Security Supports for Cyber-Physical System and Its Communication Networks

Zhengrui Qin

Xuanen, Hubei, People's Republic of China

Bachelor of Science, Beijing University in Beijing, China, 2001
Master of Science, Beijing University in Beijing, China, 2004
Master of Science, Dartmouth College in Hanover, NH, USA, 2007

A Dissertation presented to the Graduate Faculty
of the College of William and Mary in Candidacy for the Degree of
Doctor of Philosophy

Department of Computer Science

The College of William and Mary
May 2016

APPROVAL PAGE

This Dissertation is submitted in partial fulfillment of
the requirements for the degree of

Doctor of Philosophy

Zhengrui Qin

Approved by the Committee, May 2016

Committee Chair
Professor Qun Li, Computer Science
The College of William and Mary

Professor Weizhen Mao, Computer Science
The College of William and Mary

Professor Andreas Stathopoulos, Computer Science
The College of William and Mary

Professor Haining Wang, Electrical and Computer Engineering
University of Delaware

Associate Professor Minh Phan, Engineering
Dartmouth College

ABSTRACT

A cyber-physical system (CPS) is a sensing and communication platform that features tight integration and combination of computation, networking, and physical processes. In such a system, embedded computers and networks monitor and control the physical processes through a feedback loop, in which physical processes affect computations and vice versa. In recent years, CPS has caught much attention in many different aspects of research, such as security and privacy. In this dissertation, we focus on supporting security in CPS and its communication networks.

First, we investigate the electric power system, which is an important CPS in modern society. As crucial and valuable infrastructure, the electric power system inevitably becomes the target of malicious users and attackers. In our work, we point out that the electric power system is vulnerable to potential cyber attacks, and we introduce a new type of attack model, in which an attack cannot be completely identified, even though its presence may be detected. To defend against such an attack, we present an efficient heuristic algorithm to narrow down the attack region, and then enumerate all feasible attack scenarios. Furthermore, based on the feasible attack scenarios, we design an optimization strategy to minimize the damage caused by the attack.

Next, we study cognitive radio networks, which are a typical communication network in CPS in the areas of security and privacy. As for the security of cognitive radio networks, we point out that a prominent existing algorithm in cooperative spectrum sensing works poorly under a certain attack model. In defense of this attack, we present a modified combinatorial optimization algorithm that utilizes the branch-and-bound method in a decision tree to identify all possible false data efficiently.

In regard to privacy in cognitive radio networks, we consider incentive-based cognitive radio transactions, where the primary users sell time slices of their licensed spectrum to secondary users in the network. There are two concerns in such a transaction. The first is the primary user's interest, and the second is the secondary user's privacy. To verify that the payment made by a secondary user is trustworthy, the primary user needs detailed spectrum utilization information from the secondary user. However, disclosing this detailed information compromises the secondary user's privacy. To solve this dilemma, we propose a privacy-preserving scheme by repeatedly using a commitment scheme and zero-knowledge proof scheme.

TABLE OF CONTENTS

ACKNOWLEDGMENTS

This dissertation would not have been possible without the help, guidance, encouragement, and support from my committee members, professors, colleagues, friends, and family.

I owe my deepest gratitude to my advisor, Dr. Qun Li, who provided me with invaluable guidance, encouragement, and support in every step of my Ph.D. study. I also thank him for all the research and life advice he gave me over the years.

I would like to express my sincere gratitude to Dr. Weizhen Mao, Dr. Andreas Stathopoulos, Dr. Haining Wang, and Dr. Minh Phan, for serving in my dissertation committee. I thank them for their help, guidance, and time.

I am grateful to my colleagues and collaborators, including Chiu Tan, Fengyuan Xu, Hao Han, Yifan Zhang, Ed Novak, Nancy Carter, Zijiang Hao, Shanhe Yi, Cheng Li, and Mooi Choo Chuah.

I am also thankful to our Computer Science Department Chair, Professor Robert Michael Lewis, and the administration team, including Ms. Vanessa Godwin, Ms. Jacqulyn Johnson, and Ms. Dale Hayes, for their persistent support and help.

Finally, I want to give my gratitude to my family, whose love, support and encouragement are always with me in this journey.

I would like to dedicate this dissertation to my parents, my parents-in-law, my wife, my daughter, and my son, for all their love and support.

LIST OF TABLES

LIST OF FIGURES

Security Supports for Cyber-Physical System and Its Communication Networks

Chapter 1

Introduction

A cyber-physical system (CPS) is a sensing and communication system that offers tight integration and combination of computation, networking and physical processes [26, 27, 28, 49, 53, 54]. CPS mainly consists of two components: a physical system and a cyber system. Typically, embedded computers and sensors in the physical system collect measurements that reflect the current state of the physical system, and then send them to the cyber system through communication networks in real time. At the same time, the cyber system processes the received measurements and obtains the status of the physical system. Based on the processing results, the cyber system responds to the physical system in real time by sending directive instructions through communication networks, achieving better performance, or maintaining system stability.

CPS plays an important role in our society, and can be found in many areas, such as aerospace, automobiles, chemical processing, civil infrastructure, energy, healthcare, manufacturing, transportation, entertainment, and consumer appliances. Due to its importance, CPS and its communication networks inevitably become the targets of attackers and malicious users. To launch an attack in a CPS, an adversary may physically break into the physical system and comprise the data acquisition component, such as the embedded computers and sensors, attack the data processing component in the cyber system, or hack into the data transmission component, i.e., the communication networks. Therefore,

to protect a CPS from attacks, security measures have to be taken in all three aspects of the CPS: data acquisition, data transmission, and data processing. In this sense, we have to protect both the CPS itself, and its communication networks with consideration at two distinct levels. First, we have to secure all components in the system, including the data acquisition components, data processing components, and the communication networks, or at least those important nodes in the system. It is important to guarantee that such an attack cannot be easily launched, if not made completely impossible. Second, we have to take measures to respond to an attack if it does happen, such as detecting the presence of the attack, pinpointing the attacked components, and taking appropriate countermeasures.

Recently, CPS has caught much attention in the research community, especially in the areas of security and privacy:

Security. CPS security can be roughly be divided into the following three major aspects. The first is sensing security, including both hardware security and software security of the sensors in the physical system. For each sensor, the sensor itself, i.e., the hardware, must be physically secure, without physical damage or malfunctioning. At the same time, the software running on each sensor must be secure. For instance, each software should be securely loaded, maintained, and verified. The second aspect is cyber security, including communication security and computing security. CPS is networked, which not only enables data fusion and delivery but also allows coordinate response actions. Therefore, the communication, including that among sensors and that between the physical system and the cyber system, must be secure. For computing security, people are concerned with data integrity and processing correctness. That is, the results of data processing must be trustworthy. The third aspect is control security. That is, the access to each entity in CPS must be authorized.

Privacy. CPS and its communication networks usually involve in many entities, including human beings and companies, and sensors continuously collect data about them. CPS privacy mainly consists of two parts. The first is identity privacy, such as whether

an entity participates in a CPS or communication network. The other is data privacy. The data collected by sensors may reveal sensitive information about human beings and corporations. For instance, smart meters in a power system can be used to infer household activities.

This dissertation focuses on the security and privacy of CPS, including its communication networks, which is an important branch of the broad research area of cybersecurity. As we can see, CPS security and privacy is a large area, and it is challenging to cover all aspects of this area. In this dissertation, we aim to use some typical CPS and communication networks, i.e., electric power systems and cognitive radio networks, as examples to demonstrate some typical problems and corresponding solutions. We hope that our work can shed light on this area and provide future guidance to researchers. Specifically, we present three security and privacy works for CPS and its communication networks. An overview of the dissertation is shown in Figure 1.1. First, we introduce a potential cyber attack in an important cyber system, i.e., the electric power system (EPS), in the cyber system domain, and propose efficient countermeasures to defend against such attacks. Next, we present a security support scheme in cognitive radio networks (CRNs), which serve as a typical communication network in CPS. Finally, we propose a privacy-preserving mechanism for spectrum transactions in cognitive radio networks. In the following, we will briefly state each problem, and describe the contribution of each work.

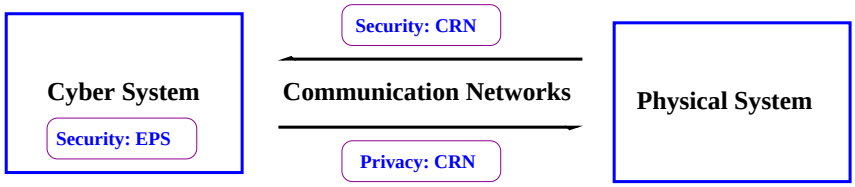

Figure 1.1: Overview of the dissertation.

1.1 Security in Electric Power Systems

An electric power system is an infrastructure network that distributes electricity from electric power generators to customers through transmission lines. It consists of several different components including power generation stations, transformers, transmission stations, distribution stations, transmission lines, and consumers. Generation stations produce electric power in variety of ways, such as water plants, nuclear plants, coal plants, wind farms, solar, tidal, and geothermal power stations. Transformers can either increase or decrease the voltage levels for efficient transmission to end users. Transmission stations transmit electricity to distribution stations, through high-voltage transmission lines, while distribution stations distribute electricity to power consumer homes through safer, low-voltage transmission lines, which are not efficient over large distances.

An electric power system is something that people cannot live without in the modern era. As a crucial component of our infrastructure in society, its security is of great importance. An electric power system may be at risk from not only natural disasters, but also malicious attacks. Active attack scenarios demand our attention and preparation in advance to cope with them effectively. For example, on March 13, 1989, a powerful solar wind storm left six million people in the Canadian province of Quebec without power for more than 9 hours [2, 23], forcing the power company to implement various mitigation strategies to recover proper power distribution.

An electric power system is usually monitored and managed by a Supervisory Control And Data Acquisition (SCADA) system installed at the control center. The SCADA system collects real-time data through meters and sensors installed at important locations throughout the power system, including power injection, voltage, current, circuit breaker status, phase shifting, and transformer settings. Normally, a state estimator in the SCADA system processes the collected data, and obtains the current status of the entire electric power system. Then, the SCADA system makes operation adjustments to the electric power system in real time, according to the current status of the entire system. This pro-

cess happens periodically, usually once every half hour, but can repeat as often as every ten of seconds!

1.1.1 Problem Statement

The most commonly used state estimators in SCADA systems are variants of the least square estimation. However, it is only possible to obtain an approximation of the real status of the electric power system due to two factors. The first is that the collected data may have errors, and the second is that there may be redundant data for the sake of reliability. Therefore, an estimator strives to find an estimation based on the collected data, which is closest to the real status of the system. Suppose that some data have large measurement errors due to a device malfunction, or some data are compromised by attackers. In either case the state estimator will obtain an incorrect estimation for the electric power system if it cannot identify and remove this ``bad'' data. As a result, SCADA systems are very likely to make bad operation adjustments to the power system and cause severe consequences.

The problem we address in this dissertation is three-fold. First, what new potential cyber attacks does the electric power system face? Second, from the view of an attacker, how should he launch such an attack with the least amount of effort and resources? Third, from the view of the control center, if such an attack does take place, how should the control center of the power system respond in order to minimize the impact of the attack, or possibly identify the attack with physical verification? The basic intuition behind our problem is this: on the one hand, an attacker wants to put forth the least amount of effort necessary to compromise some measurements in the electric power system such that these compromised measurements cannot be easily identified and removed by the control center, misleading the control center to make bad operations. On the other hand, the control center itself may not be able to identify the attack, but it still wants to find a way to respond to the attack to minimize the damage.

1.1.2 Contributions

We are the first to propose the concept of unidentifiable attacks in electric power systems. These are attacks in which the attack may be detected, but cannot be identified reliably by the control center. We demonstrate the feasibility of these types of attacks, and show that an unidentifiable attack requires the attacker to compromise a smaller number of meters, when compared with previously known attacks, and can still confuse the control center about the real state of the power system. In defense of this type of attacks, we propose a heuristic algorithm to narrow down the attack region and then enumerate all feasible attack cases. Based on all the feasible cases, we provide an optimization strategy to minimize the damage of the attack. We formulate the optimization strategy as a nonlinear programming problem, and solve it using a standard optimization package. Additionally, we provide an algorithm that may be used by a savvy attacker, that can reveal the regions requiring less attack resources, and an algorithm to be used by the control center that can determine the best set of meters for physical verification in order to eliminate the highest number of feasible attack cases. We have evaluated our optimization strategy with convincing results through simulation, and we present those results in our work.

1.2 Security in Cognitive Radio Networks

Cognitive radio networks play an important role in CPS as a typical communication network. Cognitive radio networks have created an efficient communication paradigm to enhance the utilization of the wireless spectrum. Within the traditional spectrum regulation framework, spectrum bands are exclusively assigned to wireless services by the Federal Communications Commission (FCC), resulting in inefficiency of spectrum utilization spatially and temporally. According to the FCC's statistics, the utilization of the licensed spectrum varies from 15% to 85%. The FCC has also designated a few unlicensed spectrum bands, such as the industrial scientific and medical (ISM) bands. With rapid growth

of wireless services, these unlicensed spectrum bands are filling up fast. Thus, the majority of the wireless spectrum is licensed but under-utilized, and the unlicensed spectrum is over-crowded due to the rapid increase of wireless services.

Cognitive radio networks solve the above dilemma intelligently by allowing unlicensed users to opportunistically access the licensed bands providing that they don't interfere with licensed users. Typically, a cognitive radio network consists of two types of users: primary users (also called licensed users) and secondary users (also called unlicensed users). The secondary users are capable of sensing and adapting to their spectral environment. Specifically, by carefully sensing the primary user's presence and adjusting their own transmission to times when the primary user is idle, the secondary users can utilize licensed spectrum for transmission and therefore can dramatically improve spectral efficiency. When the primary user is present, the secondary user can either shut off their transmission or adjust their transmission parameters to guarantee an acceptable performance for the primary user.

A key component of a cognitive radio network is spectrum sensing, which enables secondary users to continuously monitor the spectrum and detect the presence of the primary user. Among a variety of spectrum sensing approaches, cooperative spectrum sensing has earned a good reputation in improving the accuracy of primary user detection. In cooperative spectrum sensing, a set of secondary users is selected to share the sensing results with each other and make a cooperative decision on the presence of the primary user. Compared to a single user's sensing, cooperative sensing is more accurate and trustworthy.

1.2.1 Problem Statement

Though cooperative spectrum sensing can improve the accuracy of primary user detection, it opens a window for malicious users and attackers, who may remotely or physically capture some secondary users and manipulate their sensing reports simultaneously. For

example, a selfish secondary user could purposely manipulate his own sensing reports and/or others' sensing reports, and create the illusion that the primary user is present while it is actually idle, so that it can grasp the unused licensed spectrum selfishly.

The problem we want to address is how to recognize and pinpoint compromised data, and make a correct sensing decision under an attack. We note that a recent a framework called *IRIS* [38] can be used to iteratively run the *largest normalized residual method* to remove the ``bad'' data and obtain the final sensing decision. The underlying assumption of this framework is that the data with the largest normalized residual is most likely compromised. This assumption, however, is not always true. We find that *IRIS* works poorly under a special type of attack, which we name *cooperative attacks*. In a cooperative attack, the attacker compromises some secondary users and manipulates their readings simultaneously such that these data themselves are self-consistent. Our goal is to design an efficient algorithm to defend against cooperative attacks such that it can complement *IRIS*.

1.2.2 Contributions

In this work, we propose and design a new type of attacks in cooperative spectrum sensing, called cooperative attacks, in which an attacker injects self-consistent false data to multiple secondary users simultaneously. We illustrate that the existing approach *IRIS* works poorly under such an attack even though it can detect the presence of the attack. In defense of this attack, we propose a modified *Combinatorial Optimization Identification* (*COI*) algorithm that scans a decision tree with the branch-and-bound method to obtain all possible attack cases. We argue that these attack cases are equally possible, and the uncertainties can only be narrowed down with extra information. We demonstrate the feasibility of the attack and evaluated our algorithm with intensive simulations.

1.3 Privacy in Cognitive Radio Transaction

As mentioned in Section 1.2, the cognitive radio network technology can improve the utilization of spectrum resources through dynamic spectrum sharing among primary users and secondary users. To motivate a primary user to share its licensed spectrum with other secondary users, it is reasonable for secondary users to provide the primary user with monetary compensation. We call this a *cognitive radio transaction*, in which each secondary user pays the primary user a fee according to when and for how long the former utilizes the licensed spectrum owned by the latter.

1.3.1 Problem Statement

In a cognitive radio transaction, there are two concerns regarding the primary user and the secondary user, respectively. From the view of the primary user, the top concern is about whether its interest, i.e., the payment received from the secondary, is correct. To verify payment correctness, the primary user usually needs detailed spectrum utilization information about the secondary user. From the view of the secondary user, privacy is the number one concern. Unfortunately, disclosing detailed utilization information to the primary user inevitably compromises the secondary user's privacy, such as usage patterns, active periods, and data volume. We strive to protect the primary user's interest, while at the same time preserving the secondary users' privacy.

1.3.2 Contributions

To solve this dilemma, we are the first to propose a novel privacy-preserving mechanism for cognitive radio transactions. To the best of our knowledge, none of existing work has addresses the secondary user's privacy. Our mechanism not only preserves the secondary users' privacy, but also protects the primary user's interest. By employing a commitment scheme, and a zero-knowledge proof algorithm, the secondary user never sends any utilization information to the primary user in plain text. Rather, the utilization

information is sent in the form of commitments that do not disclose any sensitive information. The secondary user also provides zero-knowledge proofs for the payment. With the commitments and proofs, the primary user is able to verify the correctness of the payment without requiring any detailed utilization information in plain text. We have implemented our privacy-preserving scheme and evaluated its performance. Our results show that it only takes about 100 milliseconds for one utilization instance.

1.4 Organization

The dissertation is organized as follows. In Chapter 2, we introduce a new cyber attack in power systems and propose a heuristic approach to protect power systems from this new attack. In Chapter 3, we present the concept of cooperative attacks in cognitive radio networks and countermeasures to defend against such attacks. In Chapter 4, we point out the necessity of a privacy-preserving mechanism in cognitive radio transactions, and design and implement such a mechanism. Finally, Chapter 5 concludes the dissertation, and foresee some future work.

Chapter 2

Defending Against Unidentifiable Attacks in Electric Power Systems

Security is an important aspect of research in cyber-physical systems (CPS[1]). Though different CPS may have different security requirements and concerns, they still have a lot in common. In this chapter, we use a crucial CPS in our society, the electric power system, as a typical example to address potential attacks in data security of CPS. We believe that our work can be generalized to other CPS and contribute to the research community in this field.

2.1 Introduction

The electric power system is a distribution network that connects the electric power generators to customers through transmission lines, and its security and reliability are critical to society. In order to enable its safe and reliable operation, the power system is monitored continuously by smart meters installed at important locations throughout the power system. The meters take various measurements, including real/reactive power injections on buses and real/reactive power flows on transmission lines. Such data is then fed to

[1] In this dissertation, CPS is treated as both singular and plural.

the control center within the Supervisory Control And Data Acquisition (SCADA) system. Using the collected information, the control center estimates the state variables, which are the voltage amplitudes and phases on buses, and then makes corresponding adjustments to stabilize the power system.

To obtain reliable state estimates, it is essential for the control center to be fed reliable and accurate meter measurements. However attackers may compromise meter measurements and send malicious data to the control center, thus misleading the control center to make bad decisions that may cause severe consequences to the power system. Researchers have developed various techniques to detect bad data measurements [3, 36, 41, 52, 57], most of which are based on measurement residuals.

However, Liu *et al*. [34] has presented an undetectable false data injection that can defeat all the detection techniques based on measurement residuals. Their simulation results indicate that for medium size power system (e.g. IEEE 30-bus system), the attackers may need to compromise 60 to 75% of all meters before they can succeed in launching a random false data injection attack. However, an attacker may either have limited attack resources or only limited access to some meters. Thus, we are interested in exploring if there are other types of attacks that require fewer meters.

In this chapter, we focus on unidentifiable attacks [45, 46], which are different from undetectable attacks discussed in [34]. *In unidentifiable attacks, the control center can detect that there are bad or malicious measurements, but it cannot identify which meters have been compromised*. As a result, the attacker does not need to manipulate as many meters for unidentifiable attacks as when he is launching undetectable attacks. Under an unidentifiable attack, the control center has no way to simply eliminate some ``bad'' data and thus get accurate state estimates. However, the control center has to make a power operation decision, no matter good or bad, in response to the attack. We argue that a good decision during such an attack is one that minimizes the total cost which includes the power generation cost and the damage cost caused by the attack.

Our main contributions in this work are as follows:

1. We are the first to propose the unidentifiable attack in a power system. We demonstrate the feasibility of this type of attacks. An adversary can launch an unidentifiable attack by compromising a smaller number of meters compared with the previously proposed attacks while at the same time confuse the control center on what really happens.

2. We propose a heuristic algorithm to enumerate all feasible cases under an unidentifiable attack. The previous classic ``bad data detection'' algorithms do not work for this attack scenario. Our algorithm is the first to resolve the problems of the previous algorithms. It also significantly reduces the possible solution searching space compared with the brute force approach. We show through empirical study that the algorithm can efficiently find all possible attacks.

3. Enumerating possible attacks is not equivalent to locating the exact attack. To defend against all possible attack scenarios, we also propose a strategy to minimize the average damage to the system. We formulate the problem as a nonlinear programming problem and solve it through a standard optimization package.

4. From the viewpoint of an attacker, he is interested in finding good attack regions to attack such that the number of meters he needs to compromise to launch an unidentifiable attack is minimized. In order to find such regions, we propose a heuristic algorithm that can reveal good attack regions by performing column transformations on the Jacobian matrix of a power system.

5. We propose an algorithm that a control center with limited resource can use to identify the meters they should verify during an unidentifiable attack.

6. We analyze our system in AC model, which is nonlinear and doubles the number of variables compared to the DC model. The recent security investigations of the power system, such as [34, 71, 25, 24], are all based on DC model. Although DC model can be representative of the power system, AC model can capture more

subtleties and is more complicated and realistic to describe a power system. We believe this is the first piece of work to carefully examine the attacks and solutions in realistic AC model. The formulation and optimization can be used as a basis for future work.

The rest of the chapter is organized as follows. Section 2.2 is the related work. In Section 2.3, we demonstrate unidentifiable attacks, and show that how an attacker can find good attack regions in a power system. In Section 2.4, we formulate an optimization framework for the control center, given an unidentifiable attack. In Section 2.5, we propose algorithms to find all feasible attack cases. In Section 2.6, we bring up an interesting problem that how the control center reveals the real attack case if it has some limited resources to verify some meters. We evaluate our proposed enumerating algorithms and optimization strategy in Section 2.7 and conclude this work in Section 2.8. Since many nomenclatures are used in this chapter, they are listed together here for easy reference.

Indices:

i & j:	bus index
k:	feasible case index
g:	generator index

Sets, vectors, and elements:

L:	set of load buses
G:	set of generator buses
A:	set of all meters
z:	the vector of all meter measurements
$\mathbf{z}_a$:	the measurement vector under attack
P:	set of protected meters
D:	set of bad meters
B:	set of all buses, $B = L \cup G$
b_i:	bus i

T: set of transmission lines

t_{i_j}: transmission line between buses i and j

t_{i_*}: transmission lines incident to bus i

Constants:

m: the total number of meters, $m = |A|$

l: the total number of feasible cases

n: the total number of buses, $n = |B|$

r: the capacity of the attacker

s: the number of meters an operator can verify

C_g: the generating cost of generator g

$C_{shed,i}$: the power shedding cost of load bus i

G_{ij}: the conductance between bus i and bus j

B_{ij}: the susceptance between bus i and bus j

$PG_{g,min}$: the minimal real capacity of generator g

$PG_{g,max}$: the maximal real capacity of generator g

$QG_{g,min}$: the minimal reactive capacity of generator g

$QG_{g,max}$: the maximal reactive capacity of generator g

PL_{ij}^{max}: the maximal line capacity between bus i and bus j

$PD_{k,i}$: the real power demand on bus i in case k

$QD_{k,i}$: the reactive power demand on bus i in case k

Variables:

x: the vector of state variables

PG_g: the real power generated by generator g

QG_g: the reactive power generated by generator g

V_i: the voltage amplitude of bus i

θ_i: the voltage phase of bus i

$PS_{k,i}$: the real power shedding of bus i in case k

$QS_{k,i}$: the reactive power shedding of bus i in case k

P_{ij}: the real power flow between bus i and j

Q_{ij}: the reactive power flow between bus i and j

PL_{ij}: the power flow between bus i and j

$D_{shed,k}$: the total real power shedding cost for case k

2.2 Related Work

To ensure the power system operates correctly, the control center needs to collect measurements to estimate the state variables, and then takes control actions against any contingency. For a system with n buses and m meters, the state estimates are determined through the model of $\mathbf{z} = \mathbf{h}(\mathbf{x}) + \mathbf{e}$, where $\mathbf{x} = (V_1, ..., V_n, \theta_1, ..., \theta_n)$ is the state variable vector, $\mathbf{z} = (z_1, ..., z_m)$ is the measurement vector, and $\mathbf{e}$ is the measurement error vector.

Bad measurements, however, may exist due to faulty meters, transmission errors, or alterations by malicious attackers. Researchers have developed lots of approaches on bad data detection and identification since 1970's, such as Identification By Elimination (IBE) [52], Non-Quadratic Criteria (NQC) [36], Hypothesis Testing Identification (HTI) [57], Combinatorial Optimization Identification (COI) [3, 41]. An early comparative study of the first three approaches can be found in [37].

Liu *et al.* [34] has shown that, given the topology and line impedance of a power system in DC model, an attacker can inject malicious data without being detected by the control center, since the injected malicious data does not change the residual. Since then, the undetectable attack has drawn a lot of attention, such as in [25, 71, 72]. In [71, 72], a specific undetectable attack called the load redistribution attack is analyzed. In [25], minimum residue energy attack is discussed as well.

The unidentifiable attack considered in this work is different from the undetectable

attack in that the control center can detect the presence of an attack but cannot identify which meters have been compromised. This is in fact the concept of *nondeducibility* [12] but with an inverse form, in which the attacker maintains the property of nondeducibility. Our unidentifiable attacks aim to confuse the control center to the extent that it does not know what the exact demand scenario is and hence needs to rely on a strategy to deal with such attacks. Compared with undetectable attack, an attacker only needs to manipulate at most half as many meters to launch an unidentifiable attack as those he needs for an undetectable attack. The concept of unidentifiable attacks is hence of great value and more practical, especially for an attacker with limited attack resources. Consequently, this work complements the research in cyber-physical systems [8, 32, 50, 61, 66, 67].

2.3 Unidentifiable Attacks

The unidentifiable attack in this work is a new type of attack in the power system. Before we present the formal definition of the unidentifiable attack, we define *feasible case* for a power system and make an assumption regarding to the capability of an attacker.

Feasible case: A feasible case for a power system is a vector of all m meter readings that (1) is consistent with negligible error; and (2) hence can produce a distinct set of state variables.

We denote the collection of feasible cases by set **F**. And we assume that an attacker can at most compromise r meters. Suppose the meter reading vector under an attack is $\mathbf{z}_a$. Then the formal definition is:

Unidentifiable Attack: An attack is unidentifiable if the following two conditions are satisfied: (1) $d(\mathbf{z}_a, \mathbf{z}) \neq 0$, $\forall \mathbf{z} \in \mathbf{F}$, where d is a function that returns the number of mismatch elements between two vectors; (2) there exists a set $\mathbf{F}' \subseteq \mathbf{F}$ such that $d(\mathbf{z}_a, \mathbf{z}') \leq r$, $\forall \mathbf{z}' \in \mathbf{F}'$, and $|\mathbf{F}'| \geq 2$, where $|.|$ is a function on a set and returns the number of elements in it.

In the above definition, condition (1) makes sure that the control center can detect the

presence of an attack; condition (2) guarantees that at least two feasible cases exist to obfuscate the control center. In the following, we will first further explain the unidentifiable attack with concrete examples. Then, we will discuss the attack strategy from an attacker's viewpoint.

2.3.1 Unidentifiable Attack Examples

To further understand the unidentifiable attack, let us consider an ideal case where the measurements have no error except those which are manipulated by an attacker. Suppose there are $m = m_0 + m_1 + m_2$ measurements which can be divided into three sets, M_0, M_1 and M_2, with cardinalities m_0, m_1 and m_2 respectively. Assume that $m_1 \leq r$, $m_2 \leq r$, and an attacker has manipulated the set of measurements in M_1. Obviously, the vector of measurements is not a feasible case, that is, the control center can detect the presence of an attack. Let us further assume that, the measurements $M_0 \cup M_1$ alone are consistent and make the whole system *observable*[2], so are the measurements $M_0 \cup M_2$. In such a scenario, the control center can conclude that either set M_1 or set M_2 is compromised. However, the control center has no way to determine the exact set that has been compromised. We call such an attack unidentifiable, since the attack on set M_1 confuses the control center to believe that either set M_1 or set M_2 is compromised. We say this attack has two feasible cases: one is determined by $M_0 \cup M_1$, and the other is determined by $M_0 \cup M_2$.

In the power system, all meters are interactive to some extent. Therefore, changing one meter usually requires changes of many other meters in order to make the changes consistent. From the view of an attacker, he intends to change as few meters as possible to generate an unidentifiable attack. In this work, we focus on two types of unidentifiable attacks, which can lead to bad consequences. One is *load redistribution attack* (Type I), in which the attacker obfuscates the control center whether the power demands on some

[2]A set of measurements is said to make the system *observable* if the states of all the buses can be determined with these measurements.

load buses are redistributed, while the total power demand is unchanged. The other is *load increase attack* (Type II), in which the attacker obfuscates the control center whether the demand on a certain bus is increased, while the demands for other load buses remain the same. Compared to random changes on meter readings that may require to check multiple times before an unidentifiable attack with bad consequences can be obtained, these two types of attacks empirically require less effort from the attacker.

Let us consider two simple examples that illustrate the two types of unidentifiable attacks above. To simplify our discussion, we use DC model in our examples, but we consider AC model in the rest of the work. Figure 2.1 is a three bus power system. On each bus, there is a power injection meter; on each transmission line, there are two power flow meters, with one at each end of the line. In DC model, there is no resistance on transmission lines but only susceptance. The susceptance between bus 1 and bus 2 is 280 (we omit the unit, and thereafter); that between bus 2 and bus 3 is 70; and that between bus 1 and bus 3 is 140. Suppose the load on bus 2 is 21, and the load on bus 3 is 35. Before any attack, the meter readings are consistent as shown in Figure 2.1.

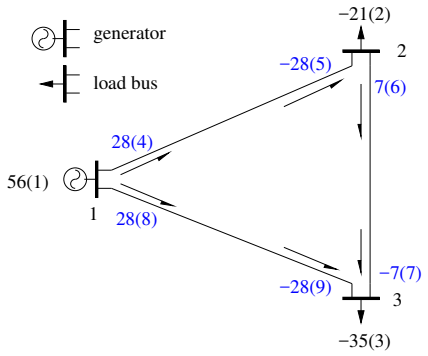

Figure 2.1: The meter readings before attack. XX(Y) means that meter Y's reading is XX. A positive value means a power flow comes out of a bus, while a negative value means a power flow goes into a bus.

Now suppose $r = 4$ and an attacker can manipulate meters {2,3,5,9}. The attacker changes the readings of these four meters to the value shown in Figure 2.2. The whole set of data is not consistent, and the control center knows that an attack is present. How-

ever, the readings on meters {1,4,6,7,8} are consistent, and they can determine a set of state variables, which corresponds to the load vector $\{bus2, bus3\}$=$\{21, 35\}$. The readings on meters {1,2,3,5,9} are also consistent, while they can determine a different set of state variables, which corresponds to the load vector $\{bus2, bus3\}$=$\{14, 42\}$. Under this scenario, even though the control center knows that four meters have been compromised, it has no way to identify which four have been manipulated. The compromised data can either be meters {2,3,5,9}, or meters {4,6,7,8}. That is, there are two feasible cases. One is determined by meters {1,4,6,7,8}, which is $\{56$, -21, -35, 28, -28, 7, -7, 28, -28$\}$. The other is determined by meters {1,2,3,5,9}, which is $\{56$, -14, -42, 24,- 24, 10, -10, 32, -32$\}$. In this example, the net effect of the attack is to have the control center guess whether there is a 7 unit load redistribution between bus 2 and bus 3 ({21,35} to {14,42}). To make this load redistribution undetectable, one has to compromise all nine meters except meter 1, which is beyond the attacker's capability. However, we only need to compromise 4 meters to make this attack unidentifiable.

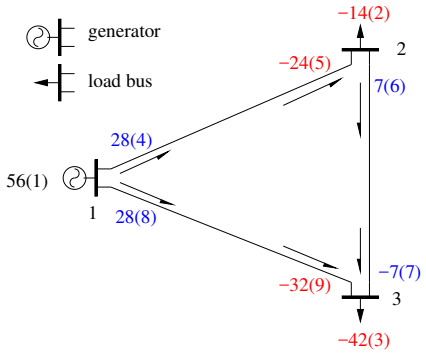

Figure 2.2: Type I unidentifiable attack, where meters 2, 3, 5, 9 are compromised (red ones).

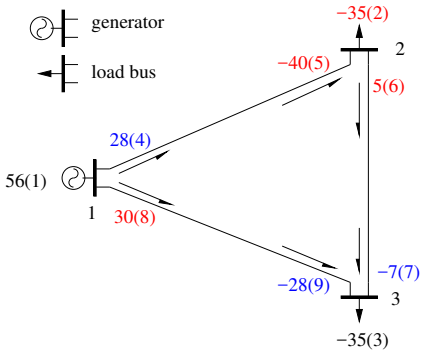

Figure 2.3: Type II unidentifiable attack, where meters 2, 5, 6, 8 are compromised (red ones).

Another similar attack is shown in Figure 2.3. In this case, meters {2,5,6,8} are compromised corresponding to the load vector $\{bus2, bus3\}$=$\{35, 35\}$. Similarly, the compromised data can either be meters {2,5,6,8}, or be meters {1,4,7,9}. That is, there are two feasible cases: one determined by meters {1,3,4,7,9} and the other by meters {2,3,5,6,8}.

In this example, the net effect of the attack is to let the control center guess whether there is a 14 unit load increase on bus 2 (from 21 to 35). Again, only four meters need to be compromised to launch this unidentifiable attack, compared to eight meters for the corresponding undetectable attack.

In each of the example, there are some meters that have the same readings for the different feasible cases of an unidentifiable attack. With more common readings among different feasible cases, fewer meters need to be compromised to launch an unidentifiable attack.

2.3.2 Best Attack Regions for Unidentifiable Attacks

From an attacker's viewpoint, he is interested in finding a region such that he only needs to compromise as few meters as possible to launch an effective attack, no matter Type I or Type II attack.

First, we have to find a metric to evaluate an attack region. Assume that an attacker needs to compromise d meters in a region to shift from one feasible case to another feasible case, i.e., to launch an undetectable attack. Now we want to know how many compromised meters are enough to launch an unidentifiable attack in the same region. We denote it by d'. Regarding r, there are four cases in total, listed in Table 2.1. From

Table 2.1: The number of meters that are enough for an unidentifiable attack (d'), where d is the number of meters that are required to compromise for an undetectable attack.

r (attacker's capability)	d'
$r \geq d-1$	1
$d-1 > r > d/2$	$d-r$
$r = d/2$	$d/2$
$r < d/2$	impossible

the table, we can see that compromising $d/2$ meters is enough to launch an unidentifiable attack, no matter what the valure of r is (since the attacker can compromise $d/2$ meters,

it means r is at least $d/2$). Thus, we use $d' = d/2$ as the metric to evaluate how good an attack region is.

Therefore, we first need to find the regions that require fewest compromised meters for an undetectable attack. In [6], Liu *et al*. proposed a heuristic algorithm for desirable attack region in DC model, where $\mathbf{z} = \mathbf{H}\mathbf{x} + \mathbf{e}$ and **H** is fixed; with column transformation of **H** matrix, they aimed to find a column vector with the greatest number of zero elements. However, in our AC model case, there is no such fixed **H** matrix and the relations between measurements and state variables are non-linear. Instead, we examine the Jacobian matrix $\partial\mathbf{h}/\partial\mathbf{x}$, denoted by **J**. Similar to [6], we apply column transformations on the Jacobian matrix to find a column vector with the greatest number of zero elements, say vector **u**. Then, we figure out the rows with non-zero elements in **u**. Finally the attack region is the region containing meters i such that $u_i \neq 0$. The algorithm is summarized as follows.

Algorithm 2.1: Finding the best attack region

1 calculate **J** at the pre-attack point;
2 **u** =the column vector in **J** with most zeros;
3 **for** *each column vector* $\boldsymbol{j} \in \boldsymbol{J}, \boldsymbol{j} \neq \boldsymbol{u},$ **do**
4 $\quad \mathbf{u}' = \mathbf{u} + c \cdot \mathbf{j}$ s.t. $\|\mathbf{u}'\|_0$ is minimized, where c can be any scalar;
5 $\quad \mathbf{j} = \mathbf{u}'$;
6 **if** $\|\boldsymbol{j}\|_0 < \|\boldsymbol{u}\|_0$, $\boldsymbol{j} \in \boldsymbol{J}, \boldsymbol{j} \neq \boldsymbol{u},$ **then**
7 $\quad$ go to line 2;
8 **else**
9 $\quad$ go to line 10;
10 find the set of meters with indices $\{i | u_i \neq 0\}$; the attack region is the region where these meters are.

Different from **H**, which is fixed, our Jacobian matrix is different at different starting points. Here, we assume a simple starting scenario that all voltages are 1 and all phases are 0, which is a feasible solution to power systems. We examine three IEEE bus systems: 9-bus system, 14-bus system, and 30-bus system. The results are shown in Table 2.2. As we can see, these attack regions usually are the regions with a leaf node or with fewer connections to the rest area of a bus system.

Table 2.2: The best attack regions with the metric d'.

bus system	d'	attack region
9-bus	2	bus 1 and bus 4
	2	bus 2 and bus 8
	2	bus 3 and bus 6
14-bus	2	bus 7 and bus 8
	5	bus 4,7,8 and 9
30-bus	2	bus 9 and bus 11
	2	bus 12 and bus 13
	4	bus 25 and bus 26

2.4 Optimization Strategy

Under an unidentifiable attack, the control center cannot identify which set of meters is compromised, even though it knows that some meters are compromised. Suppose that under an unidentifiable attack, the control center finds l feasible cases, i.e., $|F'| = l$ (we will present algorithms to find all feasible cases in section 2.5). To reduce the damage caused by such an unidentifiable attack, the control center has to consider all l feasible cases, and tries to find a solution such that the damage to the power system is as small as possible before the set of compromised meters can be identified and eliminated (it is possible that the attack cannot be identified without sending power engineers to conduct a physical check).

To evaluate whether a power generation solution is good or not, we need to assess the potential damage such a solution yields to each of l feasible cases. The damage mainly includes the followings:

- Power shedding on load buses. There is not enough power supply for load buses such that some load buses get less power than their demands, which results in the tripping of circuit breakers to shed some loads;
- Overloading of transmission lines. The power flow on a transmission line may go beyond its capacity such that the line trips, possibly resulting in severe consequences,

e.g., large area blackout;

- Overpowering on load buses. A load bus may be fed more power than its demand, which can make the power system operate at higher frequency than it can tolerate, trip certain circuit breakers and cause blackout.

The cost of the whole power system consists of two components: one is related to the cost of power generation, and the other is related to the cost of damages mentioned above. Any good power generation solution should avoid overloading and overpowering scenarios since they both can cause severe consequences. In our strategy that identifies good power generation solutions under an unidentifiable attack, we propose to avoid any potential damage caused by overloading and overpowering by including constraints that prevent overloading and overpowering from occurring. Therefore, we only need to include the power generating cost and the penalty of load shedding in the overall cost. Since all l feasible cases are unidentifiable and equally possible in the view of the control center, it is reasonable to consider the average damage caused by a power generation solution to all feasible attack cases. We are to find a power generation solution such that the sum of the generating cost and the average damage caused by load shedding is minimized. (Please refer to Section 2.1 for nomenclatures.)

$$min: \quad \sum_{b_j \in G} C_j PG_j + \frac{1}{l} \sum_{k=1}^{l} D_{shed,k} \tag{2.1}$$

where $D_{shed,k}$ is defined as follows,

$$D_{shed,k} = \sum_{b_i \in L} C_{shed,i} PS_{k,i} \tag{2.2}$$

The constraints are:

(1) Power shedding constraints:

$$0 \leq PS_{k,i} \leq PD_{k,i}, \forall b_i \in L, 1 \leq k \leq l \tag{2.3}$$

in which the positive $PS_{k,i}$ guarantees that there is no overpowering.

(2) Power flow and power injection constraints:

$$\sum_{j=1}^{n} V_i V_j (G_{ij} cos(\theta_i - \theta_j) + B_{ij} sin(\theta_i - \theta_j)) - PG_i + PD_{k,i} - PS_{k,i} = 0, 1 \leq i, j \leq n, 1 \leq k \leq l \tag{2.4}$$

$$\sum_{j=1}^{n} V_i V_j (G_{ij} sin(\theta_i - \theta_j) - B_{ij} cos(\theta_i - \theta_j)) - QG_i + QD_{k,i} - QS_{k,i} = 0, 1 \leq i, j \leq n, 1 \leq k \leq l \tag{2.5}$$

$$P_{ij} = -V_i^2 G_{ij} + V_i V_j (G_{ij} cos(\theta_i - \theta_j) + B_{ij} sin(\theta_i - \theta_j)), \forall t_{i\_j} \in T \tag{2.6}$$

$$Q_{ij} = V_i^2 B_{ij} + V_i V_j (G_{ij} sin(\theta_i - \theta_j) - B_{ij} cos(\theta_i - \theta_j)), \forall t_{i\_j} \in T \tag{2.7}$$

$$PL_{ij} = \sqrt{P_{ij}^2 + Q_{ij}^2} \tag{2.8}$$

(3) Line transmission capacity constraints:

$$-PL_{ij}^{max} \leq PL_{ij} \leq PL_{ij}^{max}, \forall t_{i\_j} \in T \tag{2.9}$$

(4) Generator capacity constraints:

$$PG_{g,min} \leq PG_g \leq PG_{g,max}, \forall b_g \in G \tag{2.10}$$

$$QG_{g,min} \leq QG_g \leq QG_{g,max}, \forall b_g \in G \tag{2.11}$$

In the above formulation, $PD_{k,i}$ and $QD_{k,i}$, which are determined by the kth feasible case, are known. $PS_{k,i}$ and $QS_{k,i}$ are variables. V_i and θ_i, $1 \leq i \leq n$, are auxiliary

variables, which connect other variables via Eq(2.4), Eq(2.5), Eq(2.6), and Eq(2.7). After solving the minimization problem (we use the free software IPOPT [58]), we will get all the variables, including PG_j, $PS_{k,i}$, V_i and θ_i, $\forall b_j \in G$, $1 \leq k \leq l$, $1 \leq i \leq n$. The control center can then determine the amount of power generation on each generator and the amount of power supply on each load bus, and send these quantities as directives to the corresponding generators and load buses. This is how the control center responds to an unidentifiable attack.

2.5 Analysis of Unidentifiable Attacks

When an unidentifiable attack occurs, the control center first has to detect the presence of an attack. One can use any typical bad data detection scheme proposed by previous work to detect the presence of an attack. Given a power system with n buses and m meters in AC model, measurements $\mathbf{z} = (z_1, ..., z_m)$ are functions of the state variables $\mathbf{x} = (V_1, ..., V_n, \theta_1, ..., \theta_n)$. That is, $\mathbf{z} = \mathbf{h}(\mathbf{x}) + \mathbf{e}$, where $\mathbf{h}(\mathbf{x}) = (h_1(\mathbf{x}), ..., h_m(\mathbf{x}))$, are defined in the following four cases (assume no error, e.g., $\mathbf{e} = \mathbf{0}$):

(i) z is real power injection on bus i:

$$z = \sum_{j=1}^{n} V_i V_j (G_{ij} cos(\theta_i - \theta_j) + B_{ij} sin(\theta_i - \theta_j)) \tag{2.12}$$

(ii) z is reactive power injection on bus i:

$$z = \sum_{j=1}^{n} V_i V_j (G_{ij} sin(\theta_i - \theta_j) - B_{ij} cos(\theta_i - \theta_j)) \tag{2.13}$$

(iii) z is real power flow from bus i to bus j:

$$z = -V_i^2 G_{ij} + V_i V_j (G_{ij} cos(\theta_i - \theta_j) + B_{ij} sin(\theta_i - \theta_j)) \tag{2.14}$$

(iv) z is reactive power flow from bus i to bus j:

$$z = V_i^2 B_{ij} + V_i V_j (G_{ij} sin(\theta_i - \theta_j) - B_{ij} cos(\theta_i - \theta_j)) \tag{2.15}$$

In case of errors or an attack, the detection scheme will compute L_2 norm $||\mathbf{z} - \mathbf{h}(\hat{\mathbf{x}})||_2$, where $\hat{\mathbf{x}}$ is the vector of estimated state variables obtained by a least square estimator. Then the L_2 norm is compared with a predefined threshold τ, and an attack is declared only if $||\mathbf{z} - \mathbf{h}(\hat{\mathbf{x}})||_2 > \tau$.

Next, the control center should discern if the attack is unidentifiable. It can draw such a conclusion by enumerating all feasible cases for the attack. It is an unidentifiable attack if at least two feasible cases exist; otherwise, it is not. In the following, we first make some assumptions and formulate the case enumerating problem. Then, we describe algorithms to enumerate all feasible cases, including a brute-force search algorithm and an empirical method that can speed up the brute-force search. Finally, we analyze the complexity and performance of the proposed algorithms.

2.5.1 Assumption and Problem Formulation

We assume that the presence of bad measurements does not make the whole system unobservable, which excludes the scenario of undetectable attacks. And we also assume that a set of meters, say set P, is protected by the power system operator.

Let set A be the set of all meters and set D be the set of bad meters that the control center deduces. With the above assumptions, the problem of finding all feasible cases under an unidentifiable attack can be formulated as follows:

Enumerate all different sets of D such that:

1. The meters in $A \backslash D$ make the whole power system observable, where $\backslash$ means *excludes*;

2. The meters in $A \backslash D$ are consistent; that is, after solving the state estimation for the power system with meters in $A \backslash D$, the norm of residuals of these meters is zero or less than a predefined value τ.

3. $|D| \leq r$, where r is the maximum number of meters an attacker can compromise;

4. $D \cap P = \emptyset$;

5. Different sets of D results in different state variables[3].

2.5.2 Enumerating Feasible Cases

Given the assumptions and formulation above, our goal is to find all feasible cases that satisfy all the constraints. When r is small, we can use brute-force search to find all feasible cases. However when r is big, the brute-force search becomes very expensive, since its search time grows exponentially with increasing r. However, if one can identify an attack region where the compromised meters are located, then the search space can be reduced and hence the search process can be sped up.

2.5.2.1 Brute-force Search to Enumerate Feasible Cases

When the maximum number of meters that an attacker can compromise, r, is small, we use brute-force search directly. In a brute-force search, every combination that meets all the constraints in the problem formulation is examined. The brute-force search algorithm works as follows.

Algorithm 2.2: Brute-force Search

Input: r, the attacker's capability;
Set A, the set of all meters;
Set P, the protected set.
Output: Set F that contains all feasible sets of D.

```
1 F = ∅;
2 for i=1, r do
3     Check every of (m choose i) bad data combinations, D, except those are supersets of
      any set in F;
4     if D ∩ P = ∅ then
5         if A\D pass the residual test then
6             put the bad data set D into F;
```

[3]This condition is to avoid finding duplicated feasible cases, since it is possible that two different sets of meters produce the same set of state variables.

In the above algorithm, the brute-force search actually does not check every combination, as shown in line 3. It does not check the combinations that have already been covered by previous combinations that are included in set F. If we have already found a feasible case with a set of meter readings D being declared as bad, then we do not need to check any other sets of meter readings D', where $D' \supset D$.

2.5.2.2 Locate Attack Region and Enumerate Feasible Cases

When r is large, the brute-force search approach becomes expensive. As the compromised meters in an unidentifiable attack are typically clustered, we are to identify the attack region and then enumerate feasible cases within the attack region. Such a strategy greatly reduces the search space and hence the search time.

To identify the attack region, we can use existing algorithms based on Identification by Elimination (IBE), such as those discussed in [52]. Though these algorithms cannot exactly identify all the bad data, especially those interacting ones, they can give us some clues about the attack region.

We propose a three-step scheme for enumerating feasible cases under an unidentifiable attack. In our first step, we use the IBE algorithm with the goal not to identify all bad data but to roughly locate the attack region. The IBE algorithm first runs the least square estimator and then deletes the measurement with the largest residual, until the norm of the residuals is less than a pre-defined threshold τ. The IBE algorithm works as follows:

Algorithm 2.3: IBE (*Step 1*)

```
1 D = ∅;
2 A = { all meters };
3 while the norm of residuals of meters in A\D ≥ τ do
4     Put the meter with the largest residual in D;
5     Run state estimation with A\D;
6     Find the meter that has the largest residual;
```

After executing *Step 1*, we identify a set of meters, D. We then check the locations of the meters in set D, and hence roughly deduce where the attack region is. We define the *attack region*, R, which is a subgraph of the whole power system, using the following algorithm:

Algorithm 2.4: Attack Region Identification (*Step 2*)

```
R = ∅;
for meter a ∈ D do
    if a is on bus i then
        R = R ∪ b_i ∪ t_{i_*}
    else if a is on line t_{i_j} then
        R = R ∪ b_i ∪ b_j ∪ t_{i_j}
```

The rationale for the above algorithm is as follows. If a meter is on a bus, its reading (real/reactive) is the summation of all power flows (real/reactive) incident to that bus according to Eq(2.4) and Eq(2.5). If a meter is on a branch, its reading is a function of the state variables of the two end buses according to Eq(2.6) and Eq(2.7). For interacting bad data, the bad data nearby a good one can make the good one has the largest residual. Thus, if a data is eliminated in *Step 1*, it is either because the data is bad or its nearby data is bad. Therefore, in the attack region, we include both the data with largest residual and its neighbors that affect it directly. That is, if a meter on a bus is declared bad, we include both that bus and all the branches incident to that bus into the attack region; if a meter on a branch is declared bad, we include that line and two end buses into the attack region. Figure 2.4 illustrates how the attack region is defined.

Therefore, by looking at the region where the bad data are located, we can roughly identify the attack region. However, we cannot guarantee that the attack region defined above includes all the bad meters, which is summarized in the following claim.

Claim: By eliminating the measurement with the largest residual until the remaining ones are consistent, the attack region defined above is not guaranteed to include all bad data.

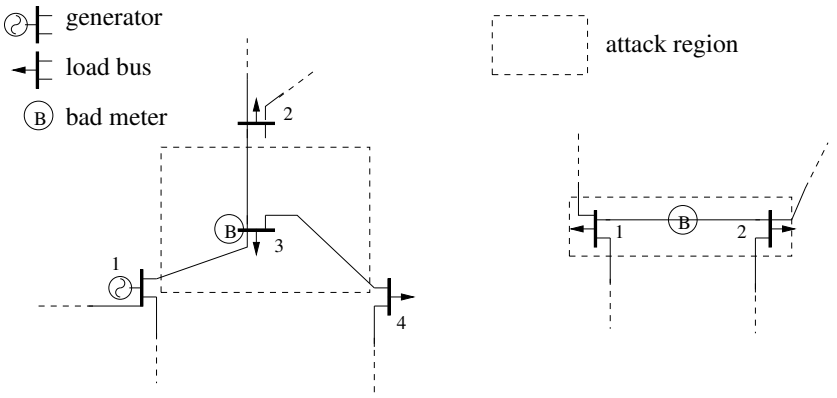

Figure 2.4: Attack region illustration. On the left, a meter on bus 3 is declared bad; on the right, a meter on the branch between bus 1 and bus 2 is declared bad.

Proof: Suppose the whole system has n buses with m measurements, then the system has $2n-1$ state variables. The adversary has modified $m-2n+1$ measurements, and the remaining $2n-1$ measurements are all critical and can make the system observable (and hence satisfy *Assumption 1*). Now the $2n-1$ measurements can give a deterministic solution for the state variables, but any $2n-2$ measurements of the same set cannot. Now let us select $2n-2$ measurements out of the set of $2n-1$ measurements, and denote the remaining one as R. The $2n-2$ measurements yield many feasible solutions of state variables for that particular power system. We select one of the feasible solutions, which is different from the one obtained using the set of $2n-1$ measurements. The adversary then modify the $m-2n+1$ measurements based on this selected feasible solution. Obviously, the largest residual will then occur on the meter that measures R. After eliminating R, the rest of the measurements are consistent. *Step 2* only identifies the attack region as a small neighborhood around R and hence does not include all bad data in the set which contains the $m-2n+1$ readings. □

Remark: The example given in the above proof is an extreme case. Consider a power system in AC model, there are four meters on each transmission line, with two at each end of the line, one for real power flow, the other for reactive power flow; and there are two injection meters on each bus, one for real power and the other for reactive power. To

produce the above attack, the attacker has to compromise more than 2/3 of all meters[4]. If the attacker has that much attack resources, he may be able to launch an undetectable attack which may produce larger damages and hence he may not have the incentive to launch an unidentifiable attack.

After obtaining the attack region, we will do a brute-force search in it. However, this brute-force search algorithm is different from **Algorithm 2.2**, since we need to consider the case that the detected attack region does not include all bad data as proved in the above claim. The algorithm is as follows.

Algorithm 2.5: Brute-force Search in the attack region (*Step 3*)

Input: R, the detected attack region with $|R|$ meters;
r, the attacker's capability;
Set A, the set of all meters;
Set P, the protected set.

Output: A set F that contains all feasible sets of D.

```
1 F = ∅;
2 for i ← 1 to r do
3     for every of (|R| choose i) bad data combination, D, except those are supersets of any
      set in F do
4         if D ∩ P = ∅ then
5             if A\D pass the residual test then
6                 put D into F
7             else
8                 run IBE with A\D and update D by including the data with largest
                  residual until passing residual test;
9                 put D in F if |D| ≤ r;
```

Although the detected attack region may not include all bad data, line 8 in *Step 3* is able to find bad data outside of the detected attack region. The three-step algorithm will usually find all the feasible cases.

[4]For a n bus system with $|T|$ branches, there are $2n + 4|T|$ meters. The attack has to compromise a portion of $\frac{2n+4|T|-(2n-1)}{2n+4|T|} > \frac{2}{3}$, since $|T|$ is usually greater than n.

2.5.3 Performance Analysis

Given a power system with m measurements and an attacker with capability r, the brute-force search (**Algorithm 2.2**) is $O(\binom{m}{r})$. This is huge when m and r are both large. Therefore, **Algorithm 2.2** only works for either a small power system or an attacker with very limited capability.

When **Algorithm 2.2** is not applicable, we should utilize the three-step scheme (**Algorithm 2.3** - **2.5**). Suppose there are $|R|$ meters in the located attack region R, then the complexity is $O(\binom{|R|}{r})$, which is much smaller than that of brute-force search, given that the compromised measurements in an unidentified attack is usually clustered and $|R|$ is much less than m. If the attack region R is not connected, i.e., there are more than one attack regions, we can apply **Algorithm 2.3 - 2.5** on each connected attack region.

Our method is better than all existing bad data detection methods in power system under an unidentifiable attack, since they cannot work in case of such attack. In an unidentifiable attack, there are more than one feasible cases. All existing methods can only find one solution, which means that they can at most find one feasible case. The attacker is always able to manipulate a set of measurements such that the set of bad measurements identified by an existing method is different from the set of manipulated measurements. Therefore, none of them can work in the scenario of an unidentifiable attack. In this sense, our method has already greatly eliminated false positive (FP) and false negative (FN) which all existing methods have. However, since our algorithms are heuristic, they may still have FP and FN. If the detected attack region contains all the bad data, there will be no FP or FN. Even if the detected attack region does not contain all the bad data, **Algorithm 2.5** is able to find some bad data outside of the detected attack region. We believe that these two facts will reduce the rate of FP and FN. As shown in the evaluation in Section 2.7, there is no FP or FN in dealing with the six unidentifiable attacks created with Matpower [77].

2.6 Direct Meter Verification to Eliminate Feasible Cases

In this section, we investigate how a control center with limited resources for meter reading verification should strategize such that it can either identify a particular feasible case as the real attack scenario or reduce the number of feasible cases. By limited resources, we mean the maximum number of meters that the control center can verify, physically or remotely.

Suppose the control center has found l feasible cases given an unidentifiable attack. Now the control center has limited resources to verify only a small number of meters, say s meters. From the perspective of the control center, it is interested in either finding which one of l feasible cases is the real attack case, or excluding as many feasible cases from the l feasible cases. The issue the control center faces is which meters should be chosen for reading verification in order to maximize its interest.

The problem is formulated as follows. Suppose there are m meters in the power system. For each feasible case, we have a set of all meter values; we call them feasible values. Of course, some of these feasible values may be exactly the same as the readings collected from the meters. Therefore, for case k, $k \in [1, l]$, we have a set of feasible values for all meters, $\{a_{k,1}, a_{k,2}, ..., a_{k,m}\}$; for each meter i, $i \in [1, m]$, we have a set of feasible values for l different cases, denoted as $\{a_{1,i}, a_{2,i}, ..., a_{l,i}\}$. Each meter i, $i \in [1, m]$ has a real measured value a_i^*, which can be obtained by checking its reading at its physical location. All these notations are shown in Table 2.3. At least one of $a_{k,i}$, $k \in [1, l]$, is equal to a_i^*.

Given a verified value a_i^*, if $a_{k,i} \neq a_i^*$, then we know that case k is not the real case; that is, by verifying meter i, we can exclude case k from the feasible set of cases; if $a_{k,i} = a_i^*$, feasible case k is possible to be the real case, depending on whether there exists $a_{k',i} = a_i^*$, $k' \neq k$. However, from the perspective of the control center, it cannot know the value a_i^* before physically checking meter i. What the control center knows is just the top part of Table 2.3. For feasible case k and case k', if $a_{k,i} \neq a_{k',i}$, we can

Table 2.3: The notations in meter verifying formulation.

	meter 1	meter 2	meter 3	...	meter m
case 1	$a_{1,1}$	$a_{1,2}$	$a_{1,3}$	...	$a_{1,m}$
case 2	$a_{2,1}$	$a_{2,2}$	$a_{2,3}$	...	$a_{2,m}$
...	...	...	...	...	...
case l	$a_{l,1}$	$a_{l,2}$	$a_{l,3}$	...	$a_{l,m}$
real case	a_1^*	a_2^*	a_3^*	...	a_m^*

guarantee that at least one of them is not the real case by verifying meter i. That is, if $a_{k,i} = a_i^*$, then case k' is not the real case; if $a_{k',i} = a_i^*$, then case k is not the real case; otherwise, both of them are not the real case. However for the case where $a_{k,i} = a_{k',i}$, we still cannot rule out either case k or case k' by verifying meter i, since in the case of $a_{k,i} = a_{k',i} = a_i^*$, case k and k' are still both feasible. Following this reasoning, in order to guarantee that one can find the real case, the control center has to verify a set of meters such that for any pair of case k and case k' there exists a meter index i in the set satisfying $a_{k,i} \neq a_{k',i}$. Thus, the unequal relations, such as $a_{k,i} \neq a_{k',i}$, are the essential information that can help us decide which meters to verify.

For each set of l values of meter i, $\{a_{1,i}, a_{2,i}, ..., a_{l,i}\}$, $i \in [1, m]$, each pair has a relation, which is either "equal" or "unequal". Thus, there are $l' = \binom{l}{2}$ relations. We denote these relations as follows: First, we list the l' relations in the following order: $O_i = [(a_{1,i}, a_{2,i})$, $(a_{1,i}, a_{3,i})$, ..., $(a_{1,i}, a_{l,i})$, $(a_{2,i}, a_{3,i})$, ..., $(a_{2,i}, a_{l,i})$, ..., $(a_{l-1,i}, a_{l,i})]$, and index them by 1, 2, ..., l' sequentially. Second, we define a set S_i for each meter i, $i \in [1, m]$, such that its elements have integer values in the range $[1, ..l']$. S_i is empty in the beginning; j is added into S_i when jth pair in O_i are not equal to each other. **Algorithm 2.6** is the pseudo-code that how S_i is populated. The complexity of **Algorithm 2.6** is $O(ml^2)$.

For example, suppose $l = 4$ and $a_{1,1} = a_{2,1} \neq a_{3,1} = a_{4,1}$ for meter 1, we can get $S_1 = \{2, 3, 4, 5\}$, as illustrated in Table 2.4. By going through this process for all meters, we can get a set $S = \{S_1, S_2, ..., S_m\}$.

Now let $U = \{1, 2, 3, ..., l'\}$, where $l' = \binom{l}{2}$, and C is a set of meter indices. Then we

Algorithm 2.6: *Set S_i for meter i*

```
S_i = ∅;
for j ← 1 to l' do
    if jth pair in O_i are unequal to each other then
        S_i = S_i ∪ j
```

Table 2.4: Set S_1 for meter 1 in case of $l = 4$.

Index	Relation	True/False	Elements in S_1
1	$a_{1,1} \neq a_{2,1}$	F	
2	$a_{1,1} \neq a_{3,1}$	T	2
3	$a_{1,1} \neq a_{4,1}$	T	3
4	$a_{2,1} \neq a_{3,1}$	T	4
5	$a_{2,1} \neq a_{4,1}$	T	5
6	$a_{3,1} \neq a_{4,1}$	F	

can formally formulate our problem for two different cases: (1) with limited resource, the control center can exclude $l-1$ out of l feasible cases (reveal the real case); and (2) with limited resource, the control center cannot exclude $l-1$ out of l feasible cases. The formulations are:

(1) the first case:

min: $|C|$, such that $\{\cup S_i | i \in C\} = U$.

(2) the second case:

max: $|\{\cup S_i | i \in C\}|$, such that: $|C| \leq s$.

Apparently this problem has the same formulation as the set cover problem, except that S_i has some restrictions; for example for $l = 4$, S_i cannot be $\{1, 2\}$, which is a set of impossible relations ($a_{1,i} \neq a_{2,i}$, $a_{1,i} \neq a_{3,i}$, $a_{1,i} = a_{4,i}$, $a_{2,i} = a_{3,i}$, $a_{2,i} = a_{4,i}$, $a_{3,i} = a_{4,i}$). The set cover problem has been well studied in the literature, and we can use a greedy algorithm to solve this problem, which selects the set that covers the greatest number of elements not yet covered each time. Though the formulations are different for the above two cases, we can use the same algorithm, **Algorithm 2.7**, as follows:

Algorithm 2.7: *Finding meters to verify: greedy algorithm*

```
1 X = U;
2 C = ∅;
3 while X ≠ ∅ or |C| < s do
4     select S_i that maximizes |S_i ∩ X|;
5     X = X − S_i;
6     C = C ∪ i;
7 output C;
```

The greedy algorithm has an approximation ratio of $O(ln(l^2))$, and it cannot guarantee an optimal solution. When l and s are very small, we can resort to brute-force search to find the optimal solution, as shown in **Algorithm 2.8**. The complexity of **Algorithm 2.8** is $O(sm^s)$.

Algorithm 2.8: *Finding meters to verify: brute-force search*

```
1 n_0 = 0;
2 for j ← 1 to s do
3     for each different set C' with j indices do
4         calculate n_1 = | ∪ {S_i | i ∈ C'} |;
5         if n_1 = l(l − 1)/2 then go to line 7;
6         if n_1 > n_0 then n_0 = n_1 and C = C';
7 output C.
```

2.7 Evaluation

In this section, we present the results of several experiments that we conduct. First, we generate six unidentifiable attacks in three bus systems. Second, we locate the attack region and enumerate all possible cases using **Algorithm 2.3 - 2.5** presented in Section 2.5; at the same time, we show that the IBE method does not correctly identify the set of bad data, especially if the bad data interact with one another. Finally, we evaluate the operating cost of the power system based on the optimization strategy we present in Section 2.4. Our results show that the optimization strategy we propose for dealing with

unidentifiable attacks is a viable solution.

2.7.1 Generating Unidentifiable Attacks

We generate one Type I attack and one Type II attack in each of 9-bus system, 14-bus system and 30-bus system, whose topologies are shown in Figure 2.5, Figure 2.6 and Figure 2.7 respectively. We generate malicious data using the Matpower tool [77], which is developed to solve power flow problems. Given the topology of a power system, the transmission line characteristics and power loads on buses (load vector), Matpower is able to output the power flow on transmission lines and power injections on buses. We first input one load vector into Matpower and record the first set of meter readings. Then, we feed another load vector into Matpower and record the second set of meter readings. Comparing the two sets of readings, some of them are the same in both sets while others are different. We merge the two sets of readings as follows: for those meter readings that are different in these two cases, we keep some obtained from the first set, and some from the second set. In this way, we can get an unidentifiable attack scenario with two feasible cases.

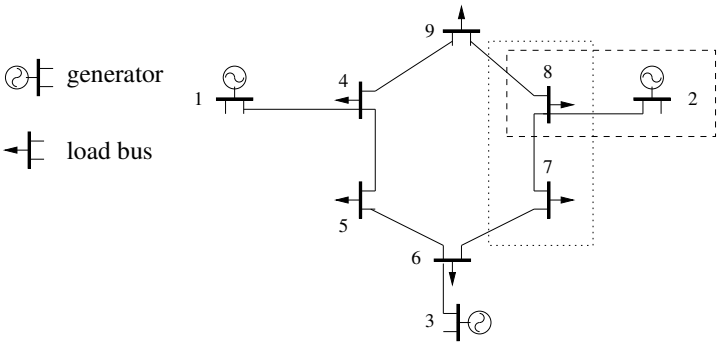

Figure 2.5: The topology of 9-bus system in Matpower.

We still consider the two types of attack scenarios illustrated in Section 2.3 using the AC model. For Type I load redistribution attack scenario, we introduce the second feasible case by increasing the load on one bus by a certain amount and decreasing the load on another bus by the same amount. Thus, together with the original case, we obtain an

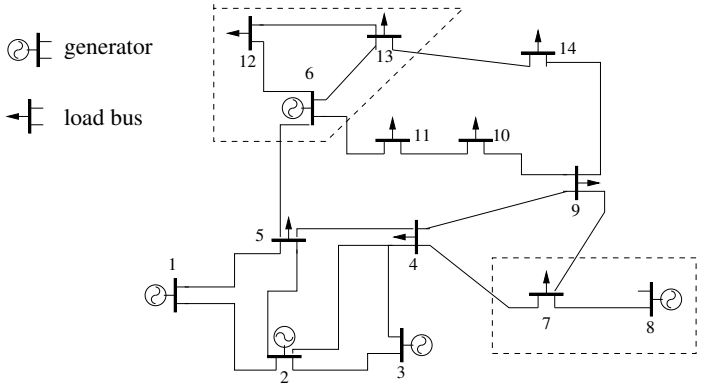

Figure 2.6: The topology of 14-bus system in Matpower.

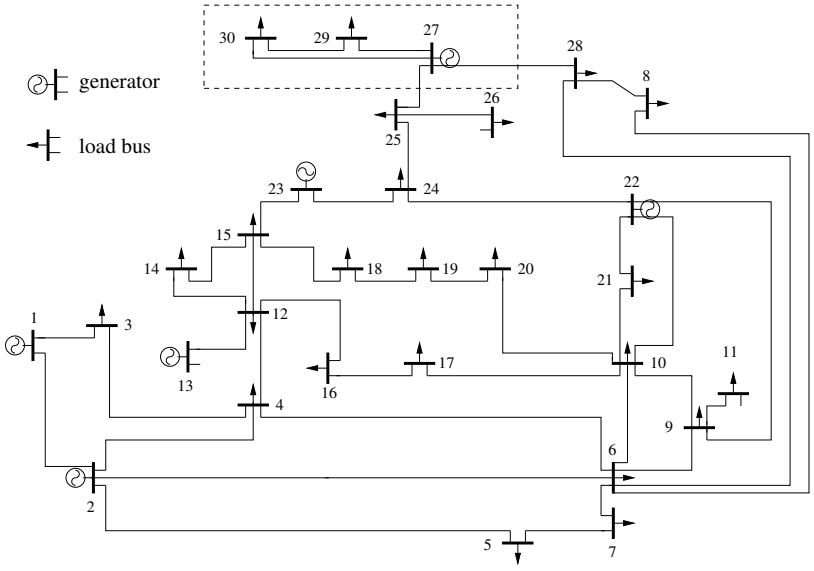

Figure 2.7: The topology of 30-bus system in Matpower.

unidentifiable attack with two feasible cases. For Type II load increase attack scenario, we introduce the second feasible case by increasing the load only on one bus by a certain amount; similarly, we get an unidentifiable attack with two feasible cases.

We first generate two unidentifiable attacks using the 9-bus system in Matpower. For the Type I attack, the compromised meters are listed in Table 2.5, and the rest meters remain intact and their readings are omitted. The meter readings before the attack are based on the real power load vector $\{bus7, bus8\} = \{100, 0\}$, and the meter readings after the attack are based on the real power load vector $\{bus7, bus8\} = \{80, 20\}$; the loads in

other buses have the same values as those in the Matpower distribution package. For the Type II attack, the compromised data are listed in Table 2.6. The meter readings before the attack are obtained when the load of bus 8 is 0, the meter readings after the attack are obtained when the load of bus 8 is 20; the remaining loads are the same as those in the Matpower distribution package.

Table 2.5: Type I attack in 9-bus system. The bold ones are the changed.

Meters	Before attack	After attack (Type I)
PI on bus7	-100	$-$**80**
PL from bus7 to bus8	-75.95	$-$**58.03**
PL from bus8 to bus7	76.51	**58.34**
PL from bus6 to bus7	24.18	**22.09**
QL from bus6 to bus7	22.64	**21.46**

Table 2.6: Type II attack in 9-bus system.

Meters	Before attack	After attack (Type II)
PL from bus2 to bus8	163.0	$-$**183.0**
PL from bus8 to bus2	-163.0	**-183.0**

We then generate two unidentifiable attacks in 14-bus system, one for each type. For the Type I attack, the compromised meters are listed in Table 2.7, and the rest meters remain intact and their readings are omitted. The meter readings before the attack is based on the real power load vector $\{bus12, bus13\} = \{6.1, 13.5\}$, and the meter readings after the attack are based on the real power load vector $\{bus12, bus13\} = \{16.1, 3.5\}$; the loads in other buses have the same values as those in the Matpower distribution package. For the Type II attack, the compromised data are listed in Table 2.8. The meters readings before the attack is based on the real power $bus7 = 0$, and the readings after the attack is based on the real power $bus7 = 10$; the loads in other buses have the same values as those in the Matpower distribution package.

Finally we generate two unidentifiable attacks using the 30-bus system in Matpower.

Table 2.7: Type I attack in 14-bus system.

Meters	Before attack	After attack
PI on bus13	-13.5	$\mathbf{-3.5}$
PL from bus6 to bus12	8.05	**12.32**
QL from bus6 to bus12	3.31	**4.43**
PL from bus13 to bus6	-17.93	$\mathbf{-14.34}$
QL from bus13 to bus6	-9.99	$\mathbf{-9.33}$
PL from bus12 to bus13	1.87	$\mathbf{-3.96}$
QL from bus13 to bus12	-1.53	$\mathbf{-2.41}$

Table 2.8: Type II attack in 14-bus system.

Meters	Before attack	After attack
PL from bus7 to bus8	0	$\mathbf{-10.00}$
PL from bus8 to bus7	0	**10.00**

Both Type I and Type II attacks are shown in Table 2.9. Columns 3 show the changed meters for the Type I attack scenario. The meters in other region remain unchanged. The meter readings before the attack are based on the real power load vector $\{bus29, bus30\} = \{2.4, 10.6\}$, and the meter readings after the attack are based on the real power load vector $\{bus29\_1, bus30\_2\} = \{12.4, 0.6\}$; the loads in other buses have the same values as those in the Matpower distribution package. Column 4 shows the changed meters for the Type II attack scenario. The meter readings before the attack are obtained when the load of bus 30 is 10.6, the meter readings after the attack are obtained when the load of bus 30 is 20.6; the remaining loads are the same as those in the Matpower distribution package.

2.7.2 Locate the Attack Region and Enumerate Feasible Cases

For the six attacks listed above, we first use **Algorithm 2.3** to get the deleted set D. The deleted set for each attack is listed in Table 2.10, where where $bsxx\_1/2$ means PI/QI on bus xx respectively, and $brxx\_1/2/3/4$ means the PL/QL on the from-bus and to-bus of branch xx respectively. In 9-bus power system, $br3 = \{bus5, bus6\}$, $br5 = \{bus6, bus7\}$,

Table 2.9: Type I and II attacks in 30-bus system.

Meters	Before attack	After attack (Type I)	After attack (Type II)
PI on bus27	26.91	26.91	**37.63**
QI on bus27	11.39	11.39	**12.74**
PI on bus29	−2.4	**−12.4**	−2.4
PL from bus27 to bus29	6.17	**9.32**	**10.56**
QL from bus27 to bus29	1.68	1.68	**2.27**
PL from bus29 to bus27	−6.08	**−9.12**	−6.08
PL from bus27 to bus30	7.12	**3.96**	**13.46**
QL from bus27 to bus30	1.67	1.67	**2.45**
PL from bus30 to bus27	−6.95	**−3.91**	−6.95
PL from bus29 to bus30	3.68	**−3.28**	**7.90**
QL from bus29 to bus30	0.61	0.61	**0.88**
PL from bus30 to bus29	−3.65	**3.31**	−3.65

$br6 = \{bus7, bus8\}$, $br7 = \{bus2, bus8\}$, and $br8 = \{bus8, bus9\}$. In 14-bus power system, $br12 = (bus6, bus12)$, $br13 = (bus6, bus13)$, and $br19 = (bus12, bus13)$. And in 30-bus power system, $br37 = (bus27, bus29)$, $br38 = (bus27, bus30)$ and $br39 = (bus29, bus30)$. In order to show the effectiveness of IBE, we also list the real compromised set for each attack.

As we can see from Table 2.10, the IBE method cannot identify the real compromised measurements, and there is even no common element between the deleted set and compromised set. This illustrates that the IBE method cannot identify the interacting bad measurements as mentioned in previous work, such as [36, 57, 3, 15, 41]. Neither can other bad data detection methods, such as NQC, HIT and COI mentioned in Section 4.2, as explained in Section 2.5.3.

Next we apply **Algorithm 2.4** on the deleted set listed in Table 2.10 to get the attack region. Though the deleted sets do not even contain one real compromised measurement, the attack regions obtained from **Algorithm 2.4** do contain all the compromised measurements. The attack regions are shown in the dashed rectangle or trapezoid in Figure 2.5, Figure 2.6 and Figure 2.7; in the 30-bus system, the two attacks have the

Table 2.10: The deleted sets and compromised set for four attacks.

	Attack	Deleted set	Compromised set
9	Type I	$bs8\_1$, $br5\_3$, $br3\_3$ $br3\_1$, $bus5\_1$, $br5\_1$	$bs7\_1$, $br6\_1$, $br6\_3$ $br5\_1$, $br5\_2$
	Type II	$bs2\_1$, $bs8\_1$ $br7\_2$, $br8\_2$	$br7\_1$, $br7\_3$
14	Type I	$bs12\_1$, $br19\_3$, $br12\_3$ $br13\_1$, $br19\_2$, $br13\_2$ $br12\_4$	$bs13\_1$, $br12\_1$, $br12\_2$ $br13\_3$, $br13\_4$, $br19\_1$ $br19\_4$
	Tpye II	$bus7\_1$, $bus8\_1$	$br14\_1$, $br14\_3$
30	Type I	$bs30\_1$, $br38\_2$, $br37\_2$ $br38\_4$, $br37\_4$, $br39\_2$ $br39\_4$	$bs29\_1$, $br37\_1$, $br37\_3$ $br38\_1$, $br38\_3$, $br39\_1$ $br39\_3$
	Type II	$bs30\_1$, $br38\_3$, $br37\_3$ $br39\_3$, $br37\_4$, $br39\_4$ $br38\_4$	$bs27\_1$, $bs27\_2$, $br37\_1$ $br37\_2$, $br38\_1$, $br38\_2$ $br39\_2$

same attack region.

Finally, we apply **Algorithm 2.5** directly to enumerate all feasible cases. For all six unidentifiable attacks, we are able to find out that there are only two feasible cases for each attack, just as same as described in Section 2.7.1. Furthermore, we can tell the attack type of each attack after obtaining its feasible cases. Let us take the Type II attack in 14-bus system as an example to show the effectiveness of **Algorithm 2.5**. In the 14-bus system, there are 14 buses and 20 branches. As we assume 4 meters on each branch and 2 meters on each bus, there are 108 meters in total. To calculate the time complexity of the enumerating algorithms in Section 2.5, let us further assume that the attacker can at most compromise 8 meters and there is no protected meter in the power system ($P = \emptyset$). For the brute-force search algorithm, the search space of $\sum_{i=1}^{8} \binom{108}{i}$, is still huge, not to mention all the computations required for state estimation and residual checking. While in the attack region, there are only 16 meters. By localizing the attack region first, the search space is greatly reduced to at most $\sum_{i=1}^{8} \binom{16}{i}$. Actually, the search space is even far smaller than $\sum_{i=1}^{8} \binom{16}{i}$ for two reasons. The first is that we have already found one feasible

case via the IBE method. The second is, once a feasible case is found, the brute force search can skip some combinations. For instance, if a solution with 3 bad data has been identified, we do not need to check all bad data combinations which include those 3 bad data.

2.7.3 Optimization on the Cost

We evaluate our optimization problem using the six unidentifiable attacks we discussed in Section 2.7.1. For each of the unidentifiable attack, we have already known that there are two feasible cases and what they are. Thus, we only need to feed these feasible cases into the objective function Eq(2.1) and try to minimize it. We use the free software IPOPT [58] to solve the nonlinear optimization problem. In our analysis, we set the power shedding cost as five times as the cost of the most expensive generator. This setting is reasonable, since the power shedding cost must be higher than the cost of any generator; otherwise, the generator will choose not to satisfy the load demand even it still has available capacity.

2.7.3.1 Type I Attack in 9-bus System

In this attack, we change five meters as shown in Table 2.5. Under this unidentifiable attack, the control center may either conclude that the power demands of bus 7 and bus 8 are 100 and 0 (case 1), or they are 80 and 20 (case 2). These two load vectors are fed together with the constraints into IPOPT to determine the optimal state variables, the voltage and phase on each bus, which can minimize the total cost. In the original Matpower packet, all line capacities are 250 MVA. In order to examine the impact of line capacities, we adjust the line capacity of line 6 (between bus 7 and bu 8) to 60 MVA. The cost comparison is listed in Table 2.13, in which solution 1 is the optimal solution based on case 1, and solution 2 is the optimal solution based on case 2. ``Over-load'' means that if the control center gets a solution based on case 1 but it is actually case 2, then some branches will exceed their line capacities. As we can see, our solution is better

than solution 2. However, our solution is comparable to Solution 1. The reason is that even we limit the line capacity between bus 7 and bus 8, the power can go from generator 2 to bus 7 by circumventing the line between bus 7 and bus 8.

Table 2.11: The cost comparison for type I attack in 9-bus system.

	If case 1	If case 2	Average
Solution 1	5316.9	5316.9	5316.9
Solution 2	Over-loaded	5306.0	NA
Our solution	5317.0	5315.5	5316.2

2.7.3.2 Type II Attack in 9-bus System

Table 2.6 shows type I attack in 14-bus system. The two feasible cases are: the real power demand on bus 7 is either 0 (case 1) or 20 (case 2). In this example, we do not change any line capacity. The cost comparison is listed in Table 2.12, where ``Over-powered'' means that if the control center gets a solution based on case 2 but it is actually case 1, then some buses will get more power than their demands. As we can see, our solution is still the best, given that the control center cannot favor one case over the other.

Table 2.12: The cost comparison for type II attack in 9-bus system.

	If case 3	If case 4	Average
Solution 3	5310.0	6055.0	5682.5
Solution 4	Over-powered	5805.2	NA
Our solution	5310.1	5863.4	5586.7

2.7.3.3 Type I Attack in 14-bus System

In this attack, we change 7 meters as shown in Table 2.7. Under this unidentifiable attack, the control center may either conclude that the power demands of bus 12 and bus 13 are 6.1 and 13.5 (case 1), or they are 16.1 and 3.5 (case 2). In the original Matpower packet,

all line capacities are 9900 MVA. In order to examine the impact of line capacities, we adjust the line capacities for the following branches: branch 12, branch 13, and branch 19 to 10 MVA, 25 MVA and 10 MVA respectively. The cost comparison is listed in Table 2.13. As we can see, our solution is the best, given that the control center cannot favor one case over the other.

Table 2.13: The cost comparison for type I attack in 14-bus system.

	If case 1	If case 2	Average
Solution 1	8083	Over-loaded	NA
Solution 2	8594	8594	8594
Our solution	8573	8595	8584

2.7.3.4 Type II Attack in 14-bus System

Table 2.8 shows type I attack in 14-bus system. The two feasible cases are: the real power demand on bus 7 is either 0 (case 1) or 10 (case 2). In this example, we do not change any line capacity. The cost comparison is listed in Table 2.14. As we can see, our solution is still the best, given that the control center cannot favor one case over the other.

Table 2.14: The cost comparison for type II attack in 14-bus system.

	If case 1	If case 2	Average
Solution 1	8083	10208	9146
Solution 2	Over-powered	8486	NA
Our solution	8087	10081	9084

2.7.3.5 Two Attacks in 30-bus System

The evaluation for the two attack in 30-bus system is similar to that in 9-bus system and 14-bus system. Here we omit the details but only keep the main results. In the type I

attack, the two feasible cases are: the power demands of bus 29 and 30 are 2.4 and 10.6 (case 1), or they are 12.4 and 0.6 (case 2). And we adjust the line capacities for the following branches: branch 37, branch 38, and branch 39 from the original value of 16 MVA to 4 MVA, 8 MVA and 3 MVA respectively. The cost comparison is listed in Table 2.15. In the type II attack, the two feasible cases are: the real power demand on bus 30 is either 10.6 (case 1) or 20.6 (case 2), and the cost comparison is shown in Table 2.16.

Table 2.15: The cost comparison for type I attack in 30-bus system.

	If case 1	If case 2	Average
Solution 1	635.0	Over-loaded	NA
Solution 2	693.1	693.1	693.1
Our solution	680.3	693.7	687.0

Table 2.16: The cost comparison for type II attack in 30-bus system.

	If case 1	If case 2	Average
Solution 1	581.2	775.0	678.1
Solution 2	Over-powered	623.6	NA
Our solution	581.3	750.7	666.0

Again, we can see that our solutions is the best on average among all the solutions, which shows that our optimization strategy is indeed viable and effective.

2.7.3.6 Discussion

In the six examples above, our solutions are the best compared to the solutions that are optimal only to one of several feasible cases. For Type II load increase attack, our solution is always much better. For Type I load redistribution attack, if the line capacities are not limited or only a few are limited, the gain of our solution is small, such as ``Our solution'' compared to ``Solution 1'' in Table 2.11. However, if more line capacities are limited, the gain of our solution becomes big, such as the other two Type I attacks.

2.8 Conclusion

Cyber-physical systems (CPS) face various attacks and demand sophisticated countermeasures. In this chapter, we investigate the electric power system, a typical and crucial CPS in modern society. We point out that the electric power system is vulnerable to potential cyber attacks, and introduce the concept of unidentifiable attacks, which has never been proposed before. In such an attack, the control center cannot obtain a deterministic state estimation; there may be several feasible cases, and the control center cannot simply favor one over the others. We also discuss a strategy that an attacker can use to identify good attack regions where he can choose meters to compromise for an identifiable attack. We then formulate an optimization strategy from the perspective of the control center to deal with an unidentifiable attack such that the average damage caused by the attack can be minimized. Furthermore, we propose a three-step scheme that allows us to find all feasible cases under an unidentifiable attack, in which we locate attack region first and hence significantly reduce the search space when compared to the search space using the brute-force search scheme directly. We also discuss a strategy that the control center can use to reveal the real case if it has some limited resources to verify some meters. Finally, we evaluate and validate our optimization strategy and enumerating scheme using three power systems.

Chapter 3

Defending Against Cooperative Attacks in Cooperative Spectrum Sensing

When addressing the security of cyber-physical systems (CPS), we have to consider the communication networks in CPS as well. Communication networks play an important role in CPS, transmitting data among different entities in CPS. As a result, from the perspective of security, communication networks are as important as the entire security of CPS itself. Following the previous chapter, which addresses security of CPS itself, this chapter investigates security of communication networks used in CPS. Again, we use typical communication networks commonly found in CPS, i.e., the cognitive radio network, as an example to show potential threats and corresponding countermeasures.

3.1 Introduction

With the recent rapid growth of wireless services, spectrum space is becoming increasingly crowded. Within the traditional spectrum regulation framework, spectrum bands are exclusively licensed to specific services (primary users), and no violation from unlicensed

services (secondary users) is allowed. As a result, the licensed spectrum is greatly under-utilized temporally and spatially. To solve this dilemma, cognitive radio has emerged to enable secondary users to opportunistically share the licensed spectrum, with the help of spectrum sensing methods [20, 40, 65, 73, 76].

Among the spectrum sensing methods, cooperative spectrum sensing [1, 14, 39] has shown good performance in improving the accuracy of primary user detection. In cooperative spectrum sensing, a set of secondary users is selected to share the sensing results with each other to make a cooperative decision on the presence/absence of the primary user. However, this opens a window for malicious users and attackers, who may remotely or physically capture and control some secondary users and manipulate their sensing reports simultaneously. As a result, such sensing reports are not trustworthy, nor is the primary user detection decision.

A key challenge is identifying compromised sensing reports under an attack and making a detection decision only with uncompromised sensing reports. Min *et al.* [38] has proposed a novel framework called *IRIS* to pinpoint compromised sensing reports by iteratively utilizing the *largest normalized residual method* [18]. *IRIS* works well for most of attack scenarios. In this chapter, we introduce an attack model in which an attacker injects some self-consistent but false data simultaneously. Under this specific kind of attacks, a good measurement, instead of a compromised one, may have the largest normalized residual. As a consequence, the *largest normalized residual method*, as well as *IRIS*, may eliminate good measurements while keeping compromised ones. Thus, we argue that *IRIS* may not work well under this kind of attacks, and there is still room for improvement. We hence propose a modified *COI* (combinatorial optimization identification) [47] to complement *IRIS*.

Overall, this work makes the following contributions:

- We design an attack model, called *cooperative attack*, in which an attacker injects self-consistent false data to multiple sensors simultaneously.

- We propose a theorem that the center node of a cognitive radio network may face uncertainties under a cooperative attack, especially in the case when a large portion of sensors are compromised.
- We propose a modified *COI* algorithm to deal with cooperative attacks. Our algorithm is a good scheme to complement *IRIS* for cooperative attacks, and can be flexibly adjusted to fulfill the detection delay requirement.
- We intensively evaluate our algorithm through simulation, with the results validating its performance.

The rest of chapter is organized as follows. Section 3.2 is related work. Section 3.3 describes the system model, the attack model and the problem formulation. In Section 3.4, we propose our solution. We conduct simulations in Section 3.5 and conclude this work in Section 3.6.

3.2 Related Work

Cooperative spectrum sensing has emerged in recent years to improve spectrum sensing accuracy. However, it suffers from various security vulnerabilities. In cooperative spectrum sensing, multiple secondary users can easily alter the detection decision by reporting unreliable or compromised sensing results simultaneously. Thus, it is essential to designing robust cooperative sensing schemes to defend against malicious users, and much effort has already been made in recent years, such as [7, 9, 10, 22, 31, 33, 70, 75]. In [7], a reputation-based mechanism is proposed to address data falsification, in which reputations of users are used to weigh their sensing reports. In [9], the authors convert the area of interest into a grid of square cells and use it to identify and discard the outlier measurements. In [10], an approach based on machine learning is proposed, in which an initial trusted set of signal data is used to build a classifier which is subsequently used to

detect integrity violations. In [22], an outlier detection scheme is proposed to filter out extreme values in the sensing data. In [31], an approach based on abnormality detection in data mining is proposed. In [33], the authors present an abnormality detection scheme by using the path-loss exponent in signal propagation. In [70], a consensus-based scheme is proposed to defend against data falsification, in which the cooperative decision is gradually reached in a distributed manner. In [75], the authors propose to use total error probability to evaluate spectrum sensing accuracy and also propose a weighted sensing framework. Public-key schemes, such as [59, 60], can also be implemented to defend against malicious users.

As mentioned previously, Min *et al.* [38] have developed an attack detection framework in cooperative spectrum sensing, called *IRIS*. They introduce system state estimation to determine the presence/absence of a primary user, in which the system state variables are the transmission power of the primary transmitter and the path-loss exponent. Their insight is that the sensing reports are governed by the network topology and the signal propagation principle. In *IRIS*, they utilize the *largest normalized residual method* [18] to delete abnormal sensing reports iteratively until the norm of measurement residuals drops below a detection threshold or the number of remaining reports hits a protection threshold. Though *IRIS* works well for most of attack scenarios, we show in this work that it may not work well against *cooperative attacks*, and we propose a modified *COI* to complement *IRIS*.

3.3 Problem Formulation

In this section, we will first present a centralized cooperative spectrum sensing system. Then we will describe the attack model and formulate our problem.

3.3.1 System Model

In a cognitive radio network, both primary users and secondary users coexist in the same geographical area. Since secondary users are allowed to opportunistically access the licensed spectrum originally allocated to primary users, methods to detect the presence/absence of primary users are required. Cooperative spectrum sensing is a robust scheme, in which several secondary users cooperatively determine whether a primary user is using its licensed spectrum or not. In this work, we consider a centralized model. As shown in Figure 3.1, there is a primary user, N secondary users (sensors), and a center node. During each sensing period, sensors measure primary user signals and then send them to the center node using a dedicated control channel. Then the center node will determine the presence/absence of the primary user based on the collected data and broadcast the final decision to all sensors.

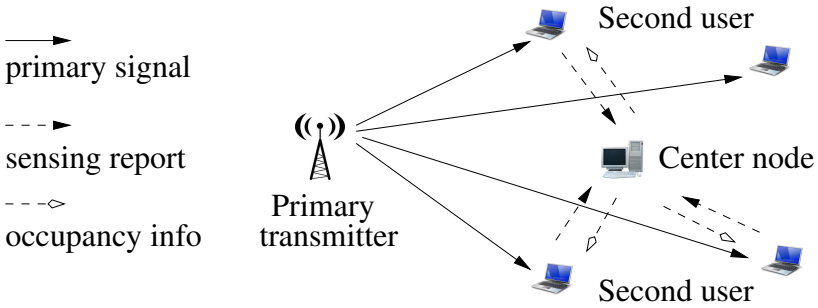

Figure 3.1: The centralized model of cooperative spectrum sensing.

According to the path-loss signal propagation model, the received signal strength of the primary user at a cooperative sensor i, P_i, can be modeled as follows:

$$P_i = P_0 + \alpha 10log_{10}(d_0/d_i) + \epsilon_i,\ i \in [1, N] \tag{3.1}$$

where P_0 is the received power at the reference distance d_0, α is the path-loss exponent, d_i is the distance between the primary user and sensor i, and ϵ_i is the measurement error of sensor i.

3.3.2 Attack Model: Cooperative Attacks

We assume that:

- The attacker can compromise some of the sensors in the network.
- Once a sensor is compromised, the attacker can read and manipulate its sensing report arbitrarily.
- The attacker can manipulate compromised sensors' reports simultaneously.
- The attacker knows a portion of the network topology, i.e., the location of the primary user and those of the sensors he has compromised.

During an attack, the attacker aims to mislead the control node to make a wrong decision. The attacker first compromises some sensors and reads their sensing reports. Based on these sensing reports, the attacker can conclude whether or not the primary user is present. Then the attacker injects false sensing reports into these compromised sensors simultaneously. The attacker's ultimate goal is to mislead the control node into making a decision totally opposite to the real scenario. Specifically, when the primary user is present, the attacker will inject false sensing reports as if the primary user is absent; when the primary is absent, the attacker will inject false data as if the primary user is present. To maximize the attack effect, the attacker will inject self-consistent false sensing reports; that is, all injected data are based on a single power level which is different from the real value. We name this type of attacks *cooperative attacks*. For a cooperative attack, the control center may be easy to detect its presence, but may be difficult to figure out what it really is.

3.3.3 Problem Formulation

We denote the set of all N sensors in a cognitive radio network by $\mathbf{S} = \{S_1, S2, ..., S_N\}$. During a sensing period, we assume that the sensing reports are $\mathbf{P} = [P_1, P_2, ..., P_N]^T$

before any attack, where T is the matrix transpose operator. Now an attacker has compromise a set of N_{com} sensors, denoted by $\mathbf{S}'$, where $\mathbf{S}' \subset \mathbf{S}$. Without loss of generality, we assume that the compromised sensors are the first N_{com} sensors in $\mathbf{S}$ to simplify the description. Based on the sensing reports of $[P_1, P_2, ..., P_{N_{com}}]^T$, the attacker conducts a linear fitting according to Eq (3.1) and obtains the primary transmission power P_0, and the path-loss exponent α. Then the attacker will inject false data into the N_{com} compromised sensors as if the two values are P_0' and α', where P_0' tells a scenario opposite to the real one and α' can be any normal value. Suppose that the compromised sensing reports are $\mathbf{P}' = [P_1', P_2', ..., P_{N_{com}}']^T$. The remaining sensing reports, $\mathbf{P}_r = [P_{N_{com}+1}, ..., P_N]^T$, are intact. Then on the control node side, the collected sensing reports are $\mathbf{P}_a = \mathbf{P}' \cup \mathbf{P}_r = [P_1', P_2', ..., P_{N_{com}}', P_{N_{com}+1}, ..., P_N]^T$. Our problem, illustrated in Figure 3.2, is to obtain the real transmission power P_0 from $\mathbf{P}_a$. At the same time, it will be a plus if all compromised sensors can be identified.

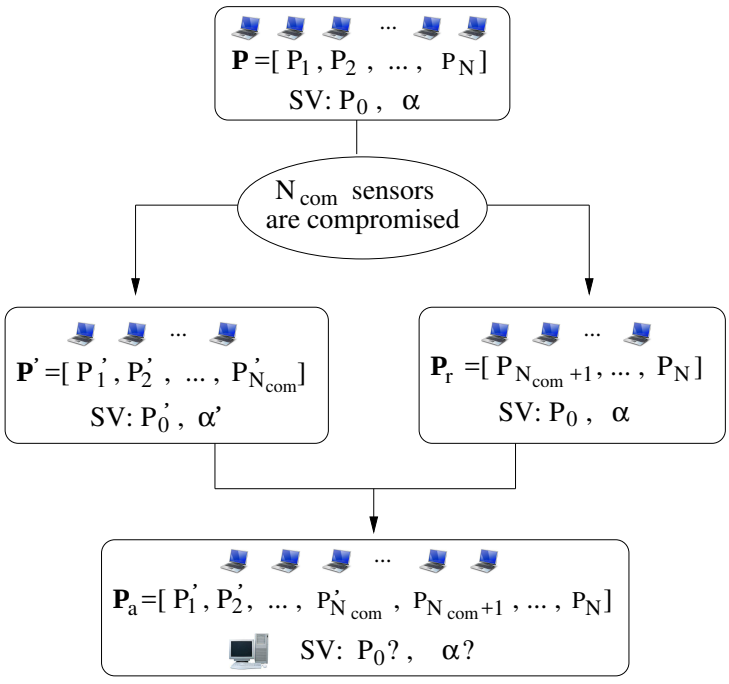

Figure 3.2: Attack model and problem formulation. SV stands for state variables.

3.4 Bad Data Identification

In this section, we will propose our solution to the problem formulated in the previous section. Since our algorithm is to complement *IRIS*, we will first review *IRIS* and point out the limitation of *IRIS*. Then we will present a theorem that the center node may face uncertainties under a cooperative attack. Finally we will present our algorithm in detail.

3.4.1 IRIS Algorithm

In [38], the authors introduce state estimation that provides a useful approach to detect the presence of attacks and further pinpoint the compromised sensors. In state estimation, the state variables are P_0 and α in Eq(3.1), denoted by vector **x**:

$$\mathbf{x} \triangleq [P_0 \;\; \alpha]^T \tag{3.2}$$

Note that $\mathbf{P} = [P_1, P_2, ..., P_N]^T$ is the vector of the primary transmitter's signal strength received by N sensors. Then Eq(3.1) for N sensors can be expressed in matrix notation as follows:

$$\mathbf{P} = \mathbf{H}\mathbf{x} + \boldsymbol{\epsilon} \tag{3.3}$$

where

$$\mathbf{H} \triangleq \begin{bmatrix} 1 & 10log_{10}(d_0/d_1) \\ \vdots & \vdots \\ 1 & 10log_{10}(d_0/d_i) \\ \vdots & \vdots \\ 1 & 10log_{10}(d_0/d_N) \end{bmatrix}_{N\times 2} \tag{3.4}$$

and $\boldsymbol{\epsilon}$ is the vector of measurement errors:

$$\boldsymbol{\epsilon} = [\epsilon_1, \epsilon_2, ..., \epsilon_N]^T \tag{3.5}$$

Three statistical estimation criteria are commonly used to obtain the state estimator **x**: the maximum likelihood criterion, the weighted least-square criterion, and the min-

imum variance criterion. When measurement errors follow the normal distributed with zero mean, i.e., $\epsilon_i \sim N(0, \sigma_i^2)$, these criteria result in the same solution:

$$\hat{\mathbf{x}} = (\mathbf{H}^T\mathbf{W}\mathbf{H})^{-1}\mathbf{H}^T\mathbf{W}\mathbf{P} \tag{3.6}$$

where $\mathbf{W}$ is a diagonal matrix whose elements are σ_i^{-2}, $1 \leq i \leq N$:

$$\mathbf{W} = \begin{bmatrix} \sigma_1^{-2} & & & \\ & \sigma_2^{-2} & & \\ & & \cdots & \\ & & & \sigma_N^{-2} \end{bmatrix}_{N \times N} \tag{3.7}$$

After obtaining state estimator $\hat{\mathbf{x}}$, the center node will first examine whether there is an attack by checking the norm of normalized residuals. The normalized residuals are defined as follows:

$$\mathbf{r}_N = \mathbf{D}^{-1/2}\mathbf{r} \tag{3.8}$$

where

$$D = diag(\mathbf{W}^{-1} - \mathbf{H}(\mathbf{H}^T\mathbf{W}\mathbf{H})^{-1}\mathbf{H}^T) \tag{3.9}$$

$$\mathbf{r} = \mathbf{P} - \mathbf{H}\hat{\mathbf{x}} \tag{3.10}$$

If the L_2 norm $||\mathbf{r}_N||$ is greater than a pre-defined threshold η, the center node will know that some data has been compromised. *IRIS* adopts the *largest normalized residual method* [18] to pinpoint and eliminate the compromised data. It eliminates the data with the largest normalized residual iteratively until the norm of normalized residuals of the remaining data is less than the threshold η or the number of remaining sensors hits a pre-defined threshold N_{min}.

Under a cooperative attack, a good measurement, however, may have the largest normalized residual. The *largest normalized residual method*, as well as *IRIS*, may eliminate good measurements while keeping compromised ones. The ultimate consequence

is that the center node may get an incorrect power level for the primary transmitter and make a decision inconsistent with the reality.

Here we give a counter-example to support our arguments. Suppose there are N = 7 cooperative sensors. To make it simple here, we assume there is no measurement error (we do consider measurement errors in the following sections), and we also omit the units. Let d_0 = 1, and **d** = $[2, 3, 3, 4, 4, 5, 6]^T$ [5]. Just as in [38], here we adopt the threshold-based rule, in which the determination of whether the primary user is active is based on two thresholds, P_h and P_l. When $P_0 > P_h$, the primary user is active. When $P_0 < P_l$, the primary user is idle. When $P_l \leq P_0 \leq P_h$, secondary users can partially use the licensed spectrum with transmission power control. Let us assume P_h = 4 and P_l = 3. Suppose before any attack, the measurements are **P** = $[-7.04, -14.08, -14.08, -19.08, -19.08, -22.96, -26.13]^T$. By Eq(3.6), we can get $\hat{\mathbf{x}} = [P_0 \ \ \alpha]^T = [5, 4]^T$ (here **W** is the identity matrix). Therefore, P_0 is greater than P_h, and the primary user is active. Now suppose an attacker has compromised the first four sensors, and he manipulates the measurements on these sensors as if the primary user is idle. After the attack, the measurements are $\mathbf{P}_a$ = $[-10.04, -17.08, -17.08, -22.08, -19.08, -22.96, -26.13]^T$, with the last three measurements intact. Initially, this data cannot pass the residual test; that is, the norm of normalized residuals is greater than the threshold η. By applying *IRIS*, the measurements from sensors 5, 6, and 7 are eliminated successively. After eliminating these three measurements, the final state vector is $\hat{\mathbf{x}}' = [P_0' \ \ \alpha']^T = [2, 4]^T$. Then the center node will conclude that the primary user is idle since P_0' is less than P_l, which is opposite to the reality.

3.4.2 Uncertainty Under Cooperative Attacks

Here we take a deep look at the counter example in Section 3.4.1, which is a cooperative attack. We can divide the seven measurements into two sets, the first four as set A

[5] Same d_i does not necessarily mean that the two sensors are at the same location, since d_i is the distance between sensor i and the primary transmitter.

and the last three as set B, and set A is compromised by the attacker. After the attack, set A alone can determine a set of state variables $\hat{\mathbf{x}}_A = [2, 4]^T$; we say set A alone is consistent. Set B can determine another set of state variables $\hat{\mathbf{x}}_B = [5, 4]^T$, which is the same as the original; we also say set B is consistent. That is, the attacker actually mixes two feasible state vectors[6] together. At the same time, the attacker makes *IRIS* eliminate the measurements in set B first rather than in set A. As a consequence, it misleads the center node to make a completely incorrect decision. Since both set A and set B are two feasible sets of measurements, by no means can we favor one set over the other. However, if we know the attacker's capability, we may be able to make a decision. For instance, if we assume that the attacker can at most compromise three measurements, then we know that set B must be compromised. While if we assume that the attacker can compromise four measurements, we still do not know which set is compromised. This raises the question, how many measurements must the attacker compromise to cause such uncertainty? We answer this question by the following theorem.

Theorem 1: For a cooperative sensing system with N sensors that do not have measurement errors, there exists a critical number $N_c = \lfloor N/2 \rfloor - 1$ such that: (1) if the attacker can at most compromise N_c sensors and all d_i are different, then there is only one feasible state vector; (2) otherwise, there may be more than one feasible state vector.

Proof: Since a state vector only has two variables, P_0 and α, any two measurements whose channel gain matrix $\mathbf{H}_{2\times 2}$ has full rank can determine a state vector. Since all d_i are different, any two measurements can determine a state vector. Thus, any two feasible sets of measurements that result in two different state vectors have at most one measurement in common. Suppose N is even (when N is odd, the proof is similar), and $N = 2f + 2$. Now the attacker has compromised $N_c = f$ measurements. There are two cases in total:

(1) These f measurements are consistent and can perfectly determine a state vector

[6]Here by feasible state vector, we mean a state vector $[P_0, \alpha]$ that is obtained from a subset of total measurements. The subset, with its cardinality maximized, is called a set of feasible measurements accordingly.

different from the original one. Thus, there are two feasible sets of measurements in total. If these two sets have one measurement in common, as shown in the top of Figure 3.3, then for the original feasible set, the center node will conclude that there are f compromised measurements; for the compromised feasible set, the center node will conclude that there are $f + 1$ compromised measurements. If these two have no measurement in common, as shown in the middle of Figure 3.3, then for the original feasible set, the center node will still conclude that there are f compromised measurements; for the compromised feasible set, the center node will conclude that there are $f + 2$ compromised measurements. Since the attacker can at most compromise f measurements, in either case, the center node can deterministically identify the set of compromised measurements.

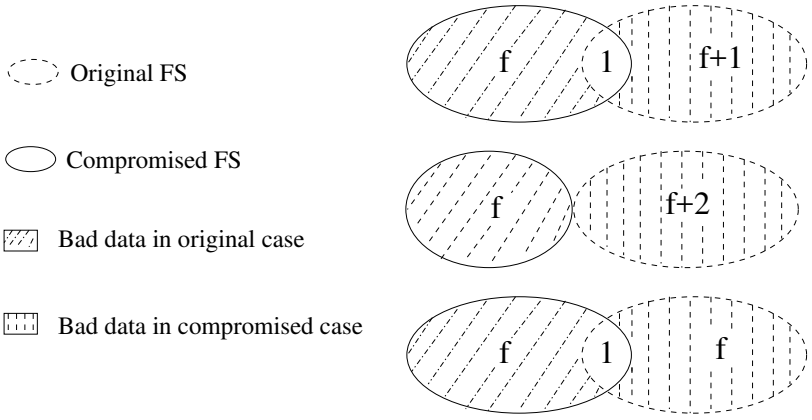

Figure 3.3: Illustration of two feasible sets. FS is the abbreviation of feasible set.

(2) These f measurements are not consistent. In this case, for the original feasible set, the center node will conclude that there are f compromised measurements. For any other feasible set, its cardinality is at most f ($f - 1$ of the f compromised measurements plus one of the intact measurements), so the center node will conclude that there are at least $f + 2$ compromised measurements. Again, the center node can identify the set of compromised measurements.

To prove that f is tight, we show by an example that the uncertainty can still exist if f out of $2f + 1$ sensors, instead of $2f + 2$ sensors, are compromised. In this example, the attacker compromises f measurements, and the original feasible set and the com-

promised feasible set have one measurement in common, as shown in the bottom of Figure 3.3. As a result, the compromised feasible set has $f + 1$ measurements, which are the f compromised measurements plus the measurement in common. The original feasible set also has $f + 1$ measurements. Therefore, for both feasible cases, the center node will conclude that there are f compromised measurements. Thus the center node still does not know which feasible case is the real case. Therefore, f is tight. □

In *Theorem 1*, we assume that the sensing reports from uncompromised sensors have no measurement error. When measurement errors do exist, the value of N_c will decrease. In the case without measurement errors, we look for a set of feasible measurements by checking whether the norm of residuals is zero. In the case with measurement errors, however, we check whether the norm of residuals is less than a positive threshold. That is, the check condition is loosened. Therefore, N_c will decrease.

Though it is hard to know in advance how many sensors are compromised, we can estimate how many sensors an attacker can compromise, considering the effort and resource the attacker can take. We do not consider the case that the attacker compromise all or most of the sensors, since in this case there is no way to pinpoint the compromised sensors. As in *IRIS* [38], the authors set a threshold, N_{min}, which is the required minimum number of remaining sensors.

Following the theorem above, we argue that *IRIS* may eliminate data from good sensors instead of bad sensors and get an incorrect feasible solution under cooperative attacks, especially when the algorithm hit the threshold N_{min} first. In the following, we will propose a modified version of *Combinatorial Optimization Identification* (COI) [41] to complement *IRIS*.

3.4.3 Modified *COI*

The original *COI* is an approach for identifying multiple instances of bad data in power system state estimation. The essential idea is to construct a partial decision tree using

the branch-and-bound method to obtain a feasible solution with the minimum number of bad data. We borrow this idea and make two modifications to fit our problem.

As mentioned above, there may be more than one feasible solution. Therefore, our first modification is to find all feasible solutions instead of only the one with the minimum number of bad data. The second modification is setting a time threshold to meet the time requirement in cooperative spectrum sensing. For instance, in IEEE WRANs, the center node must make a detection decision once every 2 seconds. We will run the branch-and-bound method with increasing bound until hitting the time threshold.

We first illustrate the branch-and-bound strategy, and then present the complete algorithm. The branch-and-bound method will construct a partial decision tree. Since the data with the largest normalized residual is usually more likely to be compromised, after each state estimation run, we pick the sensor with the largest normalized residual as the target node. Each target node has two branches, the right branch representing the case that the sensor is compromised and its left branch representing the other case that the sensor is good, as shown in Figure 3.4.

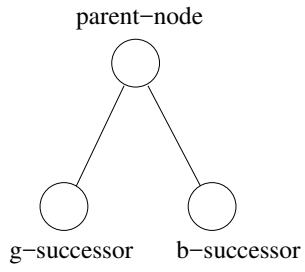

Figure 3.4: Successors of a node.

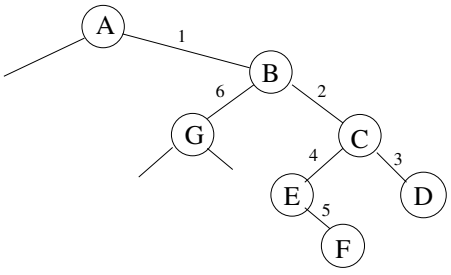

Figure 3.5: Branch strategy in the decision tree.

The state estimation is conducted at each target node assuming that all undeclared sensors are good. The next sensor to target is the one whose sensing report has the largest normalized residual among all the undeclared sensors. In Figure 3.4, the b-successor will be the sensor with the largest residual among all undeclared sensors assuming the parent node is compromised, and the g-successor will be the sensor with the largest residual among all undeclared sensors assuming the parent node is good. The

key strategy of the tree construction is to first move down towards the right until a feasible solution is reached, and then backtrack to find better solutions. In backtracking, the algorithm stays as far as possible to the right. Let us use an example to illustrate this strategy. As shown in Figure 3.5, the tree is constructed in the following order:

(1) at the beginning, sensor A has the largest normalized residual, and it becomes the root of the tree; with node A declared bad, run state estimation and node B emerges to have the largest normalized residual; then node B becomes the b-successor of node A;

(2) with node B declared bad, run state estimation and node C has the largest normalized residual; then node C becomes the b-successor of node B;

(3) similar to step (2), construct node D; suppose a feasible solution is found;

(4) backtrack to node C, assume node C is good and run state estimation; node E becomes the g-successor of node C;

(5) construct node F; suppose another feasible solution is found;

(6) backtrack to node B and construct nod G.

The above construction only shows branching. The bounding is taken care of by a heuristic parameter h. In the tree, a g-successor means that one refuses to make a decision that the data with largest normalized residual is compromised, and instead asks for more information about the remaining data. During tree construction, the algorithm keeps a record on how many times a candidate solution takes a g-successor. If a node already has h g-branches between itself and the root, no more g-branches are considered for itself and its successors.

Before we detail our modified *COI*, we define some parameters and functions. We use b_i to denote three states of sensor i, declared bad, declared good, undeclared, with 0, 1, -1, respectively. We use a vector to represent a candidate problem $v = [b_1, b_2, ..., b_N]$; for instance, in the very beginning, the candidate problem is $v = [-1, -1, ..., -1]$. We set the parameter $h = 3$. As pointed out in [41], $h = 2$ is usually enough. We define two functions, $Nzeros(\mathbf{v})$ and $Nones(\mathbf{v})$, which operate on a vector **v** and return the number of zero elements in **v** and the number of one elements in **v**, respectively. t_0 is the acceptable

time delay for a detection decision, η is the attack detection threshold. The candidate problem with different numbers of g-successors are stored in different stacks. Unless t_0 is used up, the algorithm will first examine candidate problems with no more than one g-successor in stack S_1, then those with 2 g-successors in stack S_2, and finally those with 3 g-successors in stack S_3. Our algorithm is trying to find better feasible solutions than that from *IRIS* without violating the detection delay requirement. The algorithm is detailed in **Algorithm 3.1**.

Algorithm 3.1: Modified *COI*.

Input: $h = 3$, t_0;
Output: Set **F** that contains all sets of feasible measurements.

```
1  t = 0; b_i = −1, 1 ≤ i ≤ N; v = [b_1, b_2, ..., b_N];
2  S_1 = push(v);
3  while (S_1 ≠ ∅ .or. S_2 ≠ ∅ .or. S_3 ≠ ∅) .and. t < t_0 do
4      if S_1 not empty then
5          ccp = S_1.pop;  // ccp for current candidate problem;
6      else if S_2 not empty then
7          ccp = S_2.pop;
8      else
9          ccp = S_3.pop;
10     run estimation with data having b_i ≠ 0 in ccp, i ∈ [1, N];
11     calculate ||r|| and find index i* such that P_i* has the largest normalized residual;
12     Cost = Nzeros(ccp);   // Cost counts how many reports declared bad in ccp;
13     if ||r|| < η   // find a feasible solution then
14         CB = Cost;   // CB stands for current best solution;
15         F = F ∪ ccp;
16     else
17         if Nones(ccp) < h then   // construct g-successor
18             replace i*th element of ccp by 1;
19             j = Nones(ccp);
20             S_j.push(ccp);
21         if Cost < CB − 1 then   // construct b-successor
22             replace i*th element of ccp by 0;
23             j = Nones(ccp);
24             S_j.push(ccp);
25     t = t + processing time;
```

After running **Algorithm 3.1**, we get set F, which contains sets of feasible measure-

ments. Since all of them are feasible, we cannot favor some over others without further information. In practice, we can estimate the attacker's capability, i.e., the maximum number of sensors he can compromise. Suppose the attacker can at most compromise r sensors, then we can filter out some of them by **Algorithm 3.2**.

Algorithm 3.2: Filtering out some feasible solutions.

```
1 for any vector v ∈ F do
2     if Nzeros(v) > r then
3         F = F\v
```

3.4.4 Discussion

Our modified *COI*, **Algorithm 3.1**, is complementary to *IRIS*, since its first feasible solution completely relies on *the largest normalized residual method*. The solution from *IRIS* may or may not be in **F**. If we set $r = N_{min}$, where N_{min} is a threshold in *IRIS*, and *IRIS* hits the threshold η first, then the solution from *IRIS* is definitely in **F**. Otherwise, it may not be in **F**. In this sense, our modified *COI* complements *IRIS* in two ways. First, if the solution from *IRIS* is not in **F**, *IRIS* pinpoints the compromised measurements incorrectly, while our algorithm most likely finds the truth. Second, if the solution from *IRIS* is the one and only one in **F**, our algorithm is as good as *IRIS*, while our algorithm reveals the uncertainty if there is more than one solution in **F**. Note that if there is more than one solution in **F**, the center node has to take all of them into consideration, or has to resort to some verification approaches to narrow **F** down to one feasible solution.

The problem in this work is purely combinatorial. Our modified *COI* does not scan the decision tree thoroughly, while it only scans the partial tree that most likely contains most of the feasible solutions. To figure out all feasible solutions, one has to use the brute-force search method, which, however, takes too much computational time. The brute-force search method are summarized in **Algorithm 3.3**. In **Algorithm 3.3**, we assume there are at most r bad measurements and check every possible combination (line 1). The r

assumed bad measurements, however, are not necessarily all bad; we re-examine them one by one to see if they are really bad (line 5-9), where ϵ is a pre-defined parameter.

Algorithm 3.3: The brute-force search algorithm.

Input: r, the attacker's capability;
P, the set of all sensing measurements;
Output: Set **F** that contains all sets of feasible measurements.

```
1 for each of (N choose r) bad data combinations, P_B do
2     if P\P_B are consistent then
3         G = P\P_B;  //G is the set of good measurements;
4         Calculate the state vector, x, based on G;
5         for each measurement P_i ∈ P_B, 1 ≤ i ≤ r do
6             if ||P_i − H_{P_i} x|| ≤ ε then
7                 G = G + P_i;
8     F = F ∪ G;
```

3.5 Evaluation

In this section, we will evaluate our algorithm by comparing it with *IRIS*. First, we will present a concrete example to show the process of COI, and compare with *IRIS* using different N_{min}. Next, we will set up a simulation for cooperative attacks, and statistically compare the performance between our algorithm and *IRIS*.

3.5.1 An Example

In section 3.4.1, we showed a counter example in which more than half (four out of seven) sensing measurements are compromised by the attacker and there is no measurement error. Here we detail an example with measurement errors and fewer compromised sensors. Our purpose here is to show the process of *COI* and point out why *IRIS* may not work correctly.

System setting: The transmission power P_0 at the reference distance d_0 is 5dB. The radius of the secondary network is about 1km, and the distance between the secondary

network and the primary transmitter is 5km. There are eight sensors, whose distances away from the primary transmitter are listed in Table 3.1. When there is no measurement error and the primary user is active, sensors will get perfect reports, which are marked as $perfect$ in Table 3.1. The column $with\ noise$ shows the reports with some random noise. Now suppose an attacker has compromised three sensors, sensors 5, 6, and 7, and changed their values to those listed in column *compromised* in Table 3.1.

Table 3.1: The measurements from sensors before and after the attack.

Sensor i	Distance d_i	$perfect$	$with\ noise$	$compromised$
1	4.46	-113.01	-112.81	no change
2	4.62	-113.62	-113.27	no change
3	4.90	-114.64	-114.42	no change
4	5.08	-115.27	-114.94	no change
5	5.20	-115.68	-115.36	-120.18
6	5.44	-116.46	-116.23	-120.96
7	5.72	-117.33	-117.23	-121.83
8	5.88	-117.81	-117.55	no change

By applying our algorithm, it finds four feasible solutions in order, as shown in Table 3.2. By applying *IRIS* with different N_{min}, where N_{min} is the number of required remaining reports after elimination, we get the results shown in Table 3.3.

Table 3.2: Feasible solutions from modified *COI*. CS stands for compromised sensors.

$order$	# of CS	CS	P_0	α
1	6	{2,3,4,5,6,8}	133.4	8.3
2	5	{1,2,3,4,8 }	0.4	4.0
3	4	{5,6,7,8}	1.8	3.9
4	3	{5,6,7}	4.9	4.0

Using the data before the attack, we get the original state vector $\mathbf{P} = [P_0, \alpha] = [5.0, 4.0]$. As we can see from Table 3.3, no matter what N_{min} is, *IRIS* eliminates both good measurements and compromised measurements, and the calculated P_0 is far from the original one. However, our algorithm can find the exact correct solution if we assume the attacker

Table 3.3: The transmission power with different N_{min}.

N_{min}	Compromised sensors	P_0	α
7	{8}	168.5	9.5
6	{8,4}	171.1	9.6
5	{8,4,3}	162.2	9.3
4	{8,4,3,5}	155.9	9.1
3	{8,4,3,5,6}	143.0	8.7
2	{8,4,3,5,6,2}	133.4	8.3

can at most compromise three sensors, shown as the 4th feasible solution in Table 3.2. If we assume the attacker can at most compromise four sensors, there will be two feasible solutions, the 3rd and 4th in Table 3.2. In this case, the center node has to rely on some verification approaches to filter out one of them. Nevertheless, our algorithm does not miss the correct solution.

3.5.2 Statistical Comparison

In this subsection, we statistically compare the performance of our algorithm with that of *IRIS*. We first consider a network with with the size of $N = 40$ sensors. The radius of the secondary network is about 2km, and the nodes are randomly distributed inside the circle. The distance between the secondary network and the primary transmitter is 8km. Before any attack, we introduce some random noise to all sensing reports; the noise for each sensing report is less than 1% of the original sensing report. Then we randomly compromise n_{com} of N sensors and inject cooperative bad sensing reports. n_{com} varies from 1 to 20. For each value of n_{com}, we run the simulation $n = 1000$ times and count the following numbers: (1) n_1, the number of times that *IRIS* gets the nearly correct P_0; (2) n_2, the number of times that *IRIS* identifies the right set of compromised sensors and thus gets the correct P_0; (3) n_3, the number of times that our algorithm gets the nearly correct P_0; (4) n_4, the number of times that that our algorithm identifies the right set of compromised sensors and thus gets the correct P_0. For our algorithm, we evaluate three

cases with $h = 1$, $h = 2$ and $h = 3$. We set $N_{min} = 20$ in *IRIS* and $r = 20$ in our algorithm.

After obtaining all the counts, we calculate the accuracy ratio using two different criteria. The first criterion is to check the performance of P_0 detection. If the resulting power lever is within the range of $[0.9, 1.1]$ of the real P_0, we count it as a correct detection. The accuracy ratios for IRIS and our algorithm are n_1/n and n_3/n, respectively. The second criterion is to check whether all compromised sensors are identified, and the accuracy ratios for IRIS and our algorithm are n_2/n and n_4/n, respectively. Also, we examine the time delay of our algorithm. All the results are summarized in Figure 3.6, Figure 3.7 and Figure 3.8.

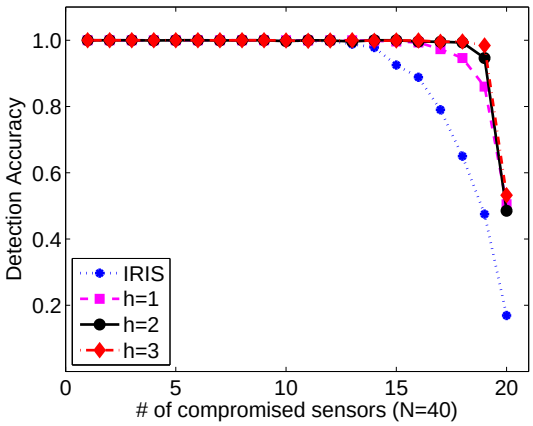

Figure 3.6: P_0 detection accuracy comparison.

Figure 3.6 shows P_0 detection accuracy, and we have the following observations. (1) When $1/3$ of all sensors or less are compromised, *IRIS* and our algorithm work identically and both yield nearly 100% accuracy. (2) When more sensors are compromised, our algorithm demonstrates improved performance over *IRIS* by a noticeable margin. For instance, when $n_{com} = 17$ our algorithm improves *IRIS* by more than 20% for all three h values. (3) When half of all sensors ($N_{com} = 20$) are compromised, our algorithm also performs poorly. The accuracy ratio is only about 50%, regardless of the value of h. This is actually what *Theorem 1* indicates; because there are two feasible solutions with the same number of compromised sensors and our algorithm picks one of them, the ratio is

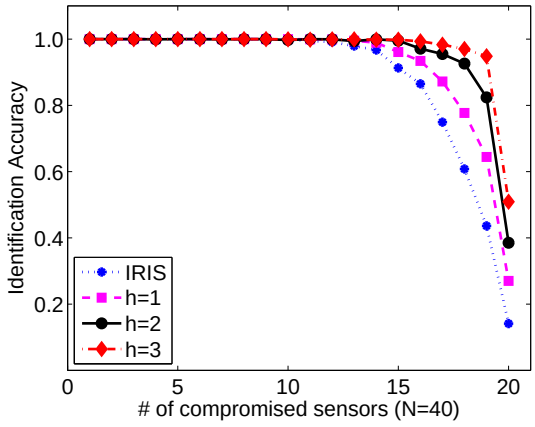

Figure 3.7: Identification accuracy comparison.

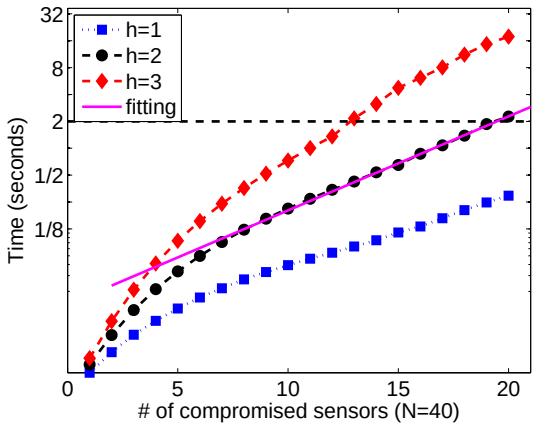

Figure 3.8: Processing time.

on average 50%. (4) The case with $h = 2$ is almost identical to that with $h = 3$, which coincides with the statement in [41] that $h = 2$ is empirically enough.

Figure 3.7 illustrates the identification accuracy. All ratios for $n_{com} > 13$ are less than those in Figure 3.6, indicating that for both algorithms they may not identify the correct set of compromised sensors even though they obtain the correct P_0. We also can see that the margin between the case with $h = 1$ and the case with $h = 2$ is larger than that in Figure 3.6. The margin between the case with $h = 2$ and the case with $h = 3$, however, is still very small, even though it is slightly larger than that in Figure 3.6.

Figure 3.8 shows the processing time for different values of h. Since the processing

time for *IRIS* is less than 60 milliseconds in our simulation, we do not plot it in the figure. Our simulations are run on an Intel Xeon CPU at 3.07GHz with MATLAB. As we can see, for the case with $h = 1$ and the case with $h = 2$, the processing time is within 2 seconds. For the case with $h = 3$, when $n_{com} \leq N/3$, the time is still within 2 seconds; as n_{com} increases, the time goes up to 18 seconds. We can also see, when $n_{com} \in [N/5, N/2]$, the time increases exponentially, indicated by the linear fitting in the log-scaled figure. In our algorithm, we set a time threshold t_0 and $h = 3$. The algorithm may not finish all three stacks. For instance, if we set $t_0 = 2$ seconds, it may only finish the first two stacks and terminate during processing the third stack. No matter whether or not it finishes all the stacks, our algorithm will complement *IRIS*, since it starts with *IRIS* solution and then tries to find other possibilities.

We are also interested in the detection accuracy and identification accuracy with different values of N. Figure 3.9 and Figure 3.10 show P_0 detection accuracy with $N = 20$ and $N = 60$, respectively. And Figure 3.11 and Figure 3.12 show the identification accuracy with $N = 20$ and $N = 60$, respectively. Combining with Figure 3.6 and Figure 3.7, we can see that *IRIS* always drops sharply when n_{com} is within the range $[N/2 - 5, N/2]$, while our algorithm only drops sharply when n_{com} is 1 or 2 away from $N/2$, especially with $h = 3$. That is, for each algorithm with different values of N, the drop-off of the accuracy starts at the same value away from $N/2$, rather than the percentage of N.

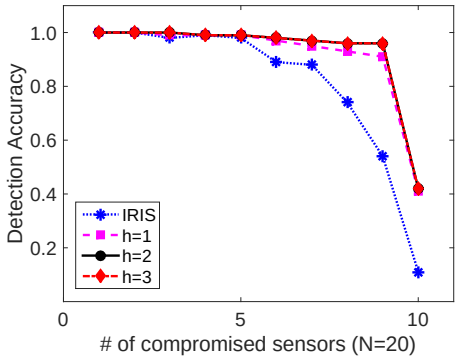

Figure 3.9: P_0 detection accuracy comparison (N=20).

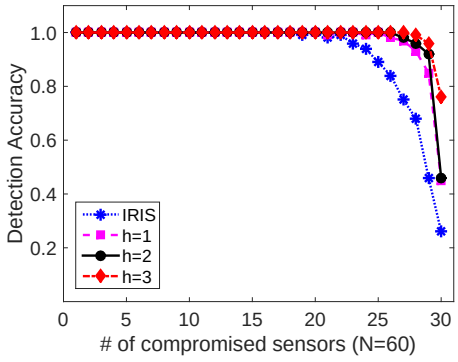

Figure 3.10: P_0 detection accuracy comparison (N=60).

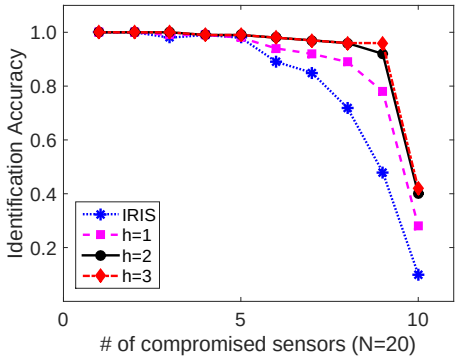

Figure 3.11: Identification accuracy comparison (N=20).

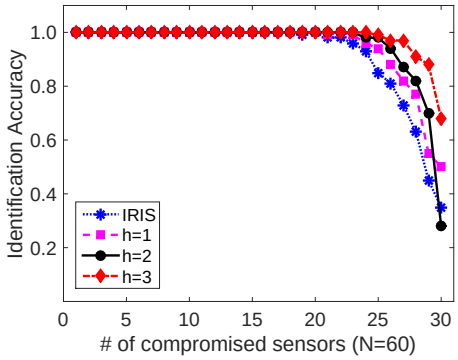

Figure 3.12: Identification accuracy comparison (N=60).

3.6 Conclusion

In this work, we study the security problem providing a trustworthy and reliable communication network in cyber-physical systems. We notice that an existing algorithm for cooperative spectrum sensing in cognitive radio networks, called *IRIS*, is vulnerable to cooperative attacks. To complement *IRIS*, we propose a modified *COI* algorithm to enhance spectrum sensing performance. We also present a theorem to show uncertainties under a cooperative attack. Our proposed algorithm can be flexibly adjusted to meet time delay requirements. We intensively evaluate our algorithm with simulations, and the results show that our algorithm improves the detection accuracy rate noticeably compared to *IRIS*.

Chapter 4

Preserving Secondary Users' Privacy in Cognitive Radio Networks

In this chapter, we continue to investigate cognitive radio networks while with a focus on privacy, which is another important aspect of research in the communication networks of cyber-physical systems (CPS). Specifically, we are interested in secondary users' privacy in cognitive radio transactions, which has been ignored by researchers for a long time.

4.1 Introduction

Cognitive radio has emerged as a fundamental approach in improving the utilization of spectrum resource through dynamic spectrum sharing. As a key technology in wireless communication, cognitive radio enables secondary users (SUs) to opportunistically share the licensed spectrum with primary users (PUs). To make the spectrum sharing attractive, it is essential to design incentive mechanisms for both PU and SU. It is reasonable that the owner of the licensed spectrum, i.e., PU, charges SU whenever the latter is utilizing licensed spectrum. Recently, several approaches, such as game theory and auction

theory, have been applied to this topic, maximizing either PU's gain or the social welfare. However, SU's privacy in cognitive radio transactions has been neglected or left untouched.

In a typical cognitive radio transaction, SU's payment to PU depends on SU's detailed usage information, such as when and how long the licensed spectrum has been utilized. PU needs this usage information to calculate or verify the payment, but detailed usage information is sensitive to SU and the disclosure of this information may compromise SU's privacy. Therefore, protecting PU's interest and preserving SU's privacy simultaneously becomes very challenging. To the best of our knowledge, none of the existing work has addressed this problem in cognitive radio transactions.

In this chapter, we propose a novel privacy-preserving mechanism for cognitive radio transactions [48]. This mechanism not only preserves SU's privacy but also protects PU's interest. In this mechanism, PU only knows a little portion of SU's sensitive information during a billing period with the result that SU's privacy is preserved. At the same time, PU knows the total payment for a billing period and is guaranteed that the payment is correctly calculated and PU's interest is protected. This mechanism employs the commitment scheme and the zero-knowledge proof. At the end of a billing period, SU commits all detailed usage information and the payment for each utilization instance, and provides a zero-knowledge proof for each utilization instance that the payment is correctly calculated. These commitments and proofs, along with the total fee, are sent to PU. Due to the *hiding* property of the commitment scheme, the commitments do not reveal any information about the detailed usage information. Due to the *binding* property of the commitment scheme, SU cannot deny the values that are used to create the commitments. Furthermore, by verifying the zero-knowledge proofs provided by SU, PU is able to verify that the payment for each utilization instance is correct. To prevent SU from committing fraud, such as choosing not to submit all utilization instances or submitting false utilization instances, we introduce a random-checking monitor that can provide some ground-truth information on the spectrum utilization status. PU can opportunistically query the monitor

to ask for a few pieces of ground-truth information, and use this information to challenge SU. Once SU is challenged, it must provide proofs to match the random-checking information.

Our main contributions in this chapter are three-fold. First, we have pointed out an important security problem in cognitive radio literature, preserving SU's privacy, which has been neglected for a long time. Second, we are the first to address this problem and have proposed a privacy-preserving mechanism to protect PU's interest and preserve SU's privacy at the same time. Third, we have implemented our mechanism and evaluated its performance.

The rest of chapter is organized as follows. Section 4.2 is the related work. Section 4.3 describes our privacy-preserving model, including threat model, trust model, and system architecture. In Section 4.4, we review some preliminaries and construct our privacy-preserving protocol in detail. In Section 4.5, we present the security analysis. In Section 4.6, we propose the monitoring scheme and examine its effectiveness. We implement and evaluate our proposed protocol in Section 4.7, and conclude this work in Section 4.8.

4.2 Related Work

Cognitive radio is the key approach that allows SUs to opportunistically access the licensed spectrum owned by PUs through spectrum sensing [19, 29, 30, 40, 47, 69]. Here we will review some work on cognitive radio transactions, and discuss some privacy and security issues in cognitive radio market.

4.2.1 Cognitive Radio Transaction

In a cognitive radio network with incentive mechanism, PU wants to maximize its revenue, and SU wants to maximize its utility, which is usually the gain from the wireless service minus the payment. Therefore, cognitive radio pricing is an indispensable component for incentive cognitive radio networks, and has been addressed by many researchers, such

as through auction models [13, 42, 62]. In this work, our focus is on SU's privacy, which is neglected in auction models, rather than pricing itself. We assume that PU announces a pricing policy to all SUs, similar to the uniform pricing model in [13]. Then every SU who utilizes the licensed spectrum pays PU according to the pricing policy. Our interest is to hide SU' detailed usage information from PU and thus preserve SU' privacy.

4.2.2 Privacy and Security

Usage privacy usually relates to when, where and how a consumer, such as a person or an object, is consuming a non-free resource. The owner of the resource may need the detailed usage information to calculate a payment to charge the customer, while the detailed information may breach the customer's privacy. This dilemma exists in many areas, such as smarter metering and vehicular electronic tolling. Privacy-preserving solutions have been proposed in smart metering [51], vehicular tolling [4, 44], and other topics [56, 63, 74], and can potentially be applied to some attacks in smart grid and social networks [45, 46, 64].

Cognitive radio marketing faces the same dilemma. Usually SU is charged based on its spectrum usage profile, such as when and how long it utilizes a certain channel, while the usage profile inevitably breaches its privacy. Though much work has been done on the incentive mechanism for cognitive radio, to the best of our knowledge, SU's privacy, which is related to fine-grained usage profile has never been addressed.

4.3 Privacy-Preserving Model

We consider a typical cognitive radio network, which consists of a primary user (PU) and multiple secondary users (SUs). The PU has some non-utilized spectrum to sell to SUs. In a cognitive radio transaction, PU announces the pricing policy for the non-utilized spectrum, typically a price function depending on frequency, bandwidth, time, etc. SU

utilizes some spectrum and pays PU according to the pricing policy at the end of a billing period, such as a day or a month.

4.3.1 Adversary and Threat Model

In this work, we assume secure communication. Messages among all entities are signed by the senders and can be verified by the recipients. Our focus in this work is on PU's interest and SU's privacy. We consider the following threats from the prospective of both PU and SU.

PU is concerned about the following threats:

T1. SU does not report all its utilization data;

T2. SU reports false utilization data, such as the time when it starts to use the licensed spectrum and the time duration of one utilization instance;

T3. SU uses incorrect price to calculate the subfee for a utilization instance and thus reports false subfee;

T4. SU reports false total fee for a billing period.

SU is concerned the following threat:

T5. PU learns a large amount of the detailed usage information to breach SU's privacy.

4.3.2 Design Goals

There are two goals for our privacy-preserving mechanism. The first one is to preserve SU's privacy. The second one is to protect PU's interest.

SU should never send its detailed utilization information to PU in plain text. The detailed utilization information, such as when (time) and how (frequency, bandwidth) SU utilizes the licensed spectrum, compromises SU's privacy. Instead of the utilization information in plain text, SU commits to the utilization information and sends the commitments

to PU. As we explain later, the commitments do not disclose any information about the committed values; thus, SU's privacy is preserved.

At the same time, PU's interest must be protected. PU must be ensured that SU does not commit fraud. Even though PU only obtains the commitments of SU's utilization information, it must have a way to verify the payment through the commitments. We resort to a homomorphic commitment scheme, which can hide SU's utilization information and help PU verify the payment.

4.3.3 Trust Model

To achieve the design goals, trustworthy information about SU's utilization is required. Otherwise, there is no way for PU to determine whether SU commits fraud. Hence, we introduce a trusted random-checking monitor, which wakes up to check the utilization status of the spectrum in a way that SU cannot predict. The monitor is trusted, but it cannot scan all the spectrum channels all the time due to limited monitoring resources or cost concern. For instance, the monitor may not be able to cover all spectrum channels at the same time due to its functionality constraints, or the cost concern may refrain the monitor from monitoring all channels all the time even it is capable to do so.

We assume that SU does not know the pattern how the monitor scans all channels. Since the monitor records some information on the status of spectrum utilization, SU who does not report or reports false information has the risk to be caught. Once a fraud is caught, we can impose a high penalty to deter SU from lying. Therefore, the monitor will enforce SU to report true spectrum utilization, i.e., when and how long SU user has used a certain channel.

4.3.4 System Architecture

The system model comprises three components, *Primary User* (PU), *Secondary User* (SU), and *Monitor* (M), as shown in Figure 4.1. PU announces pricing policy to SU, which

is in the form of segments, with each segment representing the unit price for a certain channel in a period of time. SU calculates a subfee for each utilization instance based on the pricing policy if he utilizes a certain channel for a period of time. At the end of a billing period, SU sums up all the subfees to obtain a total fee and sends it to PU, together with the commitments of information in each utilization instance. PU only obtains the information of the total payment from SU, but it does not know when and how long SU has utilized a channel, nor does it know which channel SU has utilized. In this way, SU's privacy is preserved. To check whether SU commits fraud, PU can ask for some observations from the monitor M, and require SU to provide the corresponding commitments. If the observations and commitments are matched, then SU is considered honest; otherwise, a high penalty is imposed. We limit the number of random-checking observations that a primary user can ask for during a billing period such that only a limited amount of SU's private information is leaked. To guarantee the integrity of messages, all three entities implement a signature scheme such that any message cannot be forged.

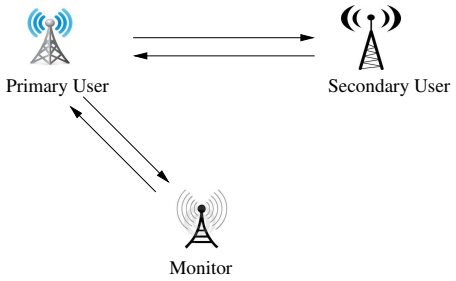

Figure 4.1: System architecture for privacy-preserving pricing in cognitive radio network.

Remark: In the above architecture, we introduce a new entity, the monitor M, into the system to randomly check whether SU is utilizing a channel. Alternatively, we may let SU carry out the monitoring functionality [68]. When an SU transmits a message over a licensed channel, it concatenates the message and the current time, and hash the concatenated message. Then the SU signs the hashed value. Other SUs nearby can capture the signed message in the corresponding channel. When PU asks an SU u_i for utilization instances of another SU u_j, user u_i can provide user u_j's signature together

with the corresponding time to PU. With the signature, user u_j cannot deny its utilization of the licensed spectrum. To motivate SU to participate in monitoring, PU can offer some rewards to SUs who provide monitoring information.

4.4 Protocol Construction

In this section, we will first review some preliminaries that are extensively used in our privacy-preserving protocol. Then, we will show our protocol in detail.

4.4.1 Preliminaries

4.4.1.1 Signature Scheme

A signature scheme consists of three fundamental components, key generation (**Keygen**), signature generation (**Sign**), and verification (**Verify**). **Keygen** generates a key pair (sk, pk), where sk and pk are the secret key and the public key, respectively. **Sign**(sk, m) outputs a signature m_s on message m. **Verify**(pk, m_s, m) outputs *accept* if m_s is a valid signature of message m; otherwise, it outputs *reject*. A signature must be unforgeable such that an adversary should not be able to obtain the signature m_s on message m unless he knows the secret key sk or has previously obtained a signature on m.

4.4.1.2 Commitment Scheme

A commitment scheme consists of three fundamental phases, commitment setup phase (**ComSetup**), commit phase (**Commit**), and open phase (**Open**). **ComSetup** generates the parameters of the commitment scheme, par. **Commit**(par, x) outputs a commitment c_x to x and auxiliary information $open_x$. **Open**(par, c_x, x, $open_x$) outputs *accept* if c_x is the commitment to x and $open_x$ is the corresponding auxiliary information. A commitment scheme has two important properties, $hiding$ and $binding$. The hiding property guaran-

tees that c_x does not reveal any information about x, and the binding property guarantees that c_x cannot be opened to values other than x.

A commitment scheme is said additively homomorphic if, given two commitments c_{x_1} and c_{x_2} with openings $open_{x_1}$ and $open_{x_2}$ respectively, **Open**(par, $c_{x_1} \cdot c_{x_2}$, $x_1 + x_2$, $open_{x_1} + open_{x_2}$) outputs *accept*.

4.4.1.3 Zero-knowledge Proof

A zero-knowledge proof of knowledge is a method for a prover to prove to a verifier that a statement is true, without revealing anything other than the veracity of the statement. A zero-knowledge proof of knowledge has three important property, *completeness*, *soundness*, and *zero-knowledge*. Completeness guarantees that an honest verifier will be convinced by an honest prover if the statement is true. Soundness guarantees that no malicious prover can convince a verifier that a statement is true if it is actually false, except with some negligible probability. *Zero-knowledge* guarantees that the verifier learns nothing other than the statement is true.

For instance, a prover has a value x and its commitment $c_x = g^x h^{open_x}$, where g and h are parameters of a commitment scheme. He sends only c_x, g, and h to a verifier. By means of zero-knowledge proof, the prover can prove to the verifier that he knows the committed value x and its corresponding open message $open_x$, without revealing x and $open_x$. During a typical proof, the verifier inquires the prover with some random challenge, and the prover makes a response to the challenge. Then the verifier verifies some equalities. This interactive proof can be turned into non-interactive zero-knowledge proof via Fiat-Shamir heuristic [11].

4.4.2 Construction Sketch

As in Figure 4.1, there are in total three entities in the system: primary user (PU), secondary users (SUs), and a random-checking monitor (M). Our privacy-preserving protocol

has three stages, i.e., *initialization*, *transaction*, and *random-checking*.

4.4.2.1 Initialization

At the beginning, each entity generates a key pair for signature scheme, i.e., (sk_{PU}, pk_{PU}) for PU, (sk_{SU}, pk_{SU}) for SU, (sk_M, pk_M) for the monitor M. Each entity distributes its own public key, and stores the public keys from other entities. With these key pairs, each entity can verify the origin and integrity of each message received. SU also generates the parameter par for its commitment scheme, and share par with PU.

When PU has some licensed spectrum f (such as a channel) to sell, he announces tuples in the form of $< T_{b_i}, T_{e_i}, f >$, each of which denotes that a certain SU may use spectrum f in the time interval $[T_{b_i}, T_{e_i}]$. PU also establishes a function $F : (T_{b_i}, T_{e_i}, f) \rightarrow \Phi$ that maps every tuple $< T_{b_i}, T_{e_i}, f >$ to a price $p_i \in \Phi$, where p_i is the unit price ($\$/second$) for spectrum f in the time interval $[T_{b_i}, T_{e_i}]$[7]. PU signs the pricing policy Φ to get Φ_s, where $\Phi_s =$ **Sign**(sk_{PU}, Φ), and sends (Φ_s, Φ) to SU.

When SU receives Φ_s, it runs **Verify**(pk_{PU}, Φ_s, Φ). If the verification outputs *accept*, SU stores the pricing policy Φ; otherwise, SU notifies PU that the message is tampered.

4.4.2.2 Transaction

When SU utilizes a segment of the spectrum with frequency f, it records the starting time t_i and the time duration $\triangle t_i$, denoted as a tuple $< t_i, \triangle t_i, f >$, which is called a utilization instance. First, it determines which pricing policy to use. Suppose $T_{b_j} \leq t_i \leq T_{e_j}$, then the unit price is p_j. Then, it calculates the payment $fee_i = p_j \triangle t_i$ [8]. Next, it computes a payment tuple $u_i = (c_{t_i}, c_{\triangle t_i}, c_f, c_{fee_i}, c_{p_j}, \pi_i)$, where c_x is a homomorphic commitment of x with auxiliary information $open_x$ and parameter par, $x \in \{t_i, \triangle t_i, f, fee_i, p_j\}$, π_i is the non-interactive zero-knowledge proof that (1) SU knows all openings of all commit-

[7] Here we assume a linear pricing policy. Our scheme can be adopted to other complicated pricing policy, such as cumulative policy in [51].

[8] When $\triangle t_i$ spans two pricing policies, we break this segment into two segments at the boundary of two pricing polices.

ments; (2) fee_i committed in c_{fee_i} is the product of p_j and $\triangle t_i$ committed in c_{p_j} and $c_{\triangle t_i}$, respectively; (3) SU possesses a C-L signature [5] on p_j computed by PU that the price p_j belongs to Φ and it is the right one to calculate the subfee fee_i.

Suppose SU has utilized n tuples of spectrum resources in a billing period. At the end of the billing period, SU adds up all the subfees of n tuples to obtain a total fee $fee_{total} = \sum_{i=1}^{n} fee_i$, and adds up all the openings to obtain an opening for the total fee, $open_{fee_{total}} = \sum_{i=1}^{n} open_{fee_i}$. Then, SU composes a payment message m that consists of $(fee_{total}, open_{fee_{total}}, id)$ and all the payment tuples $u_i = (c_f, c_{t_i}, c_{\triangle t_i}, c_{fee_i}, c_{p_j}, \pi_i)$, i.e., $m = (fee_{total}, open_{fee_{total}}, id, u_{i=1}^{n})$. SU signs message m to obtain its signature m_s, where $m_s =$ **Sign**(sk_{SU}, m) and sends (m, m_s) to PU.

When receiving (m, m_s), PU first runs **Verify**(pk_{SU}, m_s, m) to verify SU's signature on the message m. If the verification outputs *accept*, PU then verifies the proof π_i in each payment tuple u_i, $\forall i \in [1, n]$. If all the verifications output *accept*, PU times up commitments c_{fee_i} of all payment tuples to obtain the commitment $c_{fee_{total}} = \prod_{i=1}^{n} c_{fee_i}$. At the end, PU checks whether $open_{fee_{total}}$ is a valid opening by running **Open**(par, $c_{fee_{total}}$, fee_{total}, $open_{fee_{total}}$). If it outputs *accept*, PU will query the monitor M as described below.

4.4.2.3 Random-checking

The monitor, M, scans to check whether a SU is utilizing the spectrum with frequency f (such as a channel). For each observation, it generates a signed statement ϕ that an SU with its identity id is utilizing the spectrum with frequency f at time t_c, i.e., $\phi = (t_c, f, id)$. To check whether a certain SU submits all payment tuples and the payment for each tuple is correctly calculated, PU sends a request message $r = (request, id)$, together with its signature $r_s =$ **Sign**(sk_{PU}, r), to M to ask for one statement ϕ for SU with identity id. When receiving the request, M verifies PU's signature by running **Verify**(pk_{PU}, r_s, r). If the verification is accepted, M sends one piece of statement ϕ with signature, i.e., $\phi_s =$ **Sign**(sk_M, ϕ), to PU. Then PU relays ϕ_s to SU. SU first verifies whether ϕ_s has a valid

signature from M. If the signature is valid, it searches among its payment tuples and finds the one with the time domain $[t_i, t_i + \triangle t_i]$ such that $t_i < t_c < t_i + \triangle t_i$. Once the tuple is found, SU will send all the openings for all the commitments in that payment tuple in a signed message, $o_s = \textbf{Sign}(sk_{SU}, open_{t_i}, open_{\triangle t_i}, open_f, open_{fee_i})$. Then PU can verify whether the SU has submitted the payment tuple corresponding to ϕ and whether the SU computes the price fee_i correctly. PU can request more than one statements at a time. However, the number of statements must be limited. If SU commits fraud, a penalty will be imposed.

The whole protocol is shown in Figure 4.2.

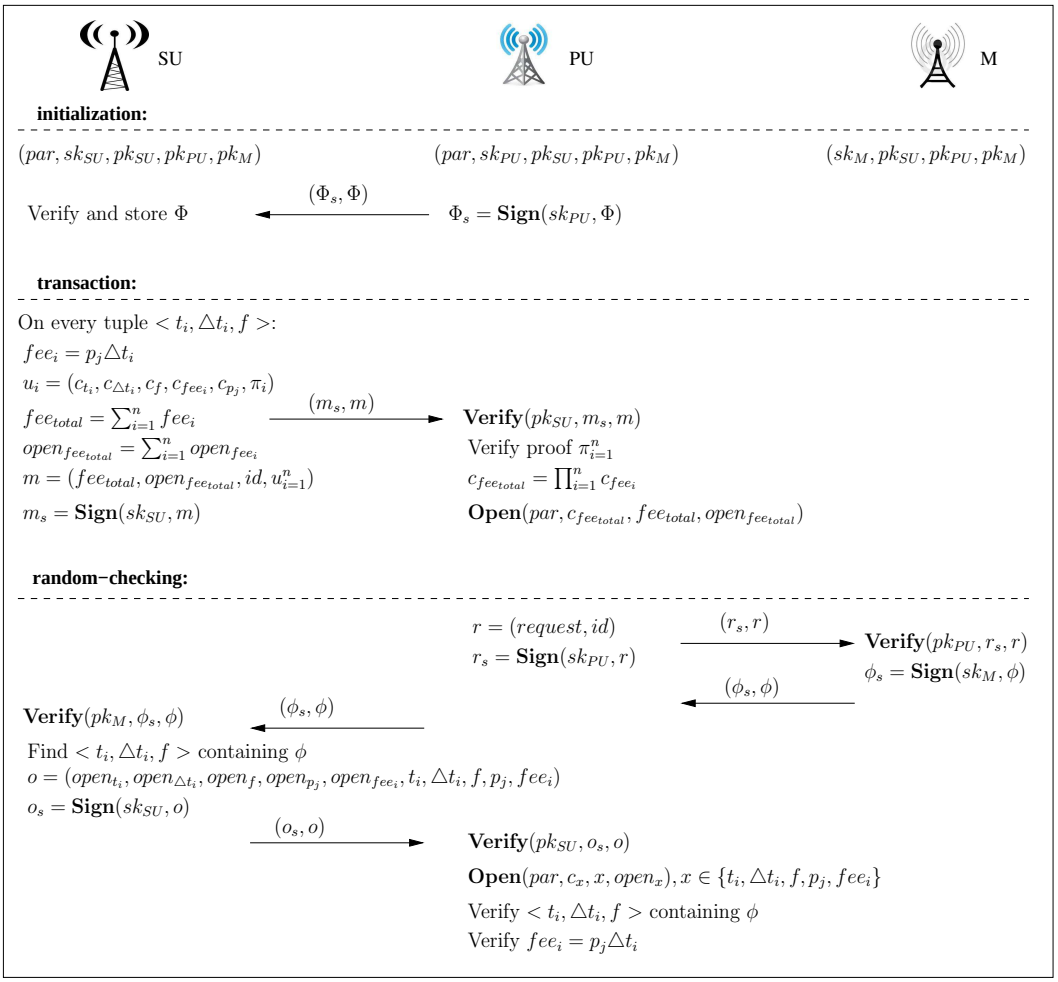

Figure 4.2: Structure of the protocol. SU=Secondary User, PU=Primary User, M=Monitor.

4.5 Security Analysis

In this section, we will analyze whether our protocol in Section 4.4 can thwart all the threats mentioned in Section 4.3.1.

4.5.1 Privacy and Interest

Claim 1: Our protocol can defend against threats **T1**, **T2**, **T3**, and **T4**, thus the primary user's interest is protected.

The random-checking monitor and the imposed penalty defend against both **T1** and **T2**. If a secondary user purposely chooses not to report some of its utilization data, it has the risk to be caught by the monitor, and the imposed penalty once being caught deters a secondary user from committing **T1**. Similarly, a secondary user has the risk to be caught by the monitor if it commits **T2** by reporting false utilization data. Regarding **T3**, the proof that a secondary user possesses a C-L signature on the pricing policy makes sure that the correct unit price is used to calculate the subfee for a utilization instance. Furthermore, a secondary user cannot commit fraud **T4** once all subfees are correct. The homomorphic property of the commitments scheme can guarantee that the total fee is the sum of all the subfees. □

Claim 2: A secondary user's privacy is preserved except that information in the responses of queries is disclosed.

At the end of a billing period, the secondary user only sends the total fee and its id in plain text to the primary users. All other information is sent as commitments and zero-knowledge proofs. As we know, the hiding property of a commitment and the zero-knowledge property of a zero-knowledge proof guarantee that all information in commitments and zero-knowledge proofs is not revealed to the primary user. The only information disclosed is from the responses of queries when the secondary user responds to the queries made by the primary user. □

4.5.2 Discussion

In our protocol, besides the information in the responses of queries, the primary user can also learn the number of utilization instances. For each utilization instance, the secondary user calculates a subfee and sends the commitment of the subfee to the primary user. By counting the number of subfee commitments, the primary user learns the number of utilization instances. Though this information does not reveal any detailed information in each utilization instance, it does disclose how often a secondary user utilizes the licensed spectrum. The primary user can also infer whether a secondary user is more active than another one. To hide the number of utilization instances, the secondary user can submit some dummy utilization instances, such as some tuples $< t_i, \triangle t_i, f >$ with $\triangle t_i = 0$. In this way, the primary user cannot learn the exact number but a upper bound of utilization instances. To even hide the upper bound, i.e., to make the upper bound meaningless, each secondary user can submit a large amount of dummy instances. Too many dummy instances, however, introduce heavy overhead, especially for less active secondary users.

Since the responses of queries disclose secondary users' information, we must limit the number of queries a primary user can make on an individual secondary user. And we assume that the information disclosed in the limited number of queries is tolerable. If even this limited information is not tolerable, we can hide the information disclosed in the responses of queries by changing a little bit the system architecture. In the architecture shown in Figure 4.2, the monitor does not communicate with the secondary user. We can add a communication link from the monitor to the secondary user. When the primary user queries the monitor, the monitor randomly chooses a statement ϕ, for instance, that the secondary user is utilizing a channel at time t_u. Instead of sending t_u to the primary user as in our protocol, the monitor only sends the commitment of t_u to the primary user. At the same time, the monitor sends the opening message of the commitment of t_u to the secondary user. Then the secondary user proves to the primary user by interval check and zero-knowledge proof that (1) it knows the opening of commitment of t_u and (2) t_u is

a point of time in one of its utilization instances submitted. Even there is no link from the monitor to the secondary user, we can utilize public key cryptography to let the primary user relay t_u to the secondary user; the monitor encrypts t_u with the secondary user's public key, and only the secondary user can decrypt it since no other entity has its secret key.

4.6 Random Checking

As described above, random checking plays an important role in our protocol. In this section, we will first propose rules for the monitor to efficiently capture secondary users' utilization instances. Then we will analyze the effectiveness of random checking. Specifically, we show that the monitor can capture enough random checking instances even when the probability to capture each individual instance is low, and the captured random checking instances can considerably catch a secondary user who commits fraud.

4.6.1 Monitoring Scheme

Suppose a primary user has n channels of licensed spectrum to share with l secondary users. And there is one monitor that can only monitor one channel at a time. Our problem is that what strategy the monitor should apply such that the monitor can capture as many utilization instances among all secondary users and all channels.

4.6.1.1 Problem Statement

Suppose that the traffic on all channels is the same. We assume that on each channel both the duration of busy time and the duration of idle time follow exponential distributions with parameter λ_B and λ_I, respectively. The average busy and idle times are denoted by $T_B = 1/\lambda_B$ and $T_I = 1/\lambda_I$, respectively. We also assume that when a monitor jumps from one channel to another, there is a switch cost; that is, it wastes a period of time T_c.

The objective is to capture as many utilization instances as possible in a given period of time T_0 among all secondary users. As the monitor does not know how secondary users choose channels and when they utilize the chosen channels, we assume that secondary users randomly choose channels and utilization instances can be randomly distributed on each channel.

4.6.1.2 Monitoring Rules

It is hard to give an optimal solution to the above problem deterministically. Here we propose some heuristic rules to control the monitor. Remember that a random check instance is a point of time that a certain secondary user is utilizing a channel. The monitor can choose to either stay on the current channel or switch to another channel. Our intuition is to let the monitor make a better choice between stay and switch. Suppose at a time the monitor is not scanning any channel, and now it jumps to a channel to start scanning. Then the monitor have to decide what to do after jumping to a channel. There are in total two scenarios: (i) the monitor jumps to a channel currently busy; (ii) the monitor jumps to a channel currently idle.

• When a monitor jumps to a channel which is currently busy, it will stay until it is able to figure out which secondary user is utilizing that channel. After the monitor has determined the secondary user's identity, i.e., captured a random-check instance, it will determine whether to stay on the current channel or switch to another channel. If it chooses to stay, the probability that it encounters a busy channel during T_c, p_{stay_b}, is:

$$p_{stay_b} = \int_0^{T_c} \int_0^{y} \lambda_B \lambda_I e^{-\lambda_B x} e^{-\lambda_I (y-x)} dx dy = \begin{cases} 1 - \frac{1}{\lambda_B - \lambda_I}(\lambda_B e^{-\lambda_I T_c} - \lambda_I e^{-\lambda_B T_c}) & \text{if } \lambda_B \neq \lambda_I \\ 1 - \lambda_B T_c e^{-\lambda_B T_c} - e^{-\lambda_B T_c} & \text{if } \lambda_B = \lambda_I \end{cases} \tag{4.1}$$

In the case of switch, the monitor wastes T_c of time for monitoring. At the time point of

T_c, the probability that the channel is busy, p_{switch}, is:

$$p_{switch} = \frac{T_B}{T_B + T_I} \tag{4.2}$$

We can obtain the critic switch cost, T_c^*, by setting $p_{stay_b} = p_{switch}$. When $T_c > T_c^*$, we have $p_{stay_b} > p_{switch}$; the monitor should continue to stay on the channel that it is currently monitoring. When $T_c < T_c^*$, we have $p_{stay_b} < p_{switch}$; the monitor should switch to another channel which has been monitored less frequently. When $T_c = T_c^*$, we have $p_{stay_b} = p_{switch}$; the monitor can choose to either stay on the current channel or switch to another channel.

• When the monitor jumps to an idle channel, we can obtain another critic switch cost $T_c^{*'}$ similarly. In the case of stay, during the time period of T_c, the probability that the monitor encounters a busy channel, p_{stay_i}, is:

$$p_{stay_i} = \int_0^{T_c} \lambda_I e^{-\lambda_I x} dx = 1 - e^{-\lambda_I T_c} \tag{4.3}$$

In the case of switch, at the time point of T_c, the probability that the channel is busy, p_{switch}, is same as Eq(4.2). By setting $p_{stay_i} = p_{switch}$, we get $T_c^{*'} = T_I \ln \frac{T_B}{T_B + T_I}$. When $T_c > T_c^{*'}$, we have $p_{stay_i} > p_{switch}$; the monitor should continue to stay on the idle channel that it is monitoring. When $T_c < T_c^{*'}$, we have $p_{stay_i} < p_{switch}$; the monitor should switch to another channel which has been monitored less frequently. When $T_c = T_c^{*'}$, we have $p_{stay_i} = p_{switch}$; the monitor can choose to either stay on the current channel or switch to another channel.

We summarizes the rules as follows:

1. When a monitor jumps to a channel that is currently busy, it stays until it identifies the secondary user that is utilizing the channel. Then the monitor chooses to stay or switch according the values of T_c and T_c^* as described above;

2. When a monitor jumps to a channel that is currently idle, it chooses to stay or switch

according T_c and $T_c^{*'}$ as described above.

4.6.2 Random-check Effectiveness

Since the monitor cannot cover all the channels all the time, it only captures a portion of the total utilization instance. Note that in Section 4.5, we assume that the random-check mechanism refrains a secondary user from committing fraud. Here we analyze the effectiveness of the random-check mechanism. We will show that a secondary user has very high probability to be caught if it commits fraud even the monitor only captures a limited number of random-check instances.

Let p be the probability that the monitor captures a secondary user in one second interval when the latter is utilizing a channel. To simplify the analysis, we assume that the probability is the same for all channels in the licensed spectrum. Let n be the total time (in seconds) that a secondary user has utilized the licensed spectrum[9]. Let X be the number of captures in n seconds. Note that a utilization instance may last far more than one second, so it is possible that a utilization instance is captured by the monitor more than once. Obviously, X follows the binomial distribution with parameter n and p, i.e., $X \sim B(n, p)$. The probability mass function is:

$$Pr(X = k) = \binom{n}{k} p^k (1-p)^{n-k} \tag{4.4}$$

The expected number of captures is $E(X) = np$. The probability that a secondary user is captured at least k times in n second utilization is:

$$Pr(X >= k) = 1 - \sum_{i=0}^{k-1} \binom{n}{i} p^i (1-p)^{n-i} \tag{4.5}$$

Figure 4.3 shows the relation between p and n when $Pr(X >= 5) = 0.95$. We can see that n decreases almost exponentially as p increases. Even when p is as small as 0.01, it just takes 300 seconds for the monitor to capture a secondary user at least 5 times.

[9]The secondary user could use any channel in the licensed spectrum.

Therefore, it dose not take a long time before the monitor obtains considerable amount of captures even the probability to capture a secondary user in one second is small.

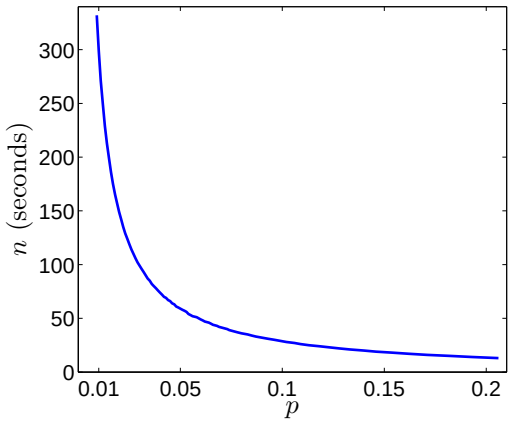

Figure 4.3: The relation between p and n when $Pr(X >= 5) = 0.95$.

Now we will show the probability with which a secondary user can be caught if it commits fraud. Suppose a secondary user has utilized the licensed spectrum for n seconds, and it only submits $n - m$ seconds of data. We also assume the primary user queries the monitor for k instances of random-check. The probability that the secondary user is caught committing fraud is:

$$Pr[caught] = 1 - \binom{n-k}{n-m-k} / \binom{n}{n-m} \tag{4.6}$$

Figure 4.4 shows the relation between m and $Pr[caught]$ when $n = 200$ and $k = 5$. We can see that the more data a secondary user chooses not to submit, the higher chance he will be caught. For instance, if a secondary user purposely chooses not to submit 20 second data (10% of total utilization data), it has about 41% of chance to be caught; if it chooses not submit 50 second data (25% of total utilization data), it has about 77% of chance to be caught. To refrain the secondary user from committing fraud, we can set a penalty higher than the gain such that a rational secondary user would not choose to commit fraud.

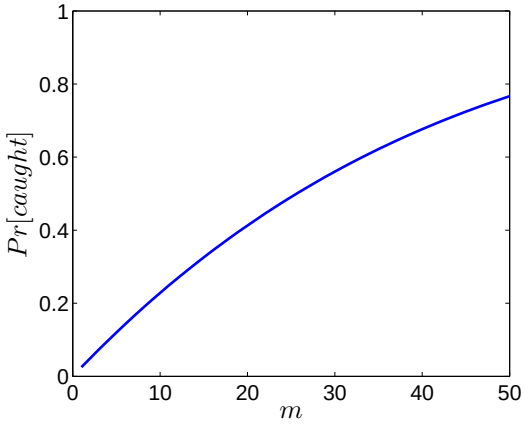

Figure 4.4: The relation between m and $Pr[caught]$ when $n = 200$ and $k = 5$.

4.7 Implementation and Evaluation

In this section, we will first present in detail our implementation of commitment scheme, signature scheme, and zero-knowledge proof, which are intensively used in our protocol. Then, we will evaluate the running time of our protocol.

4.7.1 Commitment Scheme

In the implementation of commitment scheme, we use Pedersen commitments [43]. Pedersen commitments are theoretically hiding and binding, relying on a group G of prime order p with generators g and h. Under this scheme, a commitment C_x to message $x \in Z_p$ has the form of $C_x = g^x h^{o_x}$, where o_x is an *opening* nonce chosen uniformly at random in Z_p. In our implementation, we use a group G based on bilinear pairing over elliptic curves [55]. Here we say that the commitment scheme is done in ECC (Elliptic Curve Cryptography) domain.

4.7.2 Signature Scheme

All messages are signed by the sender and verified by the recipient in our protocol. For messages between the monitor and the primary user, and messages from the secondary

user to the primary user, any unforgeable signature scheme can be used. In our implementation, we choose ECDSA signature scheme [21] in ECC domain. As to the signature scheme for the message from the primary user to the secondary user, especially the pricing policy, we utilize ECC version of C-L signature based on elliptic curve groups [5]. C-L signature scheme [5] can not only compute a signature on a commitment message but also provide zero-knowledge proof of knowledge of a signature on a committed value. These properties are leveraged to ensure the primary user that the price used by the secondary user during billing process is correct.

4.7.3 Non-Interactive Zero-Knowledge Proof

Recall that in the privacy-preserving protocol,the primary user sends the pricing policy in plain text together with pricing policy signed by CL-signature scheme to the secondary user. The secondary user then calculates the payment for each utilization instance ($< t_i, \triangle t_i, f >$) and sends the commitment of the payment back to the primary user. The secondary user needs to use non-interactive zero-knowledge proof to prove that: (1) t_i is in an interval $[T_{b_j}, T_{e_j}]$ such that $T_{b_j} \leq t_i \leq T_{e_j}$; (2) the payment fee_i is the product of p_j and $\triangle t_i$; and (3) p_j is indeed the price associated with $< T_{b_j}, T_{e_j}, f >$. Therefore, the basic building blocks are a non-interactive zero-knowledge proof that a committed value lies in an interval, a non-interactive zero-knowledge proof that a committed value is the product of two committed value, and a non-interactive zero-knowledge proof that a committed value has a CL-signature.

4.7.3.1 Interval Check

Suppose we need to prove that a committed value x lies in an interval $[a, b]$, i.e., $a \leq x \leq b$. It is equivalent to prove that $x - a \geq 0$ and $b - x \geq 0$. To prove that an integer $v >= 0$, we employ the non-interactive zero-knowledge proof by Groth [17]. It is well-known from

number theory that $4v + 1$ can be written as a sum of three squares if integer v is non-negative.

In the proof in [17], integer commitment scheme is constructed with special RSA modulus (here we say RSA domain), in which the order of generators is unknown. However, all the commitments in our protocol are constructed in ECC domain, where the order of generators is known. We need first to convert a commitment in ECC domain into a commitment in RSA domain. We then use non-interactive zero-knowledge proof to prove that a commitment in ECC domain and a commitment in RSA domain commit to the same value. Next we prove that the committed value in RSA domain is non-negative with the non-interactive zero-knowledge proof mentioned above.

4.7.3.2 Proof of Multiplication

Giving commitments of p_j, $\triangle t_i$, and fee_i, i.e., $C_{p_j} = g^{p_j} h^{o_{p_j}}$, $C_{\triangle t_i} = g^{\triangle t_i} h^{o_{\triangle t_i}}$, and $C_{fee_i} = g^{fee_i} h^{o_{fee_i}}$, we need to prove that $fee_i = p_j \triangle t_i$. We adopt the notation introduced by Camenisch and Stadler [6]. The proof is as follows:

$$
\begin{aligned}
&NIPK\{(fee_i, o_{fee_i}, \triangle t_i, o_{\triangle t_i}, p_j, o_{p_j}) : \\
&\qquad (C_{fee_i}, o_{fee_i}) = Commit(par, fee_i) \wedge \\
&\qquad (C_{p_j}, o_{p_j}) = Commit(par, p_j) \wedge \\
&\qquad (C_{\triangle t_i}, o_{\triangle t_i}) = Commit(par, \triangle t_i) \wedge \\
&\qquad fee_i = p_j \triangle t_i \\
&\}
\end{aligned}
$$

In the above notation, $NIPK$ stands for *Non-Interactive Proof of Knowledge*. The variables in the parenthesis, i.e., $fee_i, o_{fee_i}, \triangle t_i, o_{\triangle t_i}, p_j, o_{p_j}$, denote quantities whose knowledge is being proven, while all other values are known to the verifier.

4.7.3.3 Possession of a CL-signature

In the above proof of multiplication, it lacks of another piece of proof that p_j is the correct unit price that should be used. This is done by the proof of possession of a CL-signature on p_j. That is, the secondary user proves to the primary user that the former possesses a CL-signature computed by the latter on $< T_{b_j}, T_{e_j}, f, p_j >$ that states that p_j is the unit price to be paid for (T_{b_j}, T_{e_j}, f). We implement the proof of knowledge of a CL-signature on a commitment in [5].

4.7.4 Running time

We have implemented all the functionality required to the privacy-preserving protocol in C. We use both pbc library [35] and GMP library [16]. Our platform is Intel(R) Xeon(R) CPU at 1.02GHz on OPENSUSE operating system. We can measure the running time for each individual block, such as ECC commitment, integer commitment, CL-signature, proof of possession of CL-signature, proof of multiplication, and interval checking. All blocks, except interval check, are done in ECC domain; the key size is 160 bits (192 bit key for ECDSA). The interval check involves both ECC and RSA modulus; for ECC part, the key size is still 160 bits, and for RSA part, the key size is 2048. These choices of key sizes are believed to achieve high security. The running time for all fundamental blocks is shown in Table 4.1.

We also measure the running time for pricing policy generation, proving a bill, and verifying a bill, which is shown in Table 4.2.

We generate some synthetic data to simulate a real scenario. We assume that there is a secondary user that utilizes licensed spectrum in a day. This secondary user can use several channels while it can only use one at a time. We suppose that the time duration of utilization instances follows an exponential distribution, and the time duration between two utilization instances also follows an exponential distribution. We then generate different traces for a secondary user. For the pricing policy, we introduce three linear policies, each

Table 4.1: The running time of individual blocks.

	Key size	time (ms)
Sign a policy (ECC)	160	16
Verify a policy (ECC)	160	18
Generate commitment (ECC)	160	4.5
Verify commitment (ECC)	160	2.4
Prove interval check (ECC/RSA)	160/2048	30
Verify interval check (ECC/RSA)	160/2048	20
Prove possession of C-L signature (ECC)	160	20
Verify possession of C-L signature (ECC)	160	11
Prove multiplication (ECC)	160	4.9
Verify multiplication (ECC)	160	4.7
ECDSA (ECC)	192	1.2

Table 4.2: The running time of proof and verification.

	Key size	time per reading (ms)
policy generation	160	35
prove bill	160	100
verify bill	160	65
random checking	192	1.2

of which spans 8 hours. Table 4.3 shows the total time required for both the primary user and the secondary user for different traces for a billing period of one day.

Table 4.3: The running time of one-day billing.

# of utilization instances	PU (second)	SU (second)
12	0.84	1.3
17	1.1	1.8
24	1.6	2.5
37	2.4	3.8
48	3.2	5.0

Discussion: The running time is not a concern at all, since most of the operations can be processed in parallel. During a billing period, there may have many utilization

instances. However, they are independent of each other. As a result, on both PU and SU sides, data processing can be done in parallel in the granularity of utilization instance. Furthermore, even inside a utilization instance, there are still probabilities for parallel processing. For instance, on the SU side, all commitments can be done in parallel; on the PU side, the verification of commitments can be done in parallel.

4.8 Conclusion

Privacy is an important aspect of research in cyber-physical systems (CPS) and their communication networks. In this work, we improve the privacy of spectrum transactions in cognitive radio networks. We propose a privacy-preserving protocol to protect both entities in cognitive radio transactions. Specifically, we utilize a commitment scheme and zero-knowledge proof to preserve secondary users' privacy, and use a random-checking monitor to protect the primary users' interest. Furthermore, we formulate the monitoring problem and analytically propose efficient rules to control the random-checking monitor such that it can capture as many utilization instances as possible. Finally, we implement our protocol and evaluate the performance. To the best of our knowledge, it is the first work to address the privacy of secondary users in cognitive radio transactions.

Chapter 5

Conclusion and Future Work

Cyber-physical systems (CPS) play an important role in modern society in various areas, such as energy, aerospace, civil infrastructure, healthcare, manufacturing, transportation, and entertainment. This dissertation explores security and privacy issues in CPS and its communication networks, specifically in electric power systems and cognitive radio networks.

First, we utilized a typical CPS as an example to show the potential cyber attacks and how to defend against them. Specifically, we introduced the unidentifiable attack, which requires relatively less effort compared to undetectable attacks, but is still capable of misleading electric power systems to make incorrect operations. We then proposed an optimization framework to defend against this new cyber attack. For this type of attack, we investigated from both the view of an attacker and the view of the system operator, i.e, how to efficiently launch an attack and how to respond to an attack, respectively. We validated our optimization framework with extensive simulations by showing that our strategies are efficient and realistic.

Next, we investigated the potential threats in communication networks for CPS. We examined the cooperative spectrum sensing in cognitive radio networks, and pointed out that an existing framework that detects bad data injection is problematic and works poorly under deliberate data manipulation. We hence defined cooperative attacks, in which an

attacker injects self-consistent bad data, and proposed a modified combinatorial optimization identification algorithm to pinpoint the most likely compromised data. Finally, we demonstrated the feasibility of cooperative attacks, and evaluated our algorithm with simulations, showing that our proposed algorithm was able to effectively improve the existing framework.

Finally, we addressed an important privacy issue in CPS's communication networks. We investigated the spectrum transactions in cognitive radio networks. To protect the seller's interest and the buyer's privacy in spectrum sharing in cognitive radio networks, we designed and implemented a privacy-preserving scheme by utilizing a commitment scheme, signature scheme, and zero-knowledge proof. Our evaluation indicated that our scheme performed well.

Based on our understanding and experience, it is challenging to thoroughly secure a CPS. We hope and believe that the projects in this dissertation can serve as examples or guidelines in securing CPS and its communication networks. And we also envision that the following research directions can enrich and expand the security research in CPS and its communication networks:

- **Potential Attacks in CPS**

  In this dissertation, we only cover a typical CPS, i.e., the electric power system. There are various CPS in our society, which may face many potential vulnerabilities and attacks. Therefore, it is necessary to explore new vulnerabilities and design countermeasures for different CPS. Taking the electric power system as an example, besides bad data injection attacks, it also faces vertex and/or edge attacks[10]. The failure of a vertex or an edge can cause devastating damage to the whole power system. It is beneficial to have plans or countermeasures to respond to these edge/vertex attacks. Another example would be body sensor networks, which are emerging CPS that promise to enhance quality of life. Sensors in such networks

[10] The electric power system can be modeled as a graph, in which vertices are generators/consumers and edges are transmission lines.

usually measure sensitive and important physiological signals, and a successful attack may lead to not only a patient's privacy breach but also life-threatening risk. Thus, the security and privacy of body sensor networks are of great importance.

- **Communication Networks**

  CPS cannot work alone without communication networks, which may be as important as CPS itself. There are still various security and privacy issue in different communication networks in CPS, such as cognitive radio networks, Bluetooth, near field communication, and WiFi, even though extensive research has been conducted. Taking cognitive radio networks as an example, The Federal Communications Commission (FCC) recently has enforced database-driven cognitive radio networks, in which unlicensed users query a database to obtain availability information by submitting location-based requests, instead of sensing the presence/absence of licensed users directly. This may cause privacy information leakage for both unlicensed and licensed users. For instance, for an unlicensed user, when it submits a location based query, its location is inevitably leaked. Furthermore, an adversary can infer the active pattern of the licensed users by submitting multiple queries at different times. Therefore, it is in great demand to have an approach to preserve the privacy of both licensed and unlicensed users in database-driven cognitive radio networks simultaneously.

- **Data Security and Privacy in CPS**

  A CPS usually generates large amounts of data, and the straightforward question is how to securely and efficiently store all the data. Nowadays, technologies such as fog computing and cloud computing are very popular. I believe that integrating CPS data with fog/cloud computing is a promising and interesting direction. Another direction is privacy of CPS data. One can apply or design various privacy schemes on CPS data for different privacy purposes. For instance, it would be beneficial to release CPS data to the public in some cases, as long as the privacy of the data

is guaranteed. In recent years, the concept of differential privacy has gained much popularity. I envision that applying the concept of differential privacy and privacy preserving mechanisms to CPS data releases will be a promising research topic.

Bibliography

[1] Ian F Akyildiz, Brandon F Lo, and Ravikumar Balakrishnan. Cooperative spectrum sensing in cognitive radio networks: A survey. *Physical communication*, 4(1):40--62, 2011.

[2] J. Allen, H. Sauer, Lou Frank, and Patricia Reiff. Effects of the march 1989 solar activity. *EOS Transactions, American Geophysical Union*, 70(46):1479--1488, 1989.

[3] Eduardo N Asada, Ariovaldo V Garcia, and R Romero. Identifying multiple interacting bad data in power system state estimation. *IEEE Power Engineering Society General Meeting*, pages 571--577, 2005.

[4] Josep Balasch, Alfredo Rial, Carmela Troncoso, Bart Preneel, Ingrid Verbauwhede, and Christophe Geuens. PrETP: Privacy-preserving electronic toll pricing. In *USENIX Security Symposium*, volume 10, pages 63--78, 2010.

[5] Jan Camenisch and Anna Lysyanskaya. Signature schemes and anonymous credentials from bilinear maps. In *Advances in Cryptology--CRYPTO*, pages 56--72, 2004.

[6] Jan Camenisch and Markus Stadler. *Proof systems for general statements about discrete logarithms*. Citeseer, 1997.

[7] Ruiliang Chen, Jung-Min Park, and Kaigui Bian. Robust distributed spectrum sensing in cognitive radio networks. In *IEEE Conference on Computer Communications (INFOCOM)*, pages 1876--1884, 2008.

[8] Min Ding, Fang Liu, Andrew Thaeler, Dechang Chen, and Xiuzhen Cheng. Fault-tolerant target localization in sensor networks. *EURASIP Journal on Wireless Communications and Networking*, 2007(1):1--9, 2007.

[9] Omid Fatemieh, Ranveer Chandra, and Carl A Gunter. Secure collaborative sensing for crowd sourcing spectrum data in white space networks. In *IEEE Symposium on New Frontiers in Dynamic Spectrum Access Networks (DySPAN)*, pages 1--12, 2010.

[10] Omid Fatemieh, Ali Farhadi, Ranveer Chandra, and Carl A Gunter. Using classification to protect the integrity of spectrum measurements in white space networks. In *Proceedings of NDSS*, volume 11, 2011.

[11] Amos Fiat and Adi Shamir. How to prove yourself: practical solutions to identification and signature problems. In *Advances in Cryptology—CRYPTO*, pages 186--194, 1987.

[12] Thoshitha Gamage and Bruce McMillin. Nondeducibility-based analysis of cyber-physical systems. *Critical Infrastructure Protection III*, pages 169--183, 2009.

[13] Sorabh Gandhi, Chiranjeeb Buragohain, Lili Cao, Haitao Zheng, and Subhash Suri. A general framework for wireless spectrum auctions. In *IEEE Symposium on New Frontiers in Dynamic Spectrum Access Networks (DySPAN)*, volume 12, pages 22--23, 2007.

[14] Ghurumuruhan Ganesan and Ye Li. Cooperative spectrum sensing in cognitive radio networks. In *IEEE International Symposium on New Frontiers in Dynamic Spectrum Access Networks (DySPAN)*, pages 137--143, 2005.

[15] Stefano Gastoni, Glan Pietro Granelli, and Mario Montagna. Multiple bad data processing by genetic algorithms. In *IEEE Power Tech Conference Proceedings*, volume 1, 2003.

[16] Torbjörn Granlund et al. GMP, the GNU multiple precision arithmetic library. *Online: http://gmplib.org*, 1991.

[17] Jens Groth. Non-interactive zero-knowledge arguments for voting. In *Applied Cryptography and Network Security*, pages 467--482, 2005.

[18] Edmund Handschin, Fred C Schweppe, Jurg Kohlas, and Armin Fiechter. Bad data analysis for power system state estimation. *IEEE Transactions on Power Apparatus and Systems*, 94(2), 1975.

[19] Simon Haykin. Cognitive radio: brain-empowered wireless communications. *IEEE journal on selected areas in communications*, 23(2):201--220, 2005.

[20] Tao Jing, Xiuying Chen, Yan Huo, and Xiuzhen Cheng. Achievable transmission capacity of cognitive mesh networks with different media access control. In *IEEE Conference on Computer Communications (INFOCOM)*, March 25-30 2012.

[21] Don Johnson, Alfred Menezes, and Scott Vanstone. The elliptic curve digital signature algorithm (ECDSA). *International Journal of Information Security*, 1(1):36--63, 2001.

[22] Praveen Kaligineedi, Majid Khabbazian, and Vijay K Bhargava. Malicious user detection in a cognitive radio cooperative sensing system. *IEEE Transactions on Wireless Communications*, 9(8):2488--2497, 2010.

[23] John G Kappernman and Vernon D Albertson. Bracing for the geomagnetic storms. *IEEE Spectrum*, 27(3):27--33, 1990.

[24] Tung T Kim and H Vincent Poor. Strategic protection against data injection attacks on power grids. *IEEE Transactions on Smart Grid*, 2(2):326--333, 2011.

[25] Oliver Kosut, Liyan Jia, Robert Thomas, and Lang Tong. Malicious data attacks on SmartGrid state estimation: Attack strategies and countermeasures. *Proceedings of IEEE SmartGrid Comm*, pages 220--225, 2010.

[26] Edward A Lee. Computing foundations and practice for cyber-physical systems: A preliminary report. *University of California, Berkeley, Tech. Rep. UCB/EECS-2007-72*, 2007.

[27] Edward A Lee. Cyber physical systems: Design challenges. In *11th IEEE International Symposium on Object Oriented Real-Time Distributed Computing (ISORC),*, pages 363--369, 2008.

[28] Edward A Lee. CPS foundations. In *Proceedings of the 47th Design Automation Conference*, pages 737--742. ACM, 2010.

[29] Hongjuan Li, Xiuzhen Cheng, Keqiu Li, Chunqiang Hu, Nan Zhang, and Weilian Xue. Robust collaborative spectrum sensing schemes for cognitive radio networks. *IEEE Trans on Parallel and Distributed Systems*, 25(8):2190--2200, 2013.

[30] Hongjuan Li, Xiuzhen Cheng, Keqiu Li, Xiaoshuang Xing, and Tao Jing. Utility-based cooperative spectrum sensing scheduling in cognitive radio networks. In *IEEE Infocom Mini-Conference*, pages 165--169, April 14-19 2013.

[31] Husheng Li and Zhu Han. Catch me if you can: an abnormality detection approach for collaborative spectrum sensing in cognitive radio networks. *IEEE Transactions on Wireless Communications*, 9(11):3554--3565, 2010.

[32] Zhen Ling, Junzhou Luo, Wei Yu, Xinwen Fu, Dong Xuan, and Weijia Jia. A new cell counter based attack against Tor. *Proceedings of ACM Conference on Computer and Communications Security (CCS)*, 2009.

[33] Song Liu, Yingying Chen, Wade Trappe, and Larry J Greenstein. Aldo: An anomaly detection framework for dynamic spectrum access networks. In *IEEE Conference on Computer Communications (INFOCOM)*, pages 675--683, 2009.

[34] Yao Liu, Peng Ning, and Michael K Reiter. False data injection attacks against state estimation in electric power grids. *ACM Transactions on Information and System Security (TISSEC)*, 14(1):13, 2011.

[35] Ben Lynn. PBC library. *Online: http://crypto. stanford. edu/pbc*, 2006.

[36] Hyde M Merrill and Fred C Schweppe. Bad data suppression in power system static state estimation. *IEEE Transactions on Power Apparatus and Systems*, 90(6):2718--2725, 1971.

[37] Lamine Mili, Mania Ribbens-Pavella, and Thierry Van Cutsem. Bad data identification methods in power system state estimation-a comparative study. *IEEE Transactions on Power Apparatus and Systems*, PAS-104, 1985.

[38] Alexander W Min, Kyu-Han Kim, and Kang G Shin. Robust cooperative sensing via state estimation in cognitive radio networks. In *IEEE Symposium on New Frontiers in Dynamic Spectrum Access Networks (DySPAN)*, pages 185--196, 2011.

[39] Shridhar Mubaraq Mishra, Anant Sahai, and Robert W Brodersen. Cooperative sensing among cognitive radios. In *IEEE International Conference on Communications*, volume 4, pages 1658--1663, 2006.

[40] Joseph Mitola III and Gerald Q Maguire Jr. Cognitive radio: making software radios more personal. *IEEE Personal Communications*, 6(4):13--18, 1999.

[41] Alex Monticelli, Felix Wu, and Maosong Yen. Mutiple Bad Data Identification for State Estimation by Combinatorial Optimization. *IEEE Transactions on Power Delivery*, 1(3):361--369, 1986.

[42] Dusit Niyato and Ekram Hossain. Market-equilibrium, competitive, and cooperative pricing for spectrum sharing in cognitive radio networks: Analysis and comparison. *IEEE Trans on Wireless Communications*, 7(11):4273--4283, 2008.

[43] Torben Pryds Pedersen. Non-interactive and information-theoretic secure verifiable secret sharing. In *Advances in Cryptology—CRYPTO*, pages 129--140, 1992.

[44] Raluca A Popa, Hari Balakrishnan, and Andrew J Blumberg. VPriv: protecting privacy in location-based vehicular services. In *Proceedings of the 18th conference on USENIX security symposium*, pages 335--350, 2009.

[45] Zhengrui Qin, Qun Li, and Mooi Chuah. Unidentifiable attacks in electric power systems. In *Proceedings of IEEE/ACM International Conference on Cyber-Physical Systems*, pages 193--202, 2012.

[46] Zhengrui Qin, Qun Li, and Mooi Chuah. Defending against unidentifiable attacks in electric power grids. *IEEE Transactions on Parallel and Distributed Systems*, pages 1961--1971, 2013.

[47] Zhengrui Qin, Qun Li, and George Hsieh. Defending against cooperative attacks in cooperative spectrum sensing. *IEEE Transactions on Wireless Communications*, 12(6):2680--2687, 2013.

[48] Zhengrui Qin, Shanhe Yi, Qun Li, and Dmitry Zamkov. Preserving secondary users' privacy in cognitive radio networks. In *IEEE Conference on Computer Communications (INFOCOM)*, pages 772--780, 2014.

[49] Ragunathan Raj Rajkumar, Insup Lee, Lui Sha, and John Stankovic. Cyber-physical systems: the next computing revolution. In *ACM Proceedings of the 47th Design Automation Conference*, pages 731--736, 2010.

[50] Shansi Ren, Qun Li, Haining Wang, Xin Chen, and Xiaodong Zhang. Analyzing object detection quality under probabilistic coverage in sensor networks. *International Workshop Quality of Service (IWQoS)*, pages 107--122, 2005.

[51] Alfredo Rial and George Danezis. Privacy-preserving smart metering. In *Proceedings of the 10th annual ACM workshop on Privacy in the electronic society*, pages 49--60, 2011.

[52] Fred C Schweppe and Douglas B Rom. Power system static-state estimation, Part I II & III. *IEEE Transactions on Power Apparatus and Systems*, 89(1):125--130, 1970.

[53] Lui Sha, Sathish Gopalakrishnan, Xue Liu, and Qixin Wang. Cyber-physical systems: A new frontier. In *Machine Learning in Cyber Trust*, pages 3--13. Springer, 2009.

[54] Jianhua Shi, Jiafu Wan, Hehua Yan, and Hui Suo. A survey of cyber-physical systems. In *Wireless Communications and Signal Processing (WCSP), 2011 International Conference on*, pages 1--6. IEEE, 2011.

[55] Joseph H Silverman. *The arithmetic of elliptic curves*. Springer, 2009.

[56] Jin Teng, Boying Zhang, Xiaole Bai, Zhiming Yang, and Dong Xuan. Incentive-driven and privacy-preserving message dissemination in large scale mobile networks. *IEEE Trans on Parallel and Distributed Systems*, 25(11):2909--2919, 2013.

[57] Thierry Van Cutsem, Mania Ribbens-Pavella, and Lamine Mili. Hypothesis testing identification: a new method for bad data analysis in power system state estimation. *IEEE Transactions on Power Apparatus and Systems*, 103(11), 1984.

[58] Andreas Wächter and Lorenz T Biegler. On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming. *Mathematical Programming*, 106(1):25--57, 2006.

[59] Haodong Wang and Qun Li. Efficient implementation of public key cryptosystems on MICAz and TelosB motes. Technical Report WM-CS-2006-7, College of William and Mary, 2005.

[60] Haodong Wang, Bo Sheng, Chiu C. Tan, and Qun Li. WM-ECC: an elliptic curve cryptography suite on sensor motes. Technical Report WM-CS-2007-11, College of William and Mary, 2007.

[61] Haodong Wang, Chiu C Tan, and Qun Li. Snoogle: A search engine for the physical world. *IEEE Conference on Computer Communications (INFOCOM)*, pages 2056--2064, 2008.

[62] Xinbing Wang, Zheng Li, Pengchao Xu, Youyun Xu, Xinbo Gao, and Hsiao-Hwa Chen. Spectrum sharing in cognitive radio networks—an auction-based approach. *IEEE Trans on Systems, Man, and Cybernetics, Part B: Cybernetics*, 40(3):587--596, 2010.

[63] Wei Wei, Fengyuan Xu, and Qun Li. MobiShare: Flexible privacy-preserving location sharing in mobile online social networks. In *IEEE Conference on Computer Communications (INFOCOM)*, pages 2616--2620, 2012.

[64] Wei Wei, Fengyuan Xu, Chiu C. Tan, and Qun Li. Sybildefender: Defend against sybil attacks in large social networks. In *IEEE Conference on Computer Communications (INFOCOM)*, pages 1951--1959, 2012.

[65] Chunsheng Xin, Min Song, Liangping Ma, and Chien-Chung Shen. Performance analysis of a control-free dynamic spectrum access scheme. *IEEE Transactions on Wireless Communications*, 10(12):4316--4323, 2011.

[66] Kai Xing, Min Ding, Xiuzhen Cheng, and Shmuel Rotenstreich. Safety warning based on highway sensor networks. *IEEE Wireless Communications and Networking Conference*, 4:2355--2361, 2005.

[67] Dong Xuan, Riccardo Bettati, and Wei Zhao. A gateway-based defense system for distributed DoS attacks in high-speed networks. *Workshop on Information Assurance and Security*, 1, 2001.

[68] Qiben Yan, Ming Li, Feng Chen, Tingting Jiang, Wenjing Lou, Y Thomas Hou, and Chang-Tien Lu. Non-parametric passive traffic monitoring in cognitive radio networks. In *IEEE Conference on Computer Communications (INFOCOM)*, pages 1240--1248, 2013.

[69] Qiben Yan, Ming Li, Tingting Jiang, Wenjing Lou, and Y Thomas Hou. Vulnerability and protection for distributed consensus-based spectrum sensing in cognitive radio networks. In *IEEE Conference on Computer Communications (INFOCOM)*, pages 900--908, 2012.

[70] Richard F Yu, Helen Tang, Minyi Huang, Zhiqiang Li, and Peter C Mason. Defense against spectrum sensing data falsification attacks in mobile ad hoc networks with cognitive radios. In *IEEE Military Communications Conference*, pages 1--7, 2009.

[71] Yanling Yuan, Zuyi Li, and Kui Ren. Modeling load redistribution attacks in power system. *IEEE Transactions on Smart Grid*, 2(2):382--390, 2011.

[72] Yanling Yuan, Zuyi Li, and Kui Ren. Quantitative analysis of load redistribution attacks in power systems. *IEEE Transactions on Parallel and Distributed Systems,*, 99, 2012.

[73] Tevfik Yücek and Hüseyin Arslan. A survey of spectrum sensing algorithms for cognitive radio applications. *IEEE Communications Surveys & Tutorials*, 11(1):116--130, 2009.

[74] Boying Zhang, Jin Teng, Xiaole Bai, Zhimin Yang, and Dong Xuan. P3-coupon: A probabilistic system for prompt and privacy-preserving electronic coupon distribution.

In *IEEE International Conference on Pervasive Computing and Communications*, pages 93--101, 2011.

[75] Yanxiao Zhao, Min Song, and Chunsheng Xin. A weighted cooperative spectrum sensing framework for infrastructure-based cognitive radio networks. *Computer Communications*, 34(12):1510--1517, 2011.

[76] Yanxiao Zhao, Min Song, Chunsheng Xin, and Manish Wadhwa. Spectrum sensing based on three-state model to accomplish all-level fairness for co-existing multiple cognitive radio networks. In *IEEE Conference on Computer Communications (INFOCOM)*, pages 1782--1790, 2012.

[77] Ray Daniel Zimmerman, Carlos Edmundo Murillo-Sánchez, and Robert John Thomas. MATPOWER: Steady-state operations, planning, and analysis tools for power systems research and education. *IEEE Transactions on Power Systems*, 26(1):12--19, 2011.