Verisign’s Security Card and other forms of one-time password devices are popular in big enterprise networks. One-time password thwarts keylogging malware and replay attacks. However, it is not available for general online applications due to the high cost of tokens and the maintenance. Moreover, current one-time password systems have two major problems.

First, current one-time password authentication schemes cannot resist man-in-the-middle attacks, even when the schemes use SSL or TLS. For example, Russian phishers launched a sophisticated MITM attack against the hardware-token-based, two-factor authentication scheme used by Citibank [77]. Another group of hackers ripped off customers of the Dutch bank ABN AMRO, which also issued hardware tokens [78]. The wireless environment make the situation even worse, since a rogue access point can intercept all packets that are transferred between a client and a web server.

Second, a one-time password token cannot be used in multiple web applications deployed by different institutions. Usually a token from one supplier cannot be used in another supplier’s system, although the Initiative For Open Authentication (OATH) [79] is trying to improve this situation. This is because most good one-time password technologies are patented and the patent owners do not make them available to the general public, which gives rise to versatile one-time password systems and tokens with different working mechanisms, forms, interfaces and methods of use. They are
simply not compatible to each other. Ultimately, current one-time password tokens are local resources in an institution. If a user wants to authenticate himself to multiple web sites, he must obtain multiple tokens from each of the institutions. That greatly wastes resources and poses difficulties for users.

To make one-time password system available to general public and overcome the above drawbacks, we propose to take the opportunity that a user might have a trusted device such as a mobile phone, to aid an authentication. We also propose a user-oriented one-time password system based on a new concept, i.e., the *one-time password reference service*. The one-time password reference service effectively enables the sharing of one-time password information across institutional boundaries. It makes a one-time password device a global resource. It eliminates the need for users to carry several tokens in order to log in to multiple web sites. We also propose a new one-time password calculation algorithm for this system. The new algorithm calculates a one-time password that takes the dynamic connection information and the identities of each entity involved into account. By binding the IDs of each entity involved and the connection/session context information, the system can protect users from possible attacks, such as phishing, pharming, keyloggers and MITM attacks.

5.2 Related Work

5.2.1 One-Time Password

In one-time password authentication, no static secret data is transmitted via an insecure network. A new one-time password is created for each authentication. One-time password systems have existed for at least two decades, but they have only recently received mass-market attention [80]. The well-known S/Key [81, 82] is an
early one-time password system, based on hashing, that protects against password replay. HOTP [83] is an excellent Hash-based Message Authentication Code (HMAC)-based one-time password algorithm. It is a cornerstone of the OATH. Most high-quality one-time password technologies, like RSA SecureID, are patented and the patent owners do not make them available to the general public.

One-time passwords take different forms. For example, Elcard is a paper card that contains one-time passwords, which is securely printed and mailed to the customer. One-time password can also be sent to the customer’s cell phone via Short Message Service (SMS). However, the most popular form is that of a hardware token. Usually one-time password tokens are not implemented in software on a client’s PC. In all cases, a back-end system that verifies the one-time password is needed.

5.2.2 Web Authentications

To enhance the security of online applications, many web authentication schemes [84–93] are proposed. In this chapter we only give a brief summary. As an alternative to text-based authentication methods, graphical passwords, e.g., virtual keyboards, Passmark SiteKey®, PassFaces, and PassClicks were developed and gained growing interest during the last decade. In [85], Suo et al. conducted a comprehensive survey of existing graphical password techniques.

Although it is more difficult to break graphical passwords using the traditional attack methods like brute-force search, dictionary attack and keylogging, current graphical passwords’ authentication schemes, including the latest CAPTCHA [86] systems, have a common assumption - that a server can be properly authenticated in TLS handshake procedure - which is not the case in many situations. For example,
Professor Markus Jacobsson of Indiana University announced a demonstration of a successful “deceit-augmented man in the middle attack” against Bank of America’s SiteKey service [94].

Adida’s BeamAuth [87] is a two-factor web authentication technique that relies on a specially crafted bookmark created during the registration process as the second factor. It can prevent a number of phishing attacks. However, as stated in [87], pharming attacks on SSL sites where users pay no attention to certificate validity indicators may successfully defeat BeamAuth.

Mizuno et al. [89] propose an authentication scheme using multiple communication channels. They authenticate the user with his device based on the method of binding trusted and non-trusted channels using a session ID, which has several drawbacks. First, it uses a mobile phone network-based authentication that is out of web applications’ scope. Second, users must reside in an area receiving wireless service coverage. And most importantly, by relaying a session ID, a well-designed MITM attack could be launched in un-secured links.

Parno et al. [90] proposed a Phoolproof phishing prevention system that employs trusted devices to protect users against phishing attacks. The trusted device stores a bookmark and a certificate for each of the user’s online accounts as well as a self-signed user certificate generated by the device. The device performs cryptographic operations without revealing the secret private key to the user’s computer. By storing all user certificates, the web server is configured to do TLS mutual authentication. Their solution assumes the Bluetooth channel between PC and a cell phone is secure. It also requires modifications of a web browser (Mozilla Firefox) and the client’s
SSL/TLS protocol implementation. The user must revoke his certificate if his cell phone is lost or replaced.

MP-auth [91] is a secure single-factor authentication scheme that uses a trusted device. The user’s password is input to a cell phone instead of web forms on a PC browser. A Firefox extension is needed to modify the existing SSL/TLS handshake sequence and pass messages between cell phone and web server. It requires that a web application’s public key be distributed and installed into users’ cell phones. However, it didn’t give a complete solution for this, especially for revocation and update. It also requires a communication channel between a cell phone and PC.

Public Key Infrastructures (PKIs) have long been proposed as a method for users and servers to authenticate each other. SSL and TLS both rely on PKIs and are designed to prevent eavesdropping, tampering and message forgery. They are commonly regarded as secure. However, the actual implementations use only unilateral authentication due to the lack of universal PKIs for deploying digital certificates.

Many anti-phishing proposals and dozens of software tools (e.g., Calling ID Toolbar, eBay Toolbar, EarthLink Toolbar, SpoofGuard, Microsoft Phishing Filter, GeoTrust TrustWatch Toolbar) are designed to detect spoofed web sites. However, they do not focus on authentication. Furthermore, most tools have usability problems [93] and are vulnerable to keylogging attacks.

5.2.3 The Man-In-The-Middle Attack

SSL and TLS were designed to provide two features, i.e., mutual authentication of clients and servers as well as channel encryption. In typical web applications, an SSL/TLS connection is established without using a client certificate.
A digital certificate is the most popular way to identify a web site. Web browsers will automatically try to verify and validate a server certificate when connected to a site over an SSL/TLS connection. Browsers verify that the site name in the URL is the one registered to the site in the certificate and ensure that the certificate is not out-of-date, has not been revoked, was signed by a trusted Certificate Authority (CA), etc. Warning dialog boxes may be displayed if there is something wrong or suspicious. However, in many cases, these warnings are false alarms, so most users have grown accustomed to just “clicking through” such warning dialogs. Since these unsuspecting users do not pay attention to certificate validity indicators (e.g., a lock icon in the web browser’s “security status” bar) and/or cannot identify an inappropriate SSL/TLS certificate, they are especially susceptible to MITM attacks.

Another problem is that the CAs have devalued trust in TLS certificates by removing any real verification during the certificate enrollment. One can obtain a TLS server certificate simply by owning a domain name. Some CAs do not perform background checks to validate the entity receiving the server certificate. Even well-known browser-trusted CAs sometimes issue certificates to improper entities. Cybercriminals can and do obtain certificates that they then use in phishing scams. For example, VeriSign erroneously issued two digital certificates to someone masquerading as a Microsoft representative. Equifax, a subsidiary of Geotrust, issued two digital certificates to phishers who attacked the tiny Mountain America credit union in Salt Lake City, Utah [95]. Thus, checking the certificate information every time one negotiates a TLS session to check its validity is not foolproof.
Besides phishing and pharming, there are other means to implement MITM attacks. For example, Address Resolution Protocol (ARP) cache poisoning and Domain Name System (DNS) spoofing can be used to implement MITM attacks.

The MITM attacker intercepts all communications between the client and a server by which the client is attempting to communicate via TLS. The attacker intercepts the legitimate keys that are passed back and forth during the SSL handshake, substitutes its own, and makes it appear to the client that it is the server, and to the server that it is the client.

The encrypted information exchanged at the beginning of the TLS handshake is actually encrypted with the attacker’s public key or private key, rather than the client’s or the server’s real keys. The attacker ends up establishing one set of session keys for use with the real server, and a different set of session keys for use with the client. This allows the attacker not only to read all the data that flows between the client and the real server, but also to change the data without being detected.

The SSL/TLS MITM attacks succeed because the user was fooled by something presented on the screen, not because of insecure communication technologies. Therefore, if we can help a user to verify the server, we can thwart MITM using existing technologies.

5.3 Opportunistic One-Time-Password Authentication System

5.3.1 System Infrastructure

There are several entities involved in an authentication process. They are a user, a client PC or notebook, a trusted device, e.g., a mobile phone (MP), a Web application Server (WS, e.g., a bank server, perhaps with internal or external authentication
servers), and a new one-time-password Reference Server (RS). There is no special requirement for a web server and a client PC.

*One-time-password Reference Server*: Unlike authentication servers in traditional one-time-password systems, the role of a reference server is not to authenticate a user, but to provide a one-time reference for the user. The reference server tells a web application server that if a user has cell phone number X, he should know an one-time-password Y corresponding to a challenge Z associated to a connection, where X and Z are provided in an OTP request message as shown in 5.1 (Y is in the response).

*Tokens*: Our one-time-password token runs as a secure Java application on a mobile phone and is capable of generating both time-based and challenge-response-based one-time passwords. The application is protected via a PIN code.

*Communication Link between Web Server and Reference Server*: Both the web server and the reference server have their own digital certificates and can be configured to perform mutual authentication so they can properly identify and authenticate each other. Furthermore, communication between servers is secured via VPN or SSL/TLS.
A reference server provides service to multiple web servers, which facilitates sharing of one-time-password tokens by multiple web sites and applications. This is the primary advantage of the one-time-password reference service.

*Link between Mobile Phone and PC:* A direct communication link between a mobile phone and a PC like Bluetooth, Infrared Data Association (IrDA) or a USB is not required. The presence of such a link would facilitate the use of the one-time-password token on mobile phone, make information flow transparent to a user and greatly improve usability. However, it may require the installation of additional client applications, browser add-ons, plug-ins, or special hardware and drivers that are susceptible to attacks. If the link is not present, the user acts as a relay, copying information between the phone and the PC. Though a user’s limited attention and abilities can result in mistakes, he can also actively protect himself against attackers. Our scheme gives him appropriate methods to detect frauds in progress, though they require more interaction in the authentication process than simply looking at a URL or an icon in a toolbar.

### 5.3.2 One-Time-Password Reference Service

Current one-time-password tokens are institution-oriented, meaning that if a customer wants to access several banks, he must have a token for each bank. These tokens may come from different suppliers, have different forms, and use different cryptographic mechanisms. It is inconvenient for him to bring several tokens and learn how to use each of them. A unified user-oriented one-time-password token will perform much better.
Now, to illustrate how the one-time-password reference service works, we assume for example that a wireless carrier provides the one-time-password service. We assume a customer already has a cell phone contract. Therefore, he already has an account with a carrier, which knows his name, address, mobile phone number, the device’s International Mobile Equipment Identity (IMEI), and his International Mobile Subscriber Identity (IMSI); the latter is stored in his SIM card. The carrier may even have his payment information such as his credit card number or bank account number. When he accesses his online phone account, he enters his password and answers personal verification questions – the secrets he shares with the carrier. All this information is unique to a customer and can be used to identify him in one-time-password generation. The result is an one-time-password. No personal information is disclosed to a third party.

Server Side: When a customer tries to log in to his online bank account, the HTML form requires an one-time-password in addition to his username and password. The bank’s authentication server needs to compare the one-time-password returned by the user and a second one-time-password acquired from the reference server. Unlike traditional settings, the second one-time-password is not generated by the bank’s back end system; rather, the reference server generates it. This one-time-password is bound to the bank’s identification, the reference server’s identification and the user’s mobile phone number. Since all servers are equipped with digital certificates and perform mutual authentications, an attacker cannot impersonate a bank to obtain a one-time-password.

Since bank customers have mobile phones registered with different reference servers, a web service may be needed to unify the interface. This new web service finds out
which phone number belongs to which carrier and directs a bank server to a proper reference server. Thus, a bank server needs to contact only one web service. We do not discuss the details of this case further in this chapter.

User side: There are three modes for a user to obtain a one-time-password: online mode, offline mode and mixed mode. In all modes, customers need to download and install a MIDlet - a Java application for embedded devices - from the carrier onto their mobile phones. A mobile phone may store web server information (url, certificates, no password) of each user’s online account, acting like an electronic assistant.

In the online mode, a mobile phone always connects to a reference server to obtain a one-time-password associated with a particular web server each time a password is needed, as shown in Figure 5.1. This mode applies to the case in which the mobile phone is always able to connect to the Internet.

In the offline mode, the mobile phone itself calculates a one-time-password. All the needed information for calculating a one-time-password is configured or input by customers or obtained during the communication with the PC. This mode applies when a mobile phone does not have Internet functionality or a user has no digital service plan.

In the mixed mode, a customer needs to download a web server’s information from the reference server for the first time. The download can be performed in batch mode or on-the-fly. Batch downloading means that the mobile phone fetches multiple web servers’ information at once, so the MIDlet can calculate the one-time-password using stored information. If the current web server’s information is not stored in the mobile phone, then it can connect to the reference server requesting the server’s information. This mode can reduce airtime charges.
In summary, both the user and the one-time-password reference server know the key (shared secret), the user’s phone number, the phone’s IMEI and current time. Both the user and the web server know the username, the password for the website, the user’s phone number, the challenge and the connection/session context information. In the one-time-password request message, the sender (either a web server or a user’s phone) send the challenge, server/users view of session info and the three IDs involved either explicitly or implicitly. In the one-time-password response message, the request one-time-password is returned by the reference server.

The primary advantage of the one-time-password reference service is that a one-time-password token can be shared by multiple web sites and applications. The support and operational models are relatively simple, and they are also scalable. Only a few organizations need to issue, deploy and manage one-time-password credentials for its users. This service is particular suitable for cellular service providers, ISPs or identity service providers.

5.3.3 One-Time-Password Calculation Algorithm

We take an approach similar to [83] in order to calculate a one-time-password. Our one-time-password is a 32-bit binary code (represented as 8 hexadecimal digits) derived from multiple factors. Our one-time-password calculation utilizes the following factors.

- something you know (shared secret)
- something you have (mobile device)
- where you are (INFO - client IP)
who you want to communicate with (INFO - web site identification: server name, url, certificate and its fingerprint)

- organizations we both know and trust, and perhaps they know us very well (OTP reference providers and CAs)

The factor not used in our one-time-password is who you are (biometric information, such as fingerprints, iris pattern, face and voice), and it can be included in the scheme if needed. Our one-time-password is

$$OTP = truncate(HMAC_{SHA1}(key, DS))$$ (5.1)

where $HMAC_{SHA1}$ is the well-known keyed-Hash Message Authentication Code function; $key$ is a shared-secret between a one-time-password reference server and a user (not the online account password); DS is a dynamic string that is the concatenation of the Web Server ID ($ID_{WS}$), Reference Server ID ($ID_{RS}$), User ID ($ID_{U}$), Mobile Phone device ID ($ID_{MP}$), Timestamp ($TS$), Challenge Number ($CN$) and connection information ($INFO$, we will discuss it in section 5.3.4), i.e.,

$$DS = ID_{WS}||ID_{RS}||ID_{U}||ID_{MP}||TS||CN||INFO$$ (5.2)

A server’s identification is its URL concatenated with the fingerprint, i.e., the SHA1 hash, of its certificate:

$$ID_{WS} = URL_{WS}||Hash(CERT_{WS})$$ (5.3)

$$ID_{RS} = URL_{RS}||Hash(CERT_{RS})$$ (5.4)

An institution may have multiple certificates (one for each of its servers), corresponding to one pair of private/public keys. Therefore an institution may have multiple IDs.
*truncate* is a function that transforms a 20-byte SHA1 hash into a 4-byte binary code. One possible truncate function is as follows. First, the SHA1 hash is arranged into the form of byte 0 to byte 19. Then we let the index be the low-order 4 bits of byte 19, which has a range of 0-15. Next we take 4 consecutive bytes from the hash starting from the index. Finally this 4-byte binary code is represented as 8 hexadecimal digits.

We denote the cell phone number as $ID_U$ and the cell phone IMEI as $ID_{MP}$. The timestamp will change every minute. It is in the format of MM/DD/YYYY/HH/MM in GMT time. We assume cell phones always synchronize with their carriers, which have a relatively high time accuracy.

We distinguish three cases depending on how timestamps and challenges are used. In case 1, since only the challenge is null (an empty string), then it is a time-based one-time-password, which requires the synchronization of mobile phones and the server. In case 2, since the timestamp is null and the challenge is not null, then it is a challenge-response one-time-password, which removes the synchronization requirement. In the last case, both the timestamp and the challenge are not null, which offers better assurance. If both the timestamp and the challenge are null, then the connection INFO must be dynamic, e.g., it considers a message authentication code in the TLS handshake finish message. Otherwise, the generated code is static. Thus, we did not consider the case.

A reference server can actually be the same server as a web server. This works like the traditional one-time-password system, in which a mobile phone one-time-password is used only for one web site - then $ID_{WS}$, $ID_{RS}$, $ID_U$, and $ID_{MP}$ can be null. In this case, the one-time-password is only bound to the key, the time and
the dynamic connection INFO. A one-time-password request from a web server to a reference server includes $ID_U$, $CN$ and $INFO$. $ID_{WS}$ and $ID_{RS}$ are implied and confirmed in the mutual authentication process. The key, $ID_{MP}$ and $TS$ are local information for a reference server and a mobile phone. The response message contains the one-time-password, as is shown below.

\[
WS \leftrightarrow RS: \text{mutual authentication, generate session - key}
\]

\[
WS \rightarrow RS: E_{\text{session-key}}(ID_U, CN, INFO)
\]

\[
WS \leftarrow RS: E_{\text{session-key}}(OTP_1, OTP_2)
\]

Since a user needs some time to type his username, password and OTP code in the web form, it can take from several seconds up to a minute when a user calculates or requires a code. Reference server returns two consecutive one-time-passwords to web server corresponding to current time and the next minute.

### 5.3.4 MITM Attack-Resistant Factors: Connection Context

A MITM attack breaks one single connection from a client to a server into two connections. Information specific to a connection can be utilized to calculate a one-time-password, which helps preventing MITM attacks. We discuss the effectiveness of taking information from different communication layers.

In a connection, the client and server’s Internet Protocol addresses (IP) are important pieces of information that can easily be obtained. In most cases, e.g., in DNS spoofing and/or phishing attacks, an attacker will have an IP address different from the client and the server. For example, an HTML form can display the client’s IP address and geo-location information, say, the IP address 164.107.163.232 is connecting to a server from Columbus, OH, USA. When a client is behind a proxy or a Network
Address Translation (NAT) server (usually a home wireless router), the web server cannot get the client’s real IP address. In this case, a user can still check the proxy IP, considering the fact that a user usually knows the situation. Taking IP into account will help prevent MITM attack, but it is not foolproof.

Two cases complicate the situation. First, anything displayed in an HTML page can be forged or relayed by an attacker. If an attacker does not change the IP address in an HTML page sent by the server, i.e., the attacker’s IP is displayed, a user can detect suspicious activity. If an attacker changes the IP address to the client’s real IP, this information can be used to calculate one-time-password.

Another case is in an ARP cache poisoning attack, in which an attacker can fool the gateway to believe that his Medium Access Control (MAC) address corresponds to the client’s IP address and/or fool the client to believe the attacker’s MAC address corresponds to the gateway’s IP address. Thus, the attacker’s IP address is transparent to both client and server. This kind of attack must reside in a local network and there are existing techniques to detect or prevent it.

To obtain information in the transport layer, e.g., a TCP/UDP port number, the system software must be modified. Complications different than those above arise with proxies and NAT servers, since the port numbers are dynamically calculated and users do not know what they should be.

Using information in SSL/TLS sub-layer, e.g., a message authentication code in SSL/TLS handshake protocols’ finish messages, a server’s public key or certificate and even the fingerprint thereof can bind the one-time-password to the current connection. The TLS session ID is not useful; because the server chooses it, it can be easily forged. Ideally, using a message authentication code in finish messages suffices to identify a
SSL/TLS session, though it requires modifying the client and the server’s system software as well as a user’s browser. This does not appear to be a scalable choice in the near future, whereas using a server certificate’s fingerprint appears to be the best choice for us, as many browsers can display this information to a user.

Using information in the application layer, e.g., HTTP headers and cookies, is not very useful to us, since all of them can be intercepted and forged by an attacker. However, the browser’s URL is important, as it indicates to which server a user wants to connect. Therefore, we choose connection information as in 5.5 or 5.6.

\[
INFO = IP_{client}||IP_{server}||URL||\text{Hash}(CERT_{server})
\]  
(5.5)

\[
INFO = URL||\text{Hash}(CERT_{server})
\]  
(5.6)

5.3.5 Opportunistic One-Time-Password Web Authentication Work Flow

System Setup: Registrations

*Customer and web server:* A customer needs to give his cell phone number and the name of his one-time-password reference (cellular) service provider to a bank when creating an online bank account.

*Customer and reference server:* Ideally the Java application should be pre-installed by the service provider for new customers, as is the case with some mobile phone games. However, existing customers can use a Java-enabled mobile phone to connect to the reference server as a thin client using Wireless Application Protocol (WAP) under SSL to download the Java application’s jar/jad files. Besides “traditional” information, e.g., customer’s name, address, phone number and IMEI, a shared secret *key* is needed for one-time-password generation. This secret can be established by
many ways, e.g., a temporary activation code in an email or text message that the customer changes later and input into the phone. The reference server might also provide a list of supported web sites for a user to choose as “favorites.”

*Web server and reference server:* The case of whether a reference server provides free access for web servers or a paid service for web servers is beyond the scope of this dissertation. In both cases, a web server need to register its information (such as URL and certificate) with the reference server before the first one-time-password request is issued.

**Authentication Work Flow**

The basic protocol steps are shown in Fig. 5.2 and are described below as well.
1. The user enables the one-time-password token on his mobile phone with his PIN and the secret key, launches a browser on the client PC, and visits the bank web site.

2. The PC and web server establish a TLS session and the web server sends an HTML form that is displayed on the PC’s browser.

3. The PC sends the current URL, the current certificate of the communication pair as well as other information about the current connection to the mobile phone. The mobile phone checks whether the URL is in the stored list, and if it is found there, the mobile phone compares the certificate with the stored ones. If the certificates match, a one-time-password is calculated. If the certificates do not match, an error screen pops up and no one-time-password is calculated. If the URL is not found, the mobile phone asks the user’s permission to save the url and the certificate. If the user allows this, the information is saved and a one-time-password is calculated. Otherwise, no information is saved and no one-time-password is generated.

4. The one-time-password is sent to the PC.

5. The user submits the HTML form with his username, password and one-time-password.

6. The web server first verifies the username and then checks the password. If both are valid, it sends a one-time-password request including user’s cell phone number and its “view” of the current connection information to the reference server.
7. If the phone number is found, the reference server verifies the web server’s identity via TLS mutual authentication and responds with a one-time-password.

8. The web server compares the two one-time-passwords. If they match, login succeeds; otherwise, it fails.

For steps 3 and 4, we distinguish between two cases depending on whether or not the mobile phone connects to the client PC system.

*Case 1:* If a mobile phone can connect to the PC through Bluetooth, USB or IrDA, the system can send connection information, including the server certificate, to the mobile phone, and the mobile phone can compute the one-time-password. A user still needs to type his PIN and the secret key on the keypad to launch the Java application and calculate the one-time-password, which can either be sent to the PC directly or displayed on the phone screen. In the latter case, he must manually copy the one-time-password from phone screen to an appropriate web form. This case offers the main advantage of simplifying user interactions and allowing full connection information to be transferred, thus providing better security and usability. The mobile phone can verify server certificate by comparing it to the stored web site information. If the certificates do not match, then Java application will warn the user and not return any one-time-password.

*Case 2:* If a mobile phone is not connected, the user acts as a communication path between the client PC and the mobile phone. In this case, the connection information must be carefully compressed or arranged so that the user can easily relay information between the PC browser and the mobile phone. The advantage of this case is that there is no need for extra communication paths and possible driver installations. However, to maintain usability, the human-transferred information must be limited.
5.3.6 Using Existing One-Time-Password Tokens

Our system works not only with our new cell phone one-time-password token but also with the existing hardware tokens on the market. In this case, a back-end one-time-password verifying module/subsystem is moved from the original institution to a reference service provider. We suppose a hardware token is registered off-line with a cell phone number that is associated with a user. Thus, the phone number can be used to identify a user. The original one-time-password displayed on the token acts as a changing shared secret (key) between a user and a one-time-password reference service provider. The previously described algorithm calculates a new one-time-password. To use the existing tokens, a browser plugin is needed. This plugin asks a user to input a one-time-password to the hardware token and uses it as the key to calculate a new one-time-password with the information from the browser, such as the current URL, certificate, connection information and pre-configured reference server ID. An empty string is used for the phone device ID. For challenge-response tokens, the response is used as the key.

5.4 Prototype System Implementation and Performance

5.4.1 System Implementation

We developed a prototype of the system to evaluate its performance. Our prototype consists of a one-time-password reference server, a web application server, a desktop client, and a Java MIDlet for Java 2 Micro Edition (J2ME) on a Sony-Ericsson W810i cell phone. The reference server is a Dell Dimension 5150 desktop and the web server is an IBM T60 laptop that is wirelessly connected to OSU’s campus. Both servers are set up on Windows XP Professional with IIS. We have
developed a web service for one-time-password reference server in the ASP.NET environment. Currently, communications between the web server and the reference server and the phone and the reference server are not secured by SSL/TLS. We use Sun JavaTM Wireless Toolkit for CLDC and Eclipse to develop a MIDlet based on the Mobile Information Device Profile (MIDP) specification. We use the Bouncy Castle Lightweight Crypto API to perform cryptographic operations on the MIDlet. The phone is able to connect the Internet via the new AT&T’s service. For the phone’s web service client, We use the kSOAP 2.0 library for constrained Java environments.

We developed three versions of the J2ME program. The latest version supports Bluetooth communication, the other two older versions does not. All of them provide a user-friendly interface, have a login form that authenticates the user before they are opened. Besides calculating one-time-password, they allow a user to perform tasks such as changing the PIN or secret, configuring a web server, the phone number or the working mode, browsing and synchronizing favorites. For the two earlier versions, we do not have a direct communication channel between PC and the phone. It is impossible to transmit the whole certificate or public key manually. The user need select a URL and type a hash into the phone for each connection. To reduce typing time, we provide a list of stored URLs from which the user can choose. In the first version, a full 20-byte hash needs to be typed. In the second version, a user only needs to type 4 randomly chosen bytes of a 20-byte hash generated by the program at run time. If these 4 bytes are correct, the saved hash is used to calculate the one-time-password. This version is less secure than the first one. Another possibility is to let a user compare the hash on the PC screen with that on the phone screen. However, this is the worst choice for us, since a single mistake can subvert the whole
system. In the third version, the phone can communicate with a PC via Bluetooth, thus a user need not type any connection information anymore into the phone.

We developed a toolbar for Windows IE to interact with cellphone via Bluetooth. The toolbar can automatically get connection information, such as the fingerprint of server’s certificate, and transfer all the information to the J2ME program on the phone. And retrieve the one-time-password from the phone via Bluetooth. A user only need to copy the one-time-password to the desired input field and submit the form.

5.4.2 System Performance

Table 5.1 summarizes our prototype system’s performance, which represents the average over 20 trials performed by 4 users. Synchronization time is mainly determined by the cellular network condition, Internet speed and network delay, although other factors, such as server load and the number of sites synchronized, also contribute to the time. The mobile phone and the reference server calculate a one-time-password rapidly, so the time is negligible. The time getting a one-time-password from reference server depends on Internet delay and server processing time. The times to manually configure a web server and type a certificate fingerprint are largely user-dependent. An experienced SMS user can type rapidly, which reduces the overall time. It takes some time for the PC to find and pair with the mobile phone for the first time. Once the pairing information is stored on PC, the time to obtain a one-time-password from the phone via Bluetooth is quite short.
Usability is a major concern for any new system that is to be used by the general public. We believe that typing something on a cell phone keypad and typing a one-time-password into a web form is acceptable, as many users are accustomed to type SMS messages and use one-time-password tokens. Only a few volunteers have tested our system. Initial feedback was similar to what was expected: the operation of the Java application is straightforward but it is irritating to input a long, 40-digit hash into a cell phone. Asking a user to input four randomly selected bytes of a hash improves user experience, but also reduces security. Using a browser plugin that communicates with the phone greatly improves the user experience. Further usability testing is required.

## 5.5 Discussions

In this section we provide a brief, informal security analysis and discussion of our MITM-attack-resistant one-time-password system. We assume the initial account setup procedure is secure, as many other researchers did.

Table 5.1: Operation time

<table>
<thead>
<tr>
<th>Task</th>
<th>Avg. Time(s)</th>
<th>[Min, Max] (s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Synchronize ten WSes automatically</td>
<td>12.635</td>
<td>[6.020, 23.105]</td>
</tr>
<tr>
<td>Calculate an OTP on the phone</td>
<td>0.110</td>
<td>[0.075, 0.144]</td>
</tr>
<tr>
<td>Get an OTP on phone from RS</td>
<td>12.125</td>
<td>[2.742, 33.438]</td>
</tr>
<tr>
<td>Get an OTP on WS from RS</td>
<td>0.036</td>
<td>[0.031, 0.109]</td>
</tr>
<tr>
<td>Configure a WS manually</td>
<td>175</td>
<td>[119, 241]</td>
</tr>
<tr>
<td>Type a cert fingerprint manually</td>
<td>142</td>
<td>[92, 209]</td>
</tr>
<tr>
<td>Get an OTP from phone via Bluetooth</td>
<td>5.786</td>
<td>[0.432, 35.69]</td>
</tr>
</tbody>
</table>
Secure channel over untrusted network: Currently most secure web servers are installed with digital certificates, i.e., within a PKI. As our one-time-password is bound to the identity of a web server, if the existing standard secure protocols, e.g., SSL/TLS, are properly configured for servers, it should be impossible for an attacker to impersonate a web server to steal any one-time-password from a reference server.

A secure communication channel between a mobile phone and a reference server is established via existing standard mutual authentication protocols. Without knowing the IMEI and the shared secret, an attacker cannot impersonate a user to obtain the one-time-password. Bound to connection information, a one-time-password cannot be reused by an attacker who has a different connection from the user.

Certificate checking: Ideally, a public key or certificate should be used to make sure that a web server is the intended server, which can be easily done if a direct communication channel between a PC and a mobile phone is present. Checking the fingerprint of a certificate is less secure than checking the whole certificate. The latest theoretical SHA1 collision attack reduces the number of trials for finding a collision from $2^{80}$ in a birthday attack to $2^{69}$, which does not mean that an attacker can forge a meaningful certificate for a particular web site.

By checking the certificate fingerprint in the current connection against the stored certificate fingerprints, a mobile phone protects the user from possible attacks such as DNS spoofing, pharming and domain hijacking. The involvement of a normal user in this process can also teach him to check other fields of a certificate, e.g., the issuer and subject, which may help foster good security habits. Even if the certificate fingerprint is not checked in the mobile phone, it still protects a user from MITM attacks since
the one-time-password is calculated using the attacker’s certificate fingerprint, not the real site’s certificate.

*Mobile phone theft:* Since no shared secret is stored on the device, losing the device poses no risk to a user. Holding a stolen phone, an attacker will need the user’s PIN to launch the Java application for viewing his favorites; the user’s secret to calculate a one-time-password and the user’s online password to access the account. We can remove the need for asking a user to input a secret each time the application is launched by storing it. However, this will pose a potential risk for a user, which is a trade-off between usability and security.

*Mobile phone replacement:* A user needs to update his cell phone IMEI and/or phone number with his service provider and ask for an activation code. This updating procedure should be secure, perhaps via an out-of-band channel, e.g., a “regular” phone call, postal mail or in-store person. Current businesses successfully handle these issues.

*Distributed denial of service:* Since a one-time-password server is the most important element, attacking the server poses a grave threat to the system. Although there is no general defense against such attacks, much research has been done in this area, and precautionary techniques derived therefrom can be used to alleviate the problem.

*Mobile phone malware:* Malware on mobile devices will become an increasingly important challenge as these devices become more powerful and ubiquitous, coupled with pervasive network connectivity. Much malware on mobile phones spreads via SMS, Multimedia Messaging Service (MMS) or Bluetooth. If these services are isolated from the rest of the software and resources on the platform, these malware cannot easily infect the entire platform and its associated network. Isolation of the operating
system, runtime environment, resources, and domain applications as described in the
Trusted Computing Group’s (TCG) mobile reference architecture specification can
help prevent malware infections. A Mobile Trusted Module (MTM), which is a se-
curity element and a newly approved specification of TCG, will help increase mobile
phone security. Anti-malware software for mobile devices and encryption can also
reduce the risk of malware stealing sensitive information on mobile phones.

PC malware, spyware or rootkits: Additional measures are needed to deal with
malware or rootkits on PCs. For example, a malicious browser connects to the at-
tacker using the attacker’s public key, but the information displayed when a user
checks the server certificate is the real server’s certificate. Furthermore, Transaction
Generator [96] malware simply waits for the user to log in to his account and then
issues/piggybacks transactions on behalf of the user. In these cases, only address-
ing authentication will not suffice. Transaction signing or confirmation is needed to
protect integrity. To alleviate the problem, for each single transaction, a summary is
input as part of connection INFO to calculate a one-time-password associated with
the transaction.

Deployment Incentives and Limitations: The one-time-password service is partic-
ularly suitable for cellular service providers, since they control gateway servers and
continuously develop new services such as roadside assistance and city navigation in
addition to telephony, messaging, email and web services. Other service providers, es-
pecially financial institutions, will want to adopt such a strong authentication system
that can help them reduce losses due to phishing. They already rely on other Internet
services such as DNS, those provided by ISPs, and authentication services. Adopting
a new one-time-password service should pose no difficulty. Customers would like to
use a simple secure online system that has nearly the same access method as current ones and can also protect them from various attacks. The limitation of this model is that, it is not easy to establish a trust relationship between a web service provider and a reference service provider.

5.6 Summary

Due to the open nature of wireless transmission and rogue access point problems, wireless users have more risk of being attacked by a man-in-the-middle than in traditional wired networks. In this chapter, we propose to take the opportunity that a user might have a trusted device such as a mobile phone, to aid an authentication in wireless networks to thwart man-in-the-middle attacks. By introducing the connection context information, such as the URL, the certificate and IP addresses, the one-time-password is bound to current communication session. By storing real server information and comparing the certificate of the current connection pair, a trusted device helps authenticate a server. Although based on a simple idea, this approach prevents MITM attacks against current web authentication systems. We also propose a new concept, i.e., one-time-password reference service. The one-time-password reference service makes it possible for one-time-password tokens to act as global resources that are used across institutional boundaries.
CHAPTER 6

FINAL REMARKS

In this dissertation, we focus on applying opportunistic computing to improve computation in wireless network applications. Particularly, we study four instances of opportunistic computing in wireless networks, i.e., opportunistic social networking; opportunistic localization; opportunistic encryption and opportunistic authentication.

First, we study opportunistic social networking with mobile phones. We develop E-SmallTalker, a novel distributed mobile communication system that aims to facilitate more effective social networking among strangers in physical proximity. We propose a novel privacy preserving opportunistic commonality discovery protocol which utilizes iterative Bloom filters to encode user information. We eliminate unnecessary user interactions by extending Bluetooth service attributes to publish encoded user data.

Second, we study opportunistic localization in wireless sensor networks. We propose an “anti-sensor network” system to localize an adversary’s sensors in a non-cooperative environment. We propose a localization algorithm to take full advantage of the observable signals emitted by sensors during their intermittent communications. We design improved algorithms to handle additional counter-measures that can be
employed by sensors, including message encryption and non-uniform transmission power levels.

Third, we study opportunistic encryption in wireless local networks. We identify and analyze the security issues in the current 802.11 security standard. We illustrate the severe consequence of no frame authentication by a denial of service attack example. We propose a new opportunistic encryption key-establishment algorithm that perfectly fits in the existing protocol.

Fourth, we study opportunistic authentication in wireless web applications. We propose using a trusted device to aid web authentications. We design and implement a prototype one-time-password authentication system that works seamlessly in heterogeneous environments. We propose a new concept of one-time-password reference service, which allows an OTP-token to be used in multiple webs. We also propose a new one-time-password algorithm to thwart man-in-the-middle attacks by using the connection information.

Opportunistic computing is still a new research area. We hope that our work in this dissertation can shed some light on opportunistic computing and further motivate other researchers to explore this promising area in the future.


158


[68] “Ieee 802.11w: Wireless lan medium access control (mac) and physical layer (phy) specifications: Protected management frames,” *IEEE Computer Society LAN MAN Standards Committee*.


