Data Security and Privacy in Smart Grid

A Dissertation Presented for the

Doctor of Philosophy

Degree

The University of Tennessee, Knoxville

Yue Tong

August 2015

© by Yue Tong, 2015
All Rights Reserved.

To my wife and my parents

Acknowledgements

My deepest appreciation goes to Prof. Jinyuan Sun, whose unconditional support and insightful guidance were invaluable during the course of my doctorate study. I am also indebted to Prof. Kevin Tomsovic, Prof. Kai Sun, Prof. Yulong Xing and Prof. Xueping Li for their time and patience to read my dissertation, to attend my defense, and to provide meticulous comments that tremendously improved the quality of this work.

I gratefully appreciate the financial support from CURENT and Department of EECS of the University of Tennessee.

I also would like to thank my friends and labmates: Cheng Cheng, Yiran Wang, Yongzhi Wang, Xiangyu Niu, Yingyuan Yang, Cindy Yao, Jordan Deyton, Xue Li, Qinran Hu, Bing Wang, Fengkai Hu, Xiaohu Zhang. Their supports and encouragements helped me to carry on in spite of setbacks throughout the entire doctorate program.

Finally, and importantly, I would like to express my long due gratitude to my family, especially my wife, Yi, and my parents for their moral supports and understanding over the past years, with which I conquered many difficulties in the long process of my doctoral study.

Abstract

This dissertation explores novel data security and privacy problems in the emerging smart grid.

The need for data security and privacy spans the whole life cycle of the data in the smart grid, across the phases of data acquisition, local processing and archiving, collaborative processing, and finally sharing and archiving. The first two phases happen in the private domains of an individual utility company, where data are collected from the power system and processed at the local facilities. When data are being acquired and processed in the private domain, data security is the most critical concern. The key question is how to ensure the data availability and integrity for their uses in the private domain and later in the public space. Hence, we devote Chapter 2 to the vulnerability assessment of the phasor network, which deals with the data security in the private domain.

After the initial two phases, some of the data enter the public space. In the public space, data privacy is another challenging requirement on top of data security. We select three aspects of data privacy issues in the smart grid to illustrate the how the new technologies like cloud computing, homomorphic encryption, and advanced cryptography can form a synergy to solve these problems. Chapter 3 introduces a solution of secure outsourcing of dynamic simulations leveraging cloud computing; Chapter 4 presents a multiparty privacy-preserving spectrum estimation; Chapter 5 put forward a new mechanism for secure storage and sharing of power flow data for situational awareness.

Table of Contents

1 Introduction **1**

2 Vulnerability Assessment of Phasor Networks **11**

2.1 Introduction 11

2.2 Related Works 14

2.3 Preliminary 15

2.3.1 PMU and Phasor Network 15

2.3.2 IEEE 37.118 Protocol 17

2.3.3 A small Scale Prototype of Synchrophasor Network 19

2.4 Vulnerability Assessment 20

2.4.1 Procedures of Penetration Testing 21

2.4.2 Manual Penetration Testing Techniques 22

2.5 Results of Vulnerability Assessment 24

2.5.1 Reconnaissance 24

2.5.2 Vulnerability Exploration 25

2.5.3 Exploit Development: A Practical Exploit of IEEE C37.118: Data Stream Hijacking 40

2.6 Countermeasures and Mitigations 45

3 Secure Outsourcing of Dynamic Simulations in Power Grid Leveraging Cloud Computing **48**

3.1 Introduction 48

3.2 Related Works . . . 50
3.3 Problem Statement . . . 51
3.3.1 System Model and Threat Model . . . 51
3.3.2 Background on Power System Dynamic Simulations . . . 53
3.4 Secure Outsourcing Schemes . . . 55
3.4.1 Protecting the Simulation Results with the Disguising Technique 55
3.4.2 Ensure Input Privacy Through Code Obfuscation . . . 62
3.4.3 Framework . . . 64
3.4.4 Experiment Results . . . 65
3.5 Performance analysis . . . 67
3.6 Discussions . . . 69

4 Privacy-preserving Spectral Estimation of PMU Data for the Monitoring of Inter-area Oscillations 71

4.1 Introduction . . . 71
4.2 Related Works . . . 75
4.3 Preliminaries . . . 76
4.3.1 Fourier Transform . . . 76
4.3.2 Paillier Cryptosystem . . . 76
4.3.3 Monitoring Oscillation in Power System . . . 77
4.4 Our Proposed Scheme . . . 78
4.4.1 Problem Formulation . . . 78
4.4.2 Monitoring Inter-area Oscillations . . . 80
4.4.3 Privacy Preserving Multiparty Spectral Estimation . . . 81
4.4.4 Optimization . . . 86
4.5 Security Analysis . . . 86
4.6 Complexity Analysis . . . 87
4.6.1 Complexity Analysis . . . 87
4.7 Performance Evaluations . . . 90

4.7.1 Stress Testing 90
4.7.2 Test with WECC 179-bus Power System 91

5 Secure Data Sharing for Situational Awareness 95
5.1 Introduction 95
5.2 Related Works 97
5.3 System Model 98
5.3.1 Network Model 98
5.3.2 Data Model 98
5.3.3 Threat Model 100
5.3.4 Security Goals 100
5.4 S^3A Security Architecture 101
5.4.1 Scheme Overview 101
5.4.2 Secure and Unlinkable Data Storage 102
5.4.3 Evolving Key Based Flexible-Grained Access Control 107
5.5 Security Analysis 112
5.6 Performance Evaluation 113

6 Conclusions and Future Work 115

Bibliography 117

Vita 133

List of Tables

2.1 Vulnerabilities to be validated in the *Exploration* phase 26
2.2 Command bytes of IEEE C37.118 command frame 31

4.1 Notations used in the the complexity analysis 88
4.2 Time for execution of Algorithm 1 before and after optimizations . . 91

5.1 Runtime of common operations of the proposed data sharing mechanism for situational awareness 114

List of Figures

1.1 Conceptual model of smart grid. Source: Office of the National Coordinator for Smart Grid Interoperability (2010) 2

1.2 The life cycle of the data in the smart grid and their needs for data security and privacy protections at different phases 5

2.1 Typical SCADA network, Source: Government Accountability Office Report. GAO-040354 United States Government Accountability Office (2004) 12

2.2 Synchrophasor network model 16

2.3 The structure of a PMU data block 17

2.4 A typical data transmission process by C37.118 protocol 18

2.5 A prototype of the synchrophasor network for penetration testing . . 19

2.6 Wireshark captured IEEE C37.118 CONFIG-2 frame revealing all configuration information about a PMU 24

2.7 Seven layers of OSI model and example protocols for each layer . . . 25

2.8 Captured command frame: Data transmission on 29

2.9 Captured command frame: Data transmission off 29

2.10 Captured command frame: request CONFIG-2 29

2.11 Captured Data frame 30

2.12 Legitimate Command Frame to turn on the data transmission 31

2.13 Wireshark capture of the forged command frame 32

2.14 The PMU's console after the command frame spoofing attack 33

2.15 Data frames cannot be associated with the right configuration frame under a configuration spoofing attack 34
2.16 Received phasor measurements during the data frame spoofing attack. Red shaded frames are injected by the attacker 35
2.17 179 bus power system 41
2.18 Demonstration of a data stream hijacking attack on a WECC 179-bus system with the small scaled phasor network prototype 42
2.19 Normal angle difference over simulation time 43
2.20 Time line for replaying attack in an IEEE C37.118 based phasor network 44
2.21 Wireshark capture showing the replay attack 44
2.22 Manipulated angle difference by data stream hijacking over simulation time 45

3.1 Ideal system model for secure computation outsourcing 51
3.2 Client side UI for secure outsourcing of dynamic simulations 66
3.3 Original, disguised, and recovered curves of machine angles and machine speeds 67

4.1 Two area four generator power system from Kundur et al. (1994) . . 78
4.2 System Model of privacy-preserving processing of PMU data 79
4.3 Coherent generator clusters in WECC 179 bus system 92
4.4 Encrypted spectrograms of clusters' angle 93
4.5 Decrypted spectrograms of clusters' angle 94
4.6 Decrypted spectrograms of clusters' angle 94

5.1 Data Model. 99
5.2 Construction of Secure Index 103
5.3 An example of the access-pattern-preserving SI construction process. 105

Chapter 1

Introduction

Recent years have witnessed the rapid development of smart grid technologies. Smart grid, as seen in Figure 1.1, is defined in Office of Electicity Delivery and Energy Reliability, Department of Energy (2009) as:

... a class of technology people are using to bring utility electricity delivery systems into the 21st century, using computer-based remote control and automation ...

The core idea is that by leveraging more advanced two-way communication infrastructures, the smart grid provides salient features such as improved interoperability, better efficiency, enhanced reliability, and so on.

Compared with legacy power grids, the smart grid boasts more advanced communication infrastructures, such that more and a wider variety of data are collected and used for optimizing the control and supporting decisions making during the operation and planning of the power grid. For example, with the advanced metering infrastructure (AMI), real-time power loads can be monitored, and the demand response can be more precise and timely. Take another example, the phasor network, a.k.a Phasor Measurement Units (PMU) network, enables direct and accurate measurements of the power grids' real-time state in different locations, providing better situational awareness and decision makings for operators and power system planners. Data, of a wide variety and from a host of various entities, play a

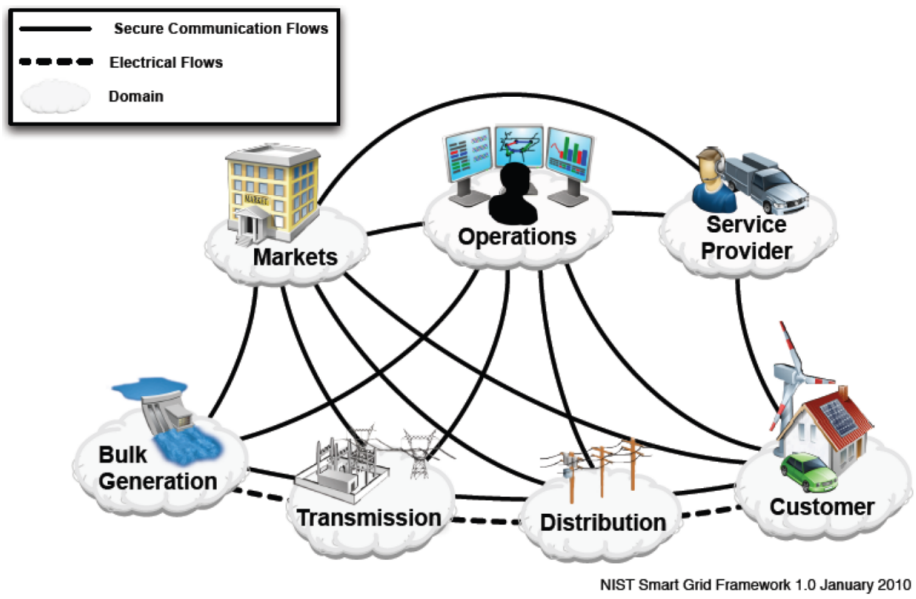


Figure 1.1: Conceptual model of smart grid. Source: Office of the National Coordinator for Smart Grid Interoperability (2010)

vital part in the emerging smart grid. As the smart grid's dependence on data keeps growing, data security and privacy become a critical concern. In reality, data security and privacy breaches continue to occur once a while. Most recently, according to Silver-Greenberg et al. (2014), the famous financial institution J.P. Morgan Chase was reportedly intruded by attackers, causing the leak of customer contact information and stealing of employees' credentials. More relevantly, Miller and Rowe (2012) surveyed numerous cyber-security incidents in critical infrastructures caused by the data security and privacy issues. These incidents led to immense losses with respective to efficiency and finance of the affected entities. For this reason, we are motivated to explore data security and privacy issues associated with a variety of smart grid technologies. However, it is worth noting that in this dissertation, we restrict our discussion to the data security and privacy at the level of entities in the bulk generation, transmission, and, control in the smart grid such as utility companies, ISOs, area control centers, and so on. In contrast, the security and privacy of the consumer data in the smart grid are widely investigated in the literature Borges et al. (2014); Thoma et al. (2013); Baharlouei and Hashemi (2014); Elaine Shi et al. (2011).

Data security in our discussion consists of two aspects: data availability and integrity. Availability means that the data should be accessible when needed. It is especially important for time-sensitive applications such as real-time situational awareness. In the network security research, data availability is primarily subject to denial-of-service (DoS) attacks if the underlying communication network fails to defend against such attacks. DoS attacks attempt to slow down, obstruct, or corrupt the normal network communications either by saturating the resources of victim machines so that services to legitimate users are affected, or by blocking the communication channel (*e.g.* taking down routers or jamming the wireless channel) between the legitimate users and the application server Kaufman et al. (2002) . Integrity dictates that the data being used should be intact *i.e.* accurate, complete, and genuine. The data integrity can be undermined by compromising and intruding the underlying data transferring and storage facilities so as to make unauthorized modifications to the data. For many applications, especially power system control algorithms, which uses data from remote sources for decision-making, wrong data could lead to misinformed decision and devastating consequences. For example, by falsifying the measurements of meters that have been compromised, the false data injection attack Liu et al. (2009) attempts to mislead the state estimation process to make falsified estimations. Degradation of data security can seriously affect the overall reliability and efficiency of the smart grid. The challenge of ensuring data security in the smart grid lies in that the smart grid involves many new technologies and standards with stricter requirements on reliability and low latency as compared to the conventional data communication network. The knowledge accumulated regarding the data security in smart grid is also much less than that of the data security in the more often used conventional Internet. Dealing with data security in the smart grid requires new ways of thinking and good understandings of both the computer network technologies and power systems.

Existing works on the aspect of data security in the smart grid such as Mo et al. (2012); Wang and Lu (2013); Yan et al. (2012a); Lu et al. (2010); Metke and

Ekl (2010); Liu et al. (2009) either provide high-level overviews of network security challenges in the context of the smart grid and point out potential weaknesses in communication infrastructures of the smart grid, or focus on analyzing aftermaths of data security breaches from the angle of the power system, assuming the presence of effective attack vectors and techniques. However, little information detailing comprehensive assessing methodologies and results regarding the data security of specific smart grid technologies can be found in the current literature. In this dissertation, we attempt to address this limitation by using the phasor network as an example to assess and demonstrate the vulnerabilities existing in the current standards and technologies employed by the smart grid.

Data privacy, in this dissertation, refers to that data should not be disclosed to any party without the authorization of the data owner, even in situations where the data are intended to be shared. Just as *North American Electricity Reliability Corporation Confidentiality Agreement for Electric System Operating Reliability Data* North American Electric Reliability Corperation (2009) points out that *"the operational reliability data may contain proprietary information and unequal access to operational reliability data may result in unfair advantages and disadvantages in the electricity market"*, in order to maintain a cooperative and sustainable ecosystem for the smart grid, ensuring data privacy is equally important to data security. Consider that if a particular smart grid technology requires sensitive data from a host of entities but provides no guarantee of privacy, prospect users are likely to restrain from using it despite the benefits the technology promises. Dealing with data privacy in the smart grid is challenging because of two reasons. First, how to satisfy the need for information while minimizing the disclosure of private data is a non-trivial consideration. Second, for time-critical applications like wide area situational awareness that requires private data from individual utility companies, there appears no easy answer to meet the strict time constraint and preserve the data privacy simultaneously.

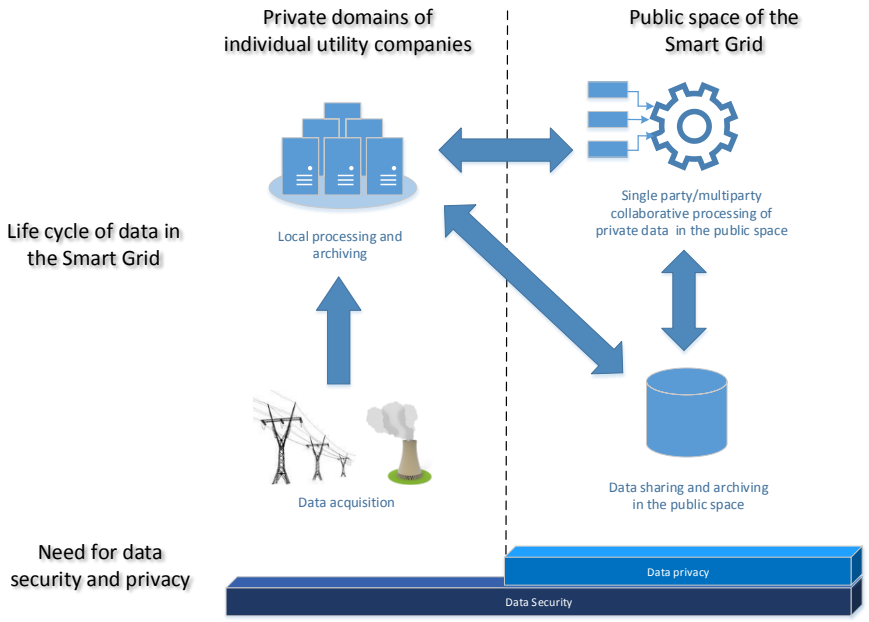


Figure 1.2: The life cycle of the data in the smart grid and their needs for data security and privacy protections at different phases

The majority of the research on the data privacy in the smart grid by Khurana et al. (2010b); Borges et al. (2014); Lu et al. (2010); Thoma et al. (2013); Baharlouei and Hashemi (2014); Elaine Shi et al. (2011) focuses on the privacy of consumer data. While obviously an important topic, it covers only a fraction of the data privacy issues in the smart grid. In this dissertation, we try to expand the scope of research of data privacy in the smart grid. Specifically, we investigate the data privacy issues in a host of scenarios involving the bulk generation, transmission, and operations.

The need for data security and privacy spans the whole life cycle of the data in the smart grid, with, however, different emphasis on distinct phases of the life cycle. As shown in Figure 1.2, any data in the smart grid go through a life cycle consisting of *up to* four phases: data acquisition, local processing and archiving, collaborative processing, and sharing and archiving. The first two phases happens in the private domains of individual utility companies, where data are collected from the power system and processed at the local computing facilities. The data acquired in this

phase range from grid data, utility data, market data, and so forth as described in Huang and Guttromson (2005). After the initial two phases, some of the data enter the public space. In the public space, data contributed by different utility companies may directly go to the sharing and archiving phase without further processing. For example, the power flow data from various utility companies are used for wide area situational awareness. Alternatively, some data may need further processing in the phase of collaborative processing in the public place. Collaborative processing occurs when parties other than the data's owner take part in the processing. It is desirable when the computation burden is so heavy that offloading it to an entity with abundant computational resources, such as the cloud, can achieve better efficiency and economy. Another reason is that if the processing involves private data from multiple parties, for the fairness and data privacy, the processing is delegated to a third party. The processing results eventually return the private domain for local usage, or enter the sharing and archiving phase in the public space, from which the utility companies, ISOs, control centers may request data.

When data are being acquired and transferred in the private domain, data security is the most critical concern since only a single utility company is concerned. The key question is how to assure the data availability and integrity for its uses in the private domain and later in the public space. Nevertheless, data security alone is insufficient once data enters the public space of the smart grid, which involves fragmented components and entities of the smart grid both providing and requesting data. These entities may include, utility companies, service providers, equipment vendors, computing and data storage service providers, regional ISOs, control centers, government authorities. For this reason, in the public space, data privacy is another challenging requirement on top of data security.

Following the typical life cycle of data in smart grid, from initial acquisition and local processing, to processing in the public space, and finally sharing and archiving, we organize the remaining chapters as follows:

- Chapter 2 deals with vulnerability assessment of the phasor network (data security in the private domain)
- Chapter 3 introduces a solution of secure outsourcing of dynamic simulations leveraging the cloud computing (data privacy in the single party secure computing)
- Chapter 4 presents a multiparty privacy-preserving spectral estimation (data privacy in multiparty secure computing)
- Chapter 5 put forward a new mechanism for secure storage and sharing of power flow data for situational awareness (data privacy in multiparty data sharing and archiving)

Chapter 2 deals with the vulnerability assessment of the phasor network. Experience with conventional IT communication networks reveals that many data security issues are caused by the flawed design, implementation, or configuration of the underlying communication network Kaufman et al. (2002); Stallings (1995). To avoid such issues, it is of great importance to examine the security of the communication infrastructure in smart grid. In Chapter 2, we focus on the vulnerability assessment of phasor networks, which are new and increasingly important components of the smart grid. Currently, there lack comprehensive examinations of the security of phasor networks in the literature. Vulnerability assessment is a defense activity by deliberately infiltrating a communication network with the intention to find the security weaknesses before the attackers can do so and make real damages Centre for the Protection of National Infrasctructure and Department of Homeland Security (2010). Our vulnerability assessment explores the vulnerabilities existing in the phasor network, by both automated and manual assessment techniques. We then test the feasibility of various attacks unique to the phasor network, by exploiting the found vulnerabilities. The attacks to be examined include, code injection, credential theft, false data injection, and DoS attacks. For each feasible attack, we report the

how to exploit the associated vulnerabilities, and discuss possible defense mechanisms. We will also develop an open-sourced automated tools to facilitate future vulnerability assessment of other phasor networks.

We start the discussion of the data privacy issues in Chapter 3, where we consider a minimal scenario: a utility company wishes to outsource the power system dynamic simulations to a semi-honest cloud but also would like to have privacy protection. In power system operations and researchers, dynamic simulations are a widely used to tool to investigate the power system's dynamic behavior under certain conditions. However, it incurs substantial computation burden due to the need to solve a large amount of Differential Algebraic Equations (DAEs). Recent endeavors Litvinov (2014); ISO New England (2014) of leveraging the power of cloud computing to conduct dynamic simulations have shown an immense potential of having faster, more efficient, and less expensive dynamic simulations by the outsourcing the computation. Outsourcing the computation, however, puts the data privacy at risk, since it entails private computing over a public platform. As a consequence, with these privacy concerns remained unresolved, the acceptance of the new computing paradigm is open to question. We thus present a solution to cope with the privacy concerns with respective to the outsourcing of dynamic simulations. In the proposed solution, we make use of both the disguising technique to disguise the state variables as well as software obfuscation technique to "encrypt" both the models and the software simulator before sending them out to the cloud. By doing so, neither the input sensitive information nor the final simulation result will be exposed to the cloud while the majority computation burden is off-loaded to the cloud. We fully implemented the proposed scheme, and it was shown that our scheme is capable of limiting the adversary's capability to obtain sensitive information during the course of outsourcing. The limitation of the proposed scheme towards the end of the chapter.

Next in Chapter 4, we extend the discussion of the data privacy to a more complicated multi-party scenario of collaborative processing of private data in the smart grid. The scenario deals with multiple mutually distrusting parties wishing

to obtain the result of a particular processing method, say aggregation, average, FFT, *etc.* , which requires private data from all the parties; the participating parties, however, do not wish to disclose their private input to neither platform that executes the processing method nor to their peer participants. Accordingly, besides the privacy protection against the untrusted platform, which is the primary concern of the last chapter, this chapter also takes into account the privacy protection against the peer data providers. Specifically, in Chapter 4, we investigate the privacy preserving multiparty spectral estimation using ambient synchrophasors for the monitoring of inter-area oscillations. Recent researches by Sun et al. (2012); Vanfretti et al. (2013); Chakrabarti et al. (2009); Brincker et al. (2001). show that it is a practical approach to continuously track the frequency domain of ambient synchrophasors for the monitoring purpose. However, as inter-area oscillations usually involve multiple utility companies in concerned areas, the method requires synchrophasor data from a number of different trust domains, which raises data privacy concerns. To address these concerns, we propose a novel scheme that enables privacy-preserving multiparty spectral estimations, which conducts spectral estimation directly over the encrypted synchrophasor data, by employing the homomorphic Paillier cryptosystem, in order to limit privacy leakages. The scheme also facilitates the access control in the resulted frequency domain to strengthen the privacy protection. Design choices on optimizing the original design in order to boost the performance are also discussed. Security analysis and performance evaluations are carried out. The proposed privacy-preserving multiparty spectral estimation is also tested with a simplified model of the WECC 179 bus system. Relevant results with respective to feasibility, scalability, and latency are provided towards the end of the chapter.

Finally, Chapter 5 proposes a privacy preserving data management mechanism for the smart grid. Different from previous chapters, which aims at fortifying the data privacy during the processing of sensitive data over a untrusted environment, the problem this chapter is concerned with is the data privacy issues during the data sharing and archiving. We are motivated by that the wide area situational

awareness of a complex power system requires power flow data from multiple competing and distrusting utility companies in the concerned area EIOC (2013); thus, a data sharing mechanism that enables efficient data sharing and archiving as well as robust privacy protection is highly desirable. As the data sharing mechanism is intended to be used for situational awareness, in the design, we especially consider how to simultaneously satisfy the requirements for the privacy preservation and stringent time constraints in case the power flow data are to be used in an emergency. Ultimately, the proposed data sharing mechanism empowers concerned utility companies with flexible access controls over the amount and categories of the data is to be shared. In addition, the trade-off between the performance and privacy preservation is adjustable. Accordingly, proposed scheme can well adapt to the both data sharing in the usual situation, which prefers high privacy protection over the performance, as well as data sharing during an emergency, where the performance is the top concern. Security analysis and performance evaluations will be carried out with the power flow data of IEEE standard test systems University of Washington (2014).

As the concluding remarks for this introduction, the transition from the legacy power grid to the smart grid entails a closer integration of new information technologies and inevitably raises concerns about data security and privacy. However, data security and privacy issues can be addressed with careful designs and implementations, as demonstrated by the four data security and privacy problems we discussed in this dissertation.

Chapter 2

Vulnerability Assessment of Phasor Networks

2.1 Introduction

The smart grid is introducing many salient features such as wide-area situational awareness, precise demand response, substation automation. These features are enabled by data communication networks that facilitate the collection, transfer, and processing of a wide variety of data regarding different components of the smart grid. For example, smart meters collect granular residential energy consumptions, which get aggregated through the advanced metering infrastructure (AMI), the communication network for the smart meter data NETL Modern Grid Strategy (2008), for the real-time load monitoring and the fast demand response. The substation automation system (SAS) allows for commands that trigger the preventative controls such as "tripping" relays, to be transferred within a substation and across substations, using the underlying data communication network compliant to IEC 61850 International Electrotechnical Commission (2014). As a result, the smart grid's heavy dependence on data inevitably poses a great challenge to ensure the data integrity and authenticity.

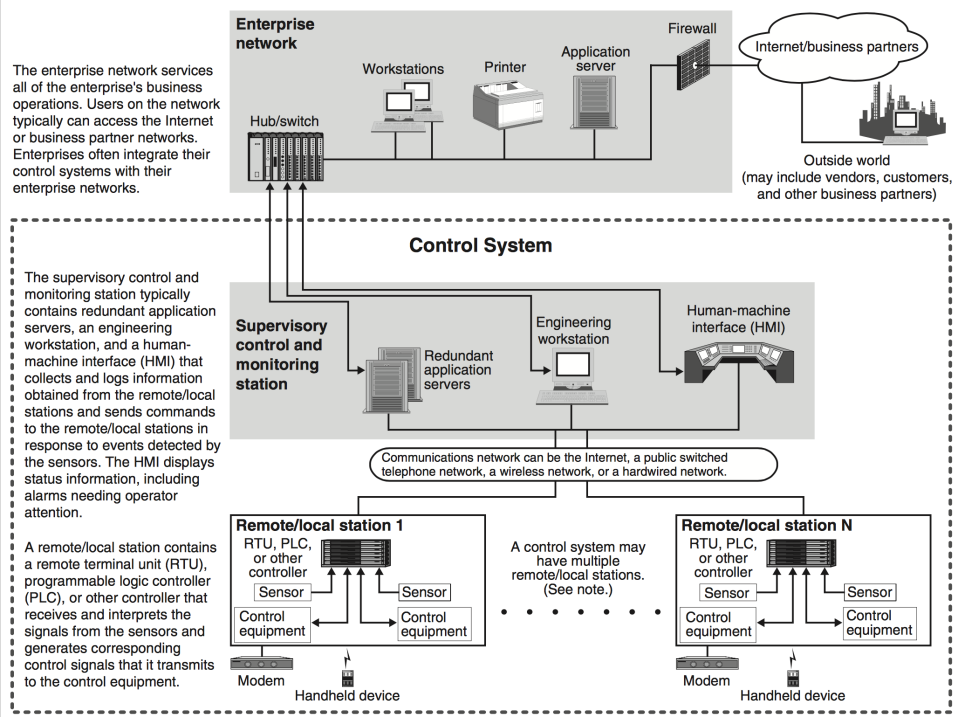


Figure 2.1: Typical SCADA network, Source: Government Accountability Office Report. GAO-040354 United States Government Accountability Office (2004)

In the meanwhile, another trend of "de-isolation" is also on the rise along with increasing adoption of computer-based communication networks in the smart grid. For decades, the legacy networks rely on its physical isolation from the outside world, *i.e.* the *air-gappedness*, to provide the security protection. However, in the past decade, in response to the demands for automation and information sharing, the boundary between the internal communication network of the power grid and the Internet has been increasingly blurring Young (2014b,a). Figure. 2.1 United States Government Accountability Office (2004) illustrates the how a typical the power system control network gets physically connected to the Internet in today's power grid system. As such, even though with defending mechanisms like firewalls deployed, the internal network can no longer be deemed physically isolated. Additionally, the experience with information security in common computer network reveals that flawed designs, implementations, and configurations of the communication network introduce vulnerabilities. These vulnerabilities open opportunities for attackers to launch cyber

attackers, leading to data security breaches like the security breach of the Chase bank Silver-Greenberg et al. (2014) and cases in Miller and Rowe (2012) affecting millions of people. As more and more components in the smart grid are adopting the same class of network technologies, we need to assess the vulnerabilities of these components, before trusting and assigning critical responsibilities to them. The assessments should include a variety of aspects including standards, software implementations, network configurations.

In this chapter, we are motivated to assess vulnerability and enhance the network security of the phasor network through penetration testings. We choose to investigate phasor network because of its indispensable role in the future smart grids as well as that the synchrophasor network has not been under comprehensive vulnerability assessment in existing literature. In this research effort, we attempt to gain more insights with respective to the cyber security of the current phasor network technologies by exploring, validating, and demonstrating vulnerabilities. Based on the results, we also present best practices and practical recommendations to defend against the identified vulnerabilities.

In the rest of the chapter, we will first revisit existing research with respective to the cyber security of phasor network in Section 2.2. We will also review the phasor network about its concept, fundamentals, and application in Section 2.3. Especially, a small prototype of the phasor network will be introduced for penetration testing in Section 2.3.3. Section 2.4 covers the details of the penetration testing, including the procedures of the penetration testing, testing techniques, and a list of potential vulnerabilities to be explored in the research effort. Furthermore, with the found vulnerability, we present a practical exploit of the C37.118 protocol with the benchmark WECC 158 model in the Section 2.5.3. Section 2.6 presents the countermeasures to defend against the vulnerabilities.

2.2 Related Works

The vulnerability assessment specific to the phasor network has been considered in previous research.

Liu et al. (2009) points out that falsified PMU measurement data can mislead the state estimation of the smart grid without being caught by the bad data detection. This vulnerability may cause serious consequences to the operation of the power grid.

As one of the earliest inquiries specialized in the cyber security of synchrophasor network, Stewart et al. (2010) investigated two aspects of the cybersecurity issue of synchrophasor technologies, *i.e.* 1) the substation security, which is concerned with the security threat where the attacker penetrates the phasor network through the substation network, and 2) the information security, which deals with the confidentiality, integrity, and availability of the synchrophasor data flowing out of the substation. Numerous practical recommendations for mitigating the investigated security problems are proposed. These include adopting multi-layered architecture to place PMU multiple layers deep in the network, using unidirectional synchrophasor streams (spontaneous mode) to defend against protocol manipulation, limiting remote access, regulating exposing ports and services with firewalls, using VPNs or link encryption to ensure the confidentiality and integrity. The work also conducts two test cases, *i.e.* network scan and man-in-the-middle attack to demonstrate the vulnerabilities of an unprotected phasor network. This work discovers some of the most serious vulnerabilities associated with the phasor network. It, however, lacks a systemic and comprehensive vulnerability assessment.

Morris et al. (2011) focuses on the network congestion testing and protocol mutation testing for the phasor network. They test PMU devices under a high volume of network traffic and send the device mutated packets, and observe their impact on the device. According to their result, all PMUs under test became unresponsive under an excessive volume of network traffic. Additionally, mutated packets were also found to cause individual network services or applications to crash. In their follow-up work

in Sprabery et al. (2013), the author proposed a rule-based intrusion detection method to detect malformed IEEE C37.118 frames injected to the phasor network.

Beasley (2014) examines the security of the phasor network in the context of the benchmark IEEE 68 bus (New England/New York) power system with real-time grid simulators and SEL PMU devices. The research successfully replicates the results of Stewart et al. (2010). It also discusses the classification of network traffic using side-channel information in cases where phasor network employs encryption to defend against traffic eavesdropping and manipulation. The paper suggests that further testing needs to take place to get better understanding of the cyber security conditions of the phasor network.

Beasley et al. (2014b) provides a survey of a wide spectrum of cyber attacks targeted at the phasor network, ranging from interruption, fabrication, to modification, reported in previous research Bou-Harb et al. (2013); Yan et al. (2012b); Campbell (2011); D'Antonio et al. (2011); Shepard et al. (2012); Baumeister (2010). This work also points out open research topics that require further investigation, including vulnerabilities of side-channels, replay attack, the fuzzing test of the phasor network.

2.3 Preliminary

2.3.1 PMU and Phasor Network

Phasor network is widely recognized as one of the key enabling technologies for the future smart grid Office of Electicity Delivery and Energy Reliability, Department of Energy (2009); North America Electric Reliability Corporation, NERC (2011). It empowers the power system operators with 1) real-time support of grid operations and decisions, 2) improve system planning and analysis, and 3) response-based control applications. Take the 2003 US-Canada blackout, if the increasing angle difference were noticed and proper measures were taken, the accident would have been prevented, and billions of losses could have been avoided.

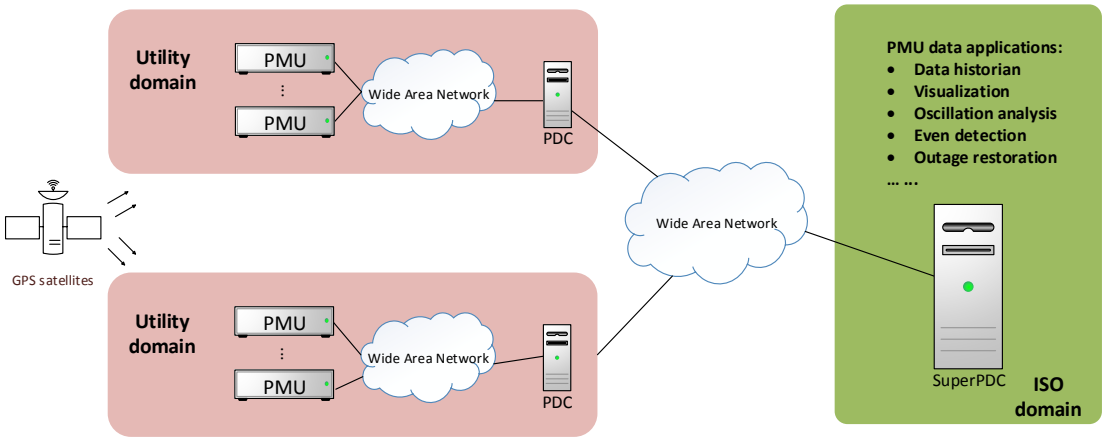


Figure 2.2: Synchrophasor network model

Figure 2.2 depicts the typical architecture of a phasor network. A utility company usually deploys phasor measurement units (PMUs) at intersecting locations of its power system. PMUs send the measured data to the local phasor data concentrator (PDCs) within the same utility. Local PDCs send the concentrated data to a central data repository called SuperPDC, usually hosted by the regional Independent System Operator (ISO). A "publisher-subscriber" model can describe the data communication in a phasor network, where PMUs or downstream PDCs publish data while upstream PDCs subscribe to receive measurement data from the subscribed PMUs Bobba et al. (2010).

Essentially, PMU measurement data, are stamped measurements taken by PMUs periodically (typically at 30 ˜120 Hz) from different locations on a power grid. All measurements use a shared time reference (usually provided by the GPS clock with one microsecond accuracy) to keep synchronized. The measurement data include:

- Voltage (three phases) magnitude and angle
- Current (all three phases) magnitude and angle
- Frequency
- dF/dt (change in frequency over time)

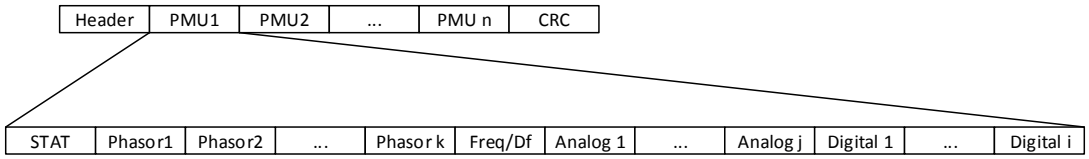


Figure 2.3: The structure of a PMU data block

Some standards have been proposed for the communication between PMUs and PDCs. These include IEEE C37.118, IEEE 1344, BPA PDCstream, IEC 61850-90-5 Grid Protection Alliance (2014). In this research effort, we base our vulnerability assessment on the most commonly used IEEE C37.118 protocol. The same methodology and testing techniques can also be adapted to other standards. As a protocol on the application layer, IEEE C37.118 can work with both serial and network communications using IP (both UDP and TCP are supported). However, recent installations have mainly used IP over Ethernet according to Martin et al. (2014). For this reason, this research also assumes a synchrophasor network using IP over Ethernet as the underlying communication technologies.

2.3.2 IEEE 37.118 Protocol

IEEE C37.118 standard is a standard proposed for the specific use of the data transmission of the synchrophasor data. C37.118 defines four types of frames, namely *data*, *configuration*, *header*, and *command* frames Martin et al. (2014). Figure 2.4 captures a example of establishing data transmission with these frames defined in C37.118 protocol.

Data frames carries real-time measurements data from a PMU or PDC to the measurement data subscribing devices, *i.e.* the recipient of the actual phasor measurement data. An IEEE C37.118 data frame is captured in Figure 2.3. A data frame consists of a header that specifies the message length, the source ID (*i.e.* sender's ID), a time-stamp, and other status information (*e.g.* source and quality

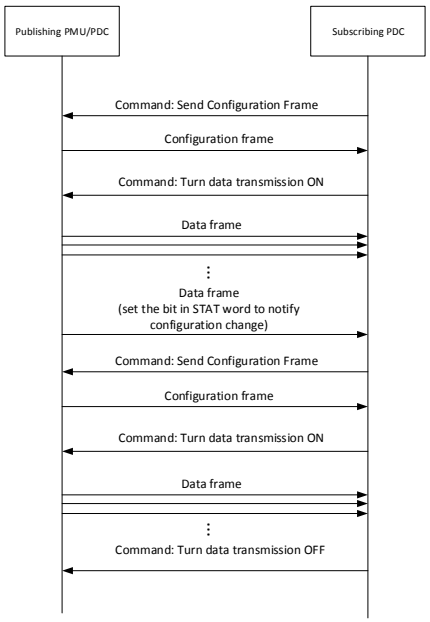


Figure 2.4: A typical data transmission process by C37.118 protocol

of the data). Data blocks follow the header. Each data block contains a PMU's real-time measurements with the timestamp specified in the header. If the data frame is sent by a single PMU, it contains only one data block. The standard allows a single data frame to encapsulate multiple data blocks corresponding to different PMUs reporting to the same PDC. A data block is headed with status words that can be defined by the network owner to indicate data validity, time source quality, *etc.* Following the status words are measurement data, including phasors in either rectangular or polar format, frequency, rate of change of frequency (dF/dt), analog and digital quantities.

Configuration frames are sent by PMU/PDC to notify the subscribing device the configuration of the publishing device, including the publisher's device ID, data format in the data frame, the number of PMU blocks in the frame, the number of phasors, the number of analog values, rate of transmission *etc.* . Configuration frames are very important, as the parsing of coming binary data frames would be impossible without the information conveyed in the configuration frame. Three versions of

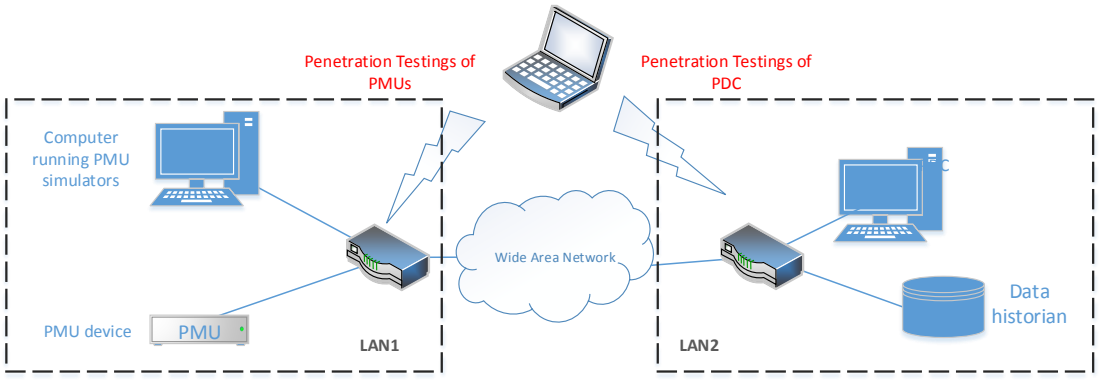


Figure 2.5: A prototype of the synchrophasor network for penetration testing

configuration frames are currently supported in the latest standard, *i.e.* *CONFIG-2*, *CONFIG-2*, and *CONFIG-3* (optional), which corresponds to their respective request for configuration command frames.

Command frames are sent from a data concentrator (subscriber, an upstream PDC) to its publisher (downstream PDC or PMU) to coordinate the communication. The command frames include request for *CONFIG-2*, *CONFIG-2*, and *CONFIG-3*, starting data transmission, and stopping data transmission.

Header Header frames are also sent from PMU/PDC to the receiving device and contain human-readable configuration and descriptions in ASCII format.

More details with respective to the C37.118 standard can be found the published standard available at IEEE Standards Association (2014).

2.3.3 A small Scale Prototype of Synchrophasor Network

To conduct penetration testing of the synchrophasor network, we set up a small scale prototype of synchrophasor network, which is shown in Figure 2.5. In the prototype, PDC and PMU are connected to their respective local area network (LAN1 and LAN2). Additionally, the PDC stores concentrated synchrophasor data on a historian server running MySQL residing on the same LAN. Both LANs connects the wide area network. As such, this prototype emulates a phasor network where the PMUs are

deployed at different substations, and they send their real-time measurement data across the wide area network to a remote PDC deployed at the control center.

We choose LANs of PMU and PDC to be the penetration point in the later vulnerability assessment. The reason is that the traffic in the WAN is typically encrypted and protected through the use of security gateway Stewart et al. (2010), performing a penetration in WAN inefficient if not impossible. Instead, since IEEE C37.118 used by PMU or PDC does not include any encryption or authentication. These communications within an unprotected LAN can be the ideal target of cyber attacks.

We adopt the openPDC by Grid Protection Alliance (2014) as the PDC server in the prototype. In addition to a real PMU device, we emulate the PMUs with computers running PMU simulators provided by iPDC Khandeparkar and Pandit (2014), which is an open-sourced Linux software package written in C++ and adheres to the IEEE C37.118. Even though the PMU simulators are purely software based and do not possess a real GPS module, they still get synchronized using the Network Time Protocol (NTP) protocol. Furthermore, although the simulator does not possess dedicated data acquisition hardware, it can synthesize PMU measurement data or use the historical measurements as the data source in a simulated phasor data transmission. Note that using PMU simulators does not affect the validity of our research findings, because we are only interested in vulnerabilities associated with the communication network and the software implementations, which are independent of the data acquisition process.

2.4 Vulnerability Assessment

Vulnerability assessment is a network defense activity to attack the network of interest ethically, with the purpose of discovering and addressing vulnerabilities before the malicious attackers can do so and make real damages. In this research, we will conduct vulnerability assessment through the penetration testing of the phasor

network prototype described in the Section 2.3.3. We hope that this research effort not only identifies vulnerabilities that undermines the network security of the phasor network, but also illustrates the methodology of the vulnerability assessment of any deployed phasor network.

2.4.1 Procedures of Penetration Testing

A penetration testing generally comprises of three phases: *reconnaissance*, *exploration*, and *exploit development* according to Centre for the Protection of National Infrasctructure and Department of Homeland Security (2010).

The *reconnaissance* phase is the initial phase to collect information about the network under test. It includes host discovering, operating system fingerprinting, traffic monitoring. It can be done using host discovering and port scanning tools like Nmap, Lyon (2009) and packet sniffing tool Wireshark Wireshark Team (2014) or Scapy Biondi (2014). The reconnaissance phase may also include social engineering, intelligence gathering that is not limited to the cyberspace as well. The result of reconnaissance gives the tester ideas about potential vulnerabilities the network may have.

The *exploration* phase aims at validating speculated vulnerabilities in the reconnaissance phase, by automated and manual approaches. The automated exploration of the vulnerabilities of the phasor network is the same to the standard hacking and leveraging scanning and enumeration. With the information gained in the reconnaissance phase, it is possible to get a list of known vulnerabilities associated with the technologies that are found to be used in the system under test. Software packages like Metasploit Maynor (2011) can automate the process. For instance, if a host is fingerprinted to be running a Windows operating system, the automated tools can perform vulnerability scan and check the host against vulnerabilities that are specific to that operating system and are previously reported to vulnerability

database like National Vulnerability Database NIST (2014), Common Vulnerabilities and Exposures CVE official site (2014).

For unknown vulnerabilities, the manual exploration is needed. Since the phasor network is less deployed and has much smaller user base compared to the Internet, few vulnerabilities are reported. As a result, manual exploration are expected to take a significant part of the exploration phase.

The final phase is *exploit development*. This phase aims at proving that the found vulnerabilities can indeed be exploited to launch practical attacks that impact the function of the network in real practice.

2.4.2 Manual Penetration Testing Techniques

In this subsection, we briefly introduce the testing techniques that will be used in our assessment.

2.4.2.1 Packet Sniffing

Packet sniffing allows a tester, or an intruder in real attacks, to eavesdrop the data communication on the network under testing. If packets are transmitted in clear text, with packet sniffing, an attacker can obtain the same information that is intended for the legitimate users. Even if packets are encrypted, packet sniffing can still get the side-channel information such as the packet arrival rate, length distribution, inter-packet delay. The side-channel information can be used to deduce, for example, whether a stream of packets belongs to a data stream of the phasor measurement through side-channel analysis.

Packet sniffing in a shared media network, *e.g.* Ethernet using a network hub or wireless local area network (IEEE 802.11) can be easily carried out by receiving all packets in promiscuous mode. In a switched media network *e.g.* a typical Ethernet network using switches to connect the Ethernet hosts, techniques like ARP poisoning

or MAC flooding are needed to reroute the traffic to the sniffer. More details about packet sniffing in a switched Ethernet can be found in Wireshark Team (2014).

In our testing, both Wireshark wireshark.org (2014) are used for packet sniffing.

2.4.2.2 Packet Injection

Packet injection refers to an unknown third party (a tester or an intruder) sending illegal packets to the network services or applications. Injected packets are constructed in such a way that they appear to be legitimate but will actually interfere normal execution of the network services or applications.

Packet injection is already known for causing well-known attacks targeting at network and transport layers, such as Smurf, IP-spoofing, TCP session hijacking. In a similar way, we use packet injection to test the vulnerabilities specific to the C37.118 standard in this research effort.

The Scapy program Biondi (2014) is used in this research for packets injection.

2.4.2.3 Fuzzing

Initially induced by Miller et al. (1990), fuzzing is an effective way to find unknown bugs and vulnerabilities in software programs and protocols. This technique enumerates all possible forms of input and supplies the input to software under test. If the software has not been implemented with security consideration *e.g.* the software inadequately validating the input or not handling exceptions properly, some inputs may cause exceptions from the normal execution. In the same way, being aware of these problematic inputs, the attacker can also leverage them to disrupt normal communications of the phasor network.

For fuzzing test against a phasor network, the payloads of the various types of frames are first enumerated according to the specification of the C37.118 standard. These payloads are then assembled with the appropriate Ethernet, IP, and UDP headers to form the respective fuzzing frames. The packets are then sent to the

corresponding device: HEADER, CONFIG and DATA frames are sent to the PDC, COMMAND frames are sent to PMU.

Among a number of tools for fuzzing tests, we use Scapy for this task. Scapy allows us to develop our own Python script to customize the fuzzing test specific to C37.118 defined network protocol and service. In contrast, other tools such as BED, SFUZZ, SICKFUZZ only support common protocols such as FTP, HTTP, IMAP.

2.5 Results of Vulnerability Assessment

2.5.1 Reconnaissance

In the reconnaissance stage, we eavesdrop the traffic at both penetration points by packet sniffing with Wireshark. The presence of PMU and PDC were successfully confirmed by the capture of C37.118 frames. In particular, the captured C37.118 CONFIG frames also leaked the PMUs' configuration information.

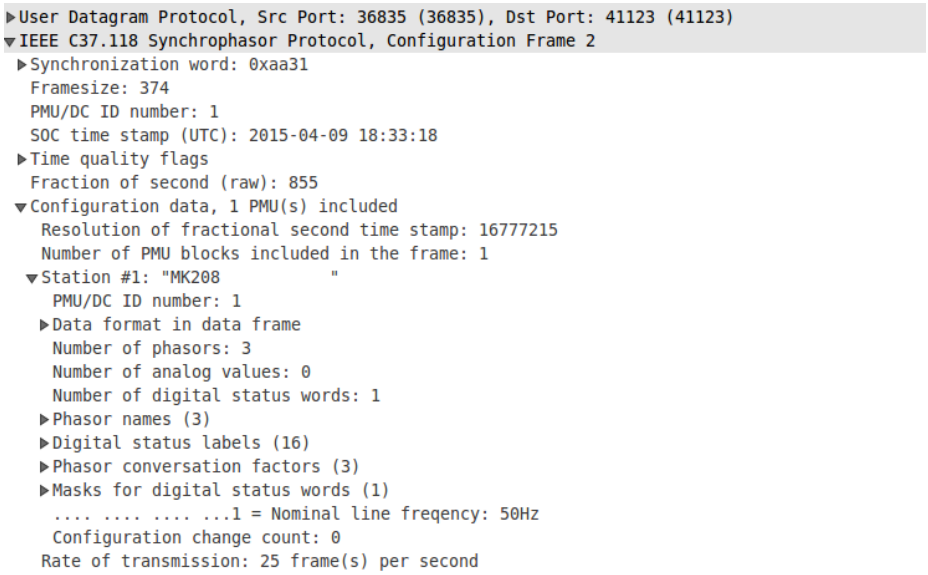


Figure 2.6: Wireshark captured IEEE C37.118 CONFIG-2 frame revealing all configuration information about a PMU

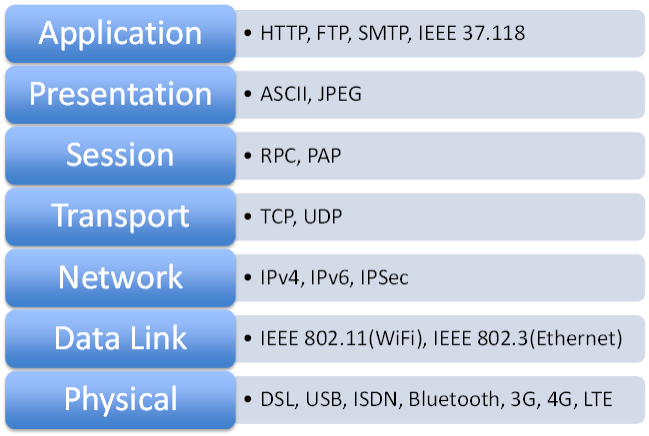


Figure 2.7: Seven layers of OSI model and example protocols for each layer

Figure 2.6 shows the captured configuration frame. As seen, the captured configuration frame divulges the attacker the information about the PMU ID Number, IP and MAC address, TCP/UDP port number, substation name, data frame format (number of phasors, number of analog signals, the length of status word, rate of transmission, etc.). The leaked information allows the attacker to understand better the overall network environment and, thus, to facilitate to explore vulnerabilities in the *exploration* phase.

2.5.2 Vulnerability Exploration

As previously mentioned, the goal of *exploration* phase is to validate the possible weaknesses of the phasor network, which are deduced from the information leaked from the reconnaissance phase.

Vulnerabilities can be categorized according to which layer in OSI (Open Systems Interconnection) model a particular vulnerability relates to. Figure 2.7 captures the OSI model and the exemplary protocols for each layer.

C37.118 is often deployed on top of TCP/UDP, IP over Ethernet. Therefore, any network vulnerabilities of the TCP/UDP, IP, or Ethernet also affects the phasor network. However, since TCP/UDP, IP or Ethernet are widely used technologies and

Table 2.1: Vulnerabilities to be validated in the *Exploration* phase

Cause of Vulnerabilities	**Possible attacks**	**Testing technique**
Lack of encryption	eavesdropping, replay	Packet sniffing
Lack of user authentication	impersonation man-in-the-middle attack	Packet sniffing Packet injection
Lack of message authentication	frame modification	Packet sniffing Packet injection
Unexpected frames	Denial-of-Service	Packet injection Fuzzing
Lack of input validation	SQL injection code injection	Packet injection

their security issues have well been studied and followed by the research community and industry such as Kaufman et al. (2002); McClure et al. (2009), we put our focus on the security issues specific to IEEE C37.118 protocol and software implementations of PMU and PDC in this research effort.

Evaluating the information gathered in the reconnaissance phase and also taking into the public information about the C37.118 standards, we list vulnerabilities that we will manually explore and validate during the exploration phase in Table 2.1.

In general,these vulnerabilities are chosen for two reasons. First, they are easy to exploit as none of these vulnerabilities requires physical access or proximity to the PDC or PMU device. As long as the attacker is on the same LAN, it is possible for the attacker to launch relevant network attacks. Second, the listed vulnerabilities directly impact the phasor measurement data with respective to data's confidentiality, availability and integrity. The compromise of any one of the three principles will render the phasor network, whose ultimate goal is to provide reliable and accurate measurement data, futile.

We briefly explain the listed vulnerabilities as follows.

First, as we have already discovered in the *reconnaissance* phase, all C37.118 frames are transferred in clear. Hence, not only can attackers intercept and eavesdrop configuration frames but it is also possible for attackers to eavesdrop for command

and data frames. We use packet sniffing to confirm whether a phasor network is vulnerable to eavesdropping. Results are reported in Section 2.5.2.2.

Second, as the C37.118 standard does not specify any user authentication mechanism, it is possible for the attackers to impersonate a legitimate publishing or subscribing devices and confuse, mislead, or sabotage other parties in the phasor network. Relevant results are covered in Section 2.5.2.3.

Third, as the case of lacking user authentication, neither C37.118 includes any message authentication mechanism. As a consequence, all frames are subject to frame modifications; a receiving device is unable to distinguish legitimated frames and modified frames. We use packet sniffing and packet injection to validate this vulnerability. Results regarding this vulnerability is also discussed in Section 2.5.2.3.

Fourth, as a stateful protocol, a device that runs C37.118 protocol manages its transition of states based on its current state and the frames it receives. If the coming frames are expected under the current state, the device should make the state transitions accordingly. If not, the device should also handle for the case properly. In real practice, implementations of PMU or PDC may ignore the rare but not impossible situation so that it become unresponsive or even crashes upon receiving an unexpected frame. We use packet injection and fuzzing to test if the PMU and PDC implementation in our phasor network prototype have such a vulnerability. Section 2.5.2.4 presents relevant results.

Lastly, PMUs or PDCs may not properly validate input, which leads to code injection attacks. Injection ranked the top security risk according to The Open Web Application Security Project (2014). Like web applications, entities on a phasor network also need to process input data that cross trust boundaries. This opens the opportunity for attackers to perform code injection attacks, especially SQL injection attacks since relational database is often used in the implementation of the PDC and also in the historian server of the phasor network. Lack of input validation contributes to a great deal of cyber attacks targeting at web applications For example, a PDC processes the measurement data reported by remote PMUs; a historian server

stores reported measurement data into its relational database. Therefore, input data should not be assumed trustworthy. In fact, knowing the flaws of the software implementations, attackers can supply crafted input data to the PDC and PMU in order to trigger the inherent flaws and cause malfunction from inside. We use fuzzing and packet injection to test if the PDC or PMU appropriately validates external input. Section 2.5.2.5 deals with the vulnerability of SQL injection.

2.5.2.1 Automated Testing

Metasploit, the automated penetration testing tool, were used to conduct automated testing. Our automated testing using Metasploit did not find any known vulnerabilities of our small-scale prototype, because the prototype was made certain that all hosts and routers on the network are updated and fully patched and that all unused port are disabled. Real network conditions vary with the actual implementation and configuration of the specific phasor network. To verify no known vulnerabilities exists in the network, it is very critical that the phasor network under test be scanned by the automated penetration testing tools.

2.5.2.2 Eavesdropping

We found that just like Wireshark can successfully capture the configuration frames and obtain the information conveyed by the frame, it can also eavesdrop on any unencrypted command and data frames.

Figure 2.8, 2.9 and 2.10 show the three types of command frames captured with Wireshark.

Figure 2.11 shows the captured data frame.

2.5.2.3 Lack of User and Message Authentication

As of this writing, the C37.118 standard does not specify any authentication mechanism to authenticate a user in the phasor network or a message's origin. In our

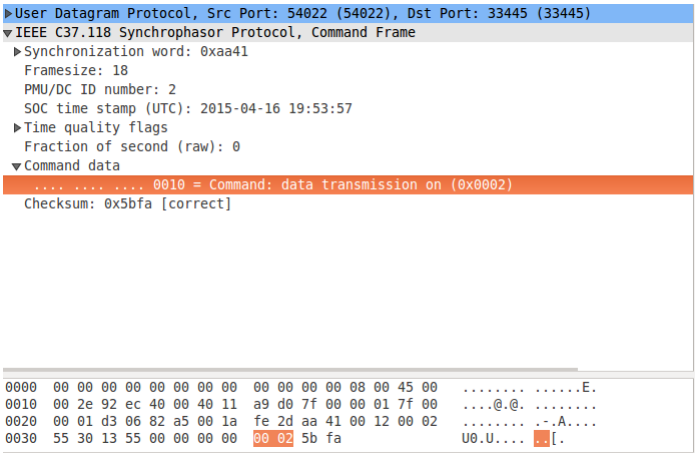


Figure 2.8: Captured command frame: Data transmission on

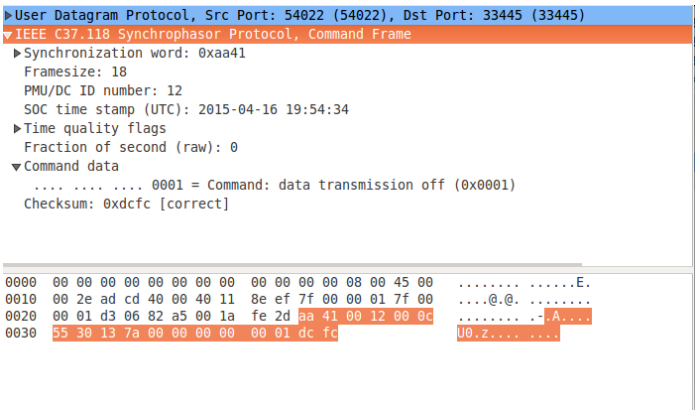


Figure 2.9: Captured command frame: Data transmission off

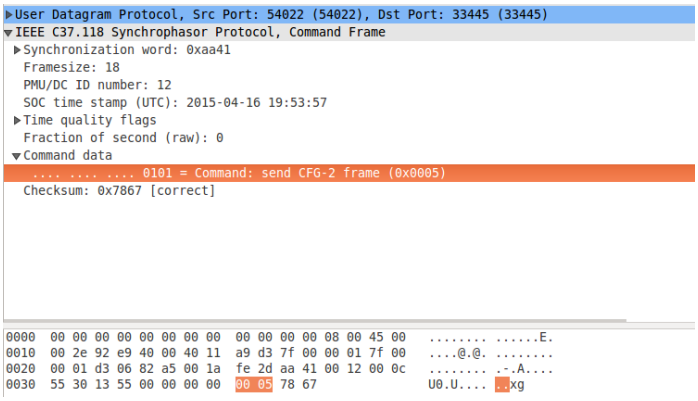


Figure 2.10: Captured command frame: request CONFIG-2

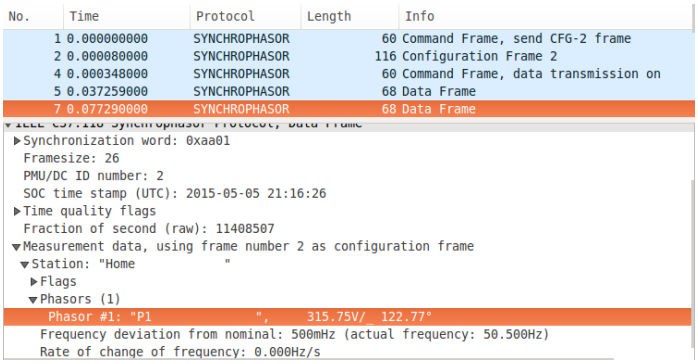


Figure 2.11: Captured Data frame

testing, we have found that anyone who are on the same local area network (LAN) of the PMU or PDC can forge and inject all four types of IEEE C37.118 frames to the network spoof PMU and PDC in the network.

The frame spoofing consists of the following steps.

1. Capture an authentic frame that is of the same type as the frame to forge and inject. The authentic frame provides necessary information for spoofing, including the source and destination MAC address, IP address, and TCP/UDP port.

2. Duplicate the captured frame but change bytes that indicate the actual commands, measurements, or configurations to the spoofing value.

3. Change the time stamp of the frame.

4. Recalculate the checksum of the command frame so the forged frames may pass the checksum checker at the receiving end.

5. Inject the forged frames into the network.

The rest of this subsection presents our testing result on spoofing attacks with respective to command, configuration, and data frames, respectively.

Command Frame spoofing

Command frames are sent by the subscribing device to publishing device to establish and control the data transmission.

A command in a command frame is encoded with the third and fourth bytes of the command frame payload. The commands and their corresponding codings are illustrated in Table 2.2.

Table 2.2: Command bytes of IEEE C37.118 command frame

	Command bytes (in Hexadecimal)
Data transmission off	0x01
Data transmission on	0x02
Request for CONFIG-2	0x05

As previously mentioned, without proper message authentication, it is trivial for an attacker who has access to the local network to inject forged frame to spoof recipients of the command frame *i.e.* the PMU.

In our test, during an ongoing data transmission, we first captured a legitimate "Turn data transmission on" command frame, then we modified the captured frame to forge a "Turn data transmission off" command frame and inject it at the LAN of the PMU.

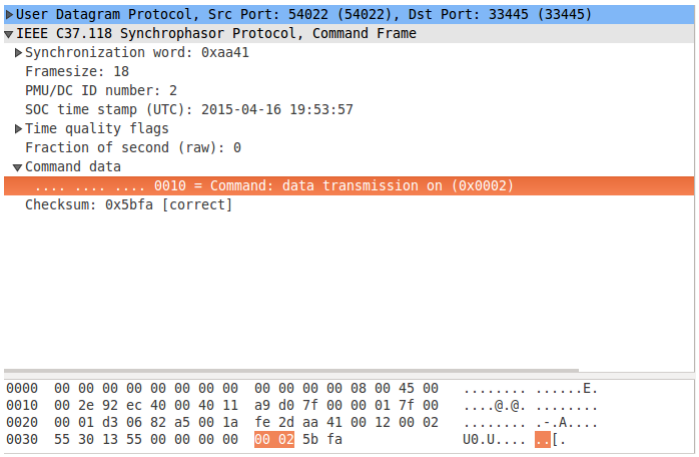


Figure 2.12: Legitimate Command Frame to turn on the data transmission

Figure 2.12 shows the legitimate command frame captured with Wireshark. As seen, the captured frame was a "Turn Data Transmission On" command frame.

With the captured command frame, we use a Scapy script to forge a new command frame with a "Turn Data Transmission Off" command. The forging was accomplished by 1) replacing the time stamp in the captured command frame with the current time stamp, 2)replacing the command bytes in the command frame to 0x01, *i.e.* "Turn data transmission off" frame, and 3) recalculate the checksum. As the checksum is the only validation mechanism, the receiving PMU could not distinguish the forged command from legitimate ones. Therefore, the PMU will accept the forged frame and execute the command contained in the forged frame.

Figure 2.13 illustrates the Wireshark capture of the forged "Turn Data Transmission Off' command frame in flight.

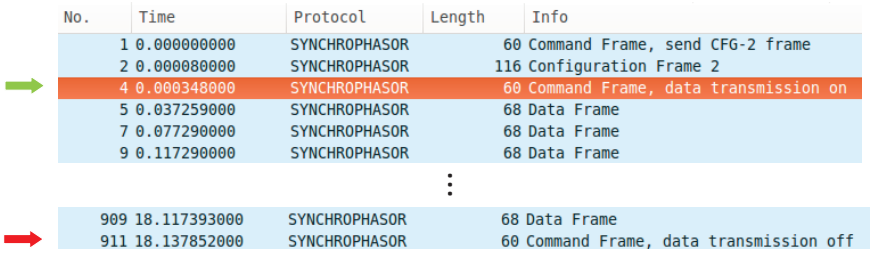


Figure 2.13: Wireshark capture of the forged command frame
Green arrow points to the legitimate command frame. The red arrow points to the forged one.

As a consequence of the spoofing, the PMU stopped sending measurement data upon receiving the forged frame. Figure 2.14 captures PMU simulator's console after it received the falsified command frame. As seen, the PMU data stream was turned off by the forged command frame.

This attack successfully resulted in a data outage for the subscribing PDC.

Configuration Frame spoofing

Configuration frames in IEEE C37.118 are sent from the subscriber upon the receipt of command frames from the subscribers.

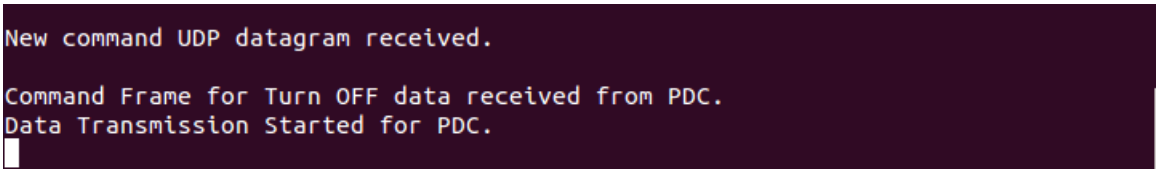


Figure 2.14: The PMU's console after the command frame spoofing attack

Wrong configuration frame misleads the subscribing PDC and prevents the PDC from correctly processing the binary data in the payloads of data frames sent subsequently from the publishing PMU due to the mismatch between the configuration frame and data frames. For instance, if the forged frame that are later accepted by the PDC changes the phasor format (from Rectangular to Polar and vice versa), all measurement later received will be parsed based on the false data format.

In our test, in the same way as performing the command frame spoofing attack, we first capture a legitimate configuration frame, based on which we forge a falsified configuration to change the number of the phasors. We then inject the forged configuration frame when the subscribing PDC requests the publishing PMU to send the configuration frame.

Our test showed that an attacker could successfully spoof the subscribing PDC by sending the wrong configuration frame.

Figure 2.15 demonstrate that after spoofing the PDC with the wrong configuration frame, subsequent data frames sent by the publishing PMU do not conform to the spoofing configuration frame. The PDC cannot parse and interpret the data frames with the received configuration frame.

Data Frame spoofing

An attacker can also inject data frames containing falsified measurement data to the phasor network. By injecting the falsified data frame, an attacker can make the PDC admits wrong data to the collection of phasor measurement data.

In our assessment, we first capture a legal data frame and use it as a template to forge falsified data frames. The falsified data frames have the same format as the legitimate one, but the only one phasor in the data frame is set to all zero value.

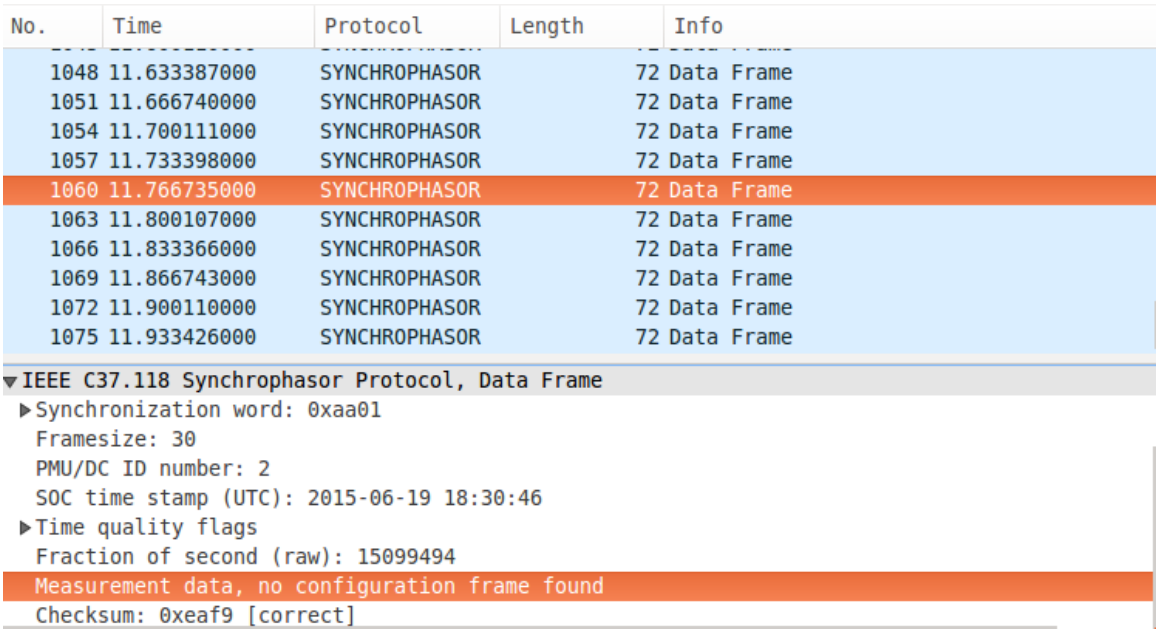


Figure 2.15: Data frames cannot be associated with the right configuration frame under a configuration spoofing attack

The falsified data frames are injected with an interval of 0.01 second. Figure 2.16 demonstrates the bogus data were reflected in the PDC's output CSV file, which are the data source to support higher level PMU data applications. As seen from the figure, the shaded lines indicate that the multiple records of $0.000000, 0.000000$ as the phasor measurements named $P1$ are inserted into the CSV file.

2.5.2.4 Fuzzing Test

Fuzzing tests were carried out to discover unknown vulnerabilities that exist in the network services and applications. Our fuzzing test covered four types of C37.118 frames. During the fuzzing test, fuzzing frames were sent to the PDC and PMU servers, and we monitored any abnormal behavior of the PDC and PMU.

To improve the efficiency of the fuzzing test, instead of blindly generating all possible frames, we tested the PDC or PMU with only frames that adheres to specification of the C37.118 standard and excluded frames that will apparently be rejected by the PMU and PDC. For instance, a checksum is calculated over the

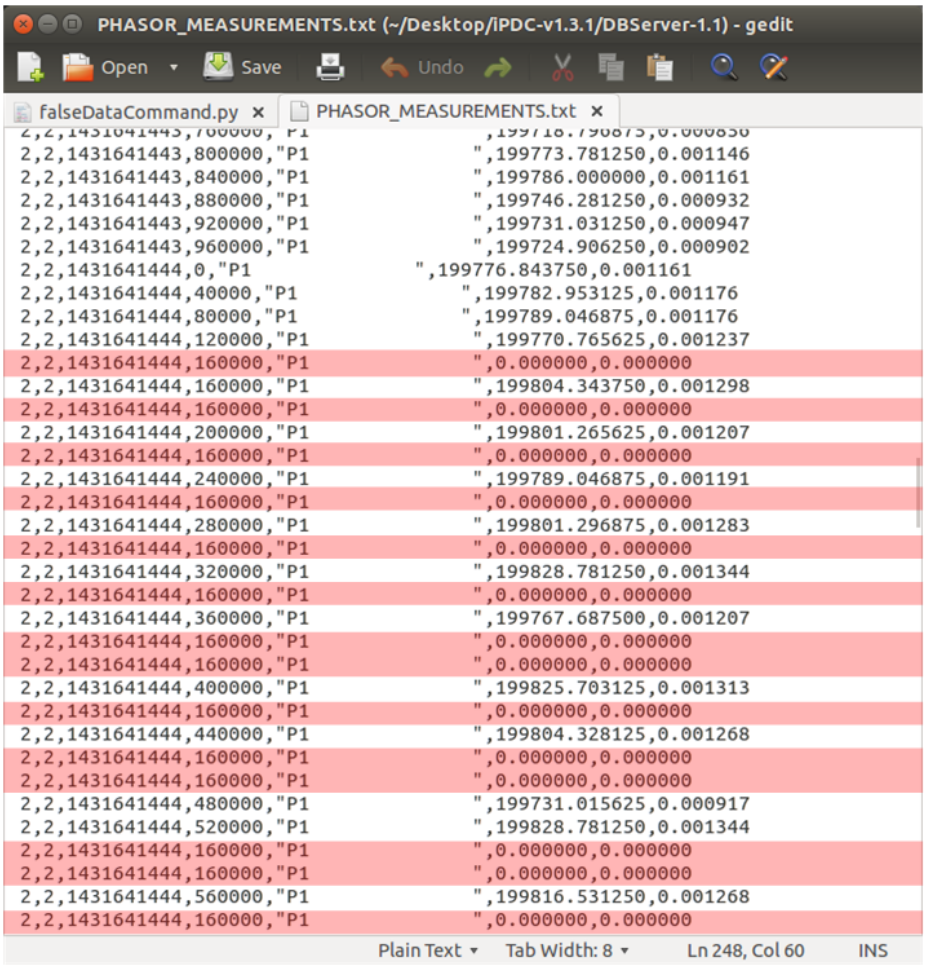


Figure 2.16: Received phasor measurements during the data frame spoofing attack. Red shaded frames are injected by the attacker

payload of any C37.118 frame and is appended to end of the frame, to detect bit errors that occur during the transmission of the frame. If the appended checksum does not agree with the checksum calculated on site with the frame at the receiving end, the frame will be rejected by the recipient. In that case, the input data carried by the frame will be discarded and not processed at all, rendering this random input data useless in this test. Therefore, we may improve the efficiency by avoiding testing fuzzing frames with an invalid checksum. To do so, we only randomize fields that carry the actual input data and append a valid checksum accordingly.

In general, to generate fuzzing frames, we first identify the bytes that are fixed, which include SYNCHRONIZE word that specifies the type of payload (`0xaa0_`, `0xaa3_`,`0xaa4_` correspond to Data, Configuration, and Command frames, respectively) at the very beginning of the every C37.118 frame; the bytes that should be fuzzed, which are the representation of input data that cross a trust boundary, *e.g.* the number of the phasors in a configuration frame, a measurement of voltage in a data frame, or a command (turn data transmission on/off) in a configuration) frame; lastly, the bytes that depend on other bytes: these are usually the checksum at the end of the frame.

As an example, in order to fuzz a command frame that is 16 bytes long, we first set the first two bytes to the respective SYNCHRONIZE word as `0xaa4_`, where `0x_` denotes an arbitrary hexadecimal digit. Since the last two bytes are reserved for the checksum, we fuzz only the 3rd through the 13th bytes, inclusive. There are 2^{14*8} possible values for these bytes. For each of the possible values, the checksum of the SYNCHRONIZE word and that value is calculated and appended to form the fuzzing frame. In our prototype, the C37.118 runs on top of UDP, and the recipient of the command frame is the publishing PMU. Therefore, the fuzzing frames should be sent to the PMU by UDP datagrams, which is accomplished via packet injection.

During the fuzzing test, the PDC or PMU may exhibit abnormal behavior due to reasons other than the fuzzing frames. For example, Morris et al. (2011) found that the PMU may become unresponsive due to excessively high packet rate of coming data frames. To determine whether a certain randomized frame may cause the failure of the PMU or PDC, we need to exclude other potential causes like network congestion. Therefore, we set the packet injection rate to 100 frames per second, which is a rate that a typical PDC or PMU can handle without any loss of frames Morris et al. (2011).

The fuzzing test are carried out with the scapy script `fuzzing_test.py`. The following are results of the fuzzing test against PMU simulator. PMU from different vendors should have different test results depending on the quality of the product

with respective to software implementation and standard compliance. However, the same testing methodology and testing scripts are still applicable.

Command frame fuzzing test result

Command frames are sent from PDC to PMU for the control of the data transmission. To test whether a PMU is affected by malformed command frames, we run the fuzzing test scapy script to send fuzzing command frames to the PMU during the normal communication. During the test, the PMU simulator became unresponsive after receiving a specific kind of malformed command frame. A closer examination reveals that the all fuzzing command frames causing the problem contain the command of "Request CONFIG-2 frame" from the subscribing PDC, with the frame's command bytes set to `0x05` and the frame's source device ID number set to the subscribing PDC's ID number. When receiving that frame, the PMU server sends its CONFIG-2 frame to the PDC without examining other fields of the frame nor verifying whether the frame is malformed. Upon receiving the CONFIG-2 frame from the PMU, the PDC follows the protocol and sends out a command frame to inform the PMU to start the data transmission. However, the PMU has already been sending phasor measurements, and it does not expect such a command at the current state. Since the software implementation lacks proper handling of duplicate "Turn data transmission on" command frames from the same subscriber, the PMU software crashes accordingly.

The problem described above exposed a major design and implementation flaw with the PMU server program under test: its design and implementation had not considered unexpected command frames; sending duplicate "Turn data transmission on" command frames can easily crash the PMU server.

In general, the behavior of a PMU depends on its current state and the command frame it receives. A certain state should properly handle all possible external inputs, including the illogical input. In this case, during an on-going data transmission, a "Turn data transmission off" command frame is the only logical frame a PMU is supposed to receive from a legitimate subscribing PDC. However, the PMU should

defensively take into consideration all possible command frames including illegal ones. Otherwise, the behavior of a PMU is undefined and the stability is open to question.

Configuration frame fuzzing test result

Configuration frames are sent from publisher (PMU) to subscriber (PDC) upon the publisher's receipt of "Send CONFIG-2 frame" from the PDC.

Likewise, we used `fuzzing_test.py` to perform the fuzzing test.

Our result showed that the openPDC software were not influenced by the fuzzing CONFIG-2 frames and did not crash or have any behavior that would otherwise indicate a vulnerability throughout the test.

Data frame fuzzing test result

Data frames are sent from the publisher (PMU) to the subscriber (PDC) to transmit the actual measurement data.

In the data frame fuzzing test, data frames are randomly generated with a scapy script and sent to the subscriber PDC during the normal data transmission. Our result shows that the PDC did not crash or showed any abnormal behavior throughout the test. The PDC accepted all randomized data frames and continued to function properly on the data frames.

2.5.2.5 SQL Injection attack

SQL injection (SQLI) takes place when a program fails to validate inputs that cross the trust boundary. As a result, malicious SQL statements are inserted for execution.

In a phasor network, SQL injection may happen when a PDC extracts configuration information from a configuration frame and saves the information into the associated database using SQL statements. To be more specific, for the purpose of dynamically manage and update the phasor network configuration, a PDC stores the configuration information of the PMUs it is interacting with in a connection table in a database. When a PDC receives a configuration frame from a PMU, the PDC first looks up the connection table for the PMU with the `PMU/DC ID Number` specified in the configuration frame, as shown in Figure 2.6. If the PMU

already exists in the connection table, the configuration specified in the configuration frame is then used to update the existing configuration. Otherwise, a new PMU is added and registered with the specified configuration. The PDC performs the table looking up by the querying the database with the statement `SELECT * FROM MAIN_CONFIG_TABLE WHERE DEVICE_ID = PMU_ID_Number`, where the `PMU_ID_Number` is the PMU ID number indicated in the configuration frame.

If the PDC does not validate the input of `PMU/DC ID Number` supplied by the configuration frame and `PMU/DC ID Number` is not strongly typed as a numeric value, it is possible for the attacker to inject executable malicious SQL queries by simply appending another query to the string of `PMU/DC ID Number`, which makes the query do more than what is originally intended. As an example, using the packet injection, an attacker can inject a configuration frame in which the `PMU/DC ID Number` is specified as "2;DROP TABLE MAIN_CONFIG_TABLE". If the input is not checked against SQL injection, the statement to be executed by the database becomes "`SELECT * FROM MAIN_CONFIG_TABLE WHERE DEVICE_ID = 2; DROP TABLE MAIN_CONFIG_TABLE`", which is a valid batched query supported by many database systems. However, the batched query will first look up the `MAIN_CONFIG_TABLE` where the `DEVICE_ID` equals `2` and then delete the whole `MAIN_CONFIG_TABLE`. Without the table, the PDC cannot function appropriately.

The SQL injection in the above example is a typical piggy-backed query based SQL injection attack, according to the classification by Halfond et al. (2006). Our test script also tested for other types of SQL injection attacks, including tautologies-based attack, illegal/Logically incorrect queries, Union queries.

During our test, we found that both openPDC and iPDC effectively prevent SQL injection attacks, because both software sanitize the input extracted from the frames and use parameterized queries with strongly typed parameters, which eliminates the chance of SQL injection.

2.5.3 Exploit Development: A Practical Exploit of IEEE C37.118: Data Stream Hijacking

When combining the above vulnerabilities found through the *exploration* phase, attackers can perform more sophisticated attacks like data stream hijacking via command and data frame spoofing together to maximize the impact of the attack. In a data stream hijacking attack, attackers aim at taking over ongoing phasor data transmission and sending wrong measurement data to the upstream PDC to mislead the user of the data. Data stream hijacking attack is difficult to detect without additional information as the receiving PDC cannot distinguish bogus data from authentic ones.

In this section, we demonstrate a practical data stream hijacking attack with the small-scale prototype of phasor network. Specifically, we show that how the attack impacts the wide-area monitoring of the WECC 179-bus system model.

Consider a scenario where a phasor network is employed to obtain the early warning and to maintain the situational awareness of the potential inter-area oscillations. As proposed by Sun et al. (2012), this can be achieved by two procedures. First, *off-line clustering analysis* is conducted to determine the zones (areas) of the power system. Through off-line clustering analysis, the power system is first divided into four zones, such that an interface between two adjacent zones may develop inter-area oscillations during or after a contingency. Figure 2.17 demonstrates the reduced WECC 179-bus model and the zones as a result of clustering analysis.

Second, PMUs are deployed to monitor the rotor angle of each zone. PMUs at three generators within each zone to measure their respective rotor angles. The PMUs monitoring the power systems are circled in 2.17. All PMUs send their real-time measurement data to a central PDC. Based on the concentrated measurements at the PDC, the real-time angle of an individual zone is obtained by averaging the measurements of angles reported by the three PMUs in that zone. Moreover, the zone angles are input to the method to calculate risk scores presented in Sun et al.

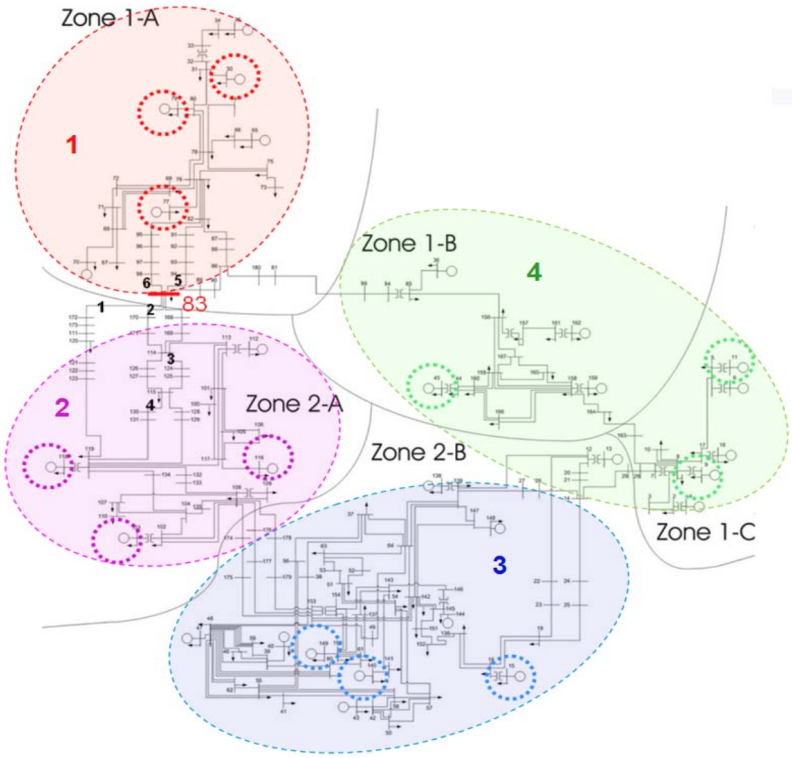


Figure 2.17: 179 bus power system

(2012) and establish real-time situational awareness via data visualization of the angle difference.

Since we are only interested in the communication infrastructure of the power system, we use the small scale prototype to simulate the phasor network described above in this demonstration. We run 12 instances of PMU simulator on two computers (6 on each) on LAN1 to emulate the twelve PMUs monitoring the power system. The central PDC is emulated by a PDC at the other LAN2, which corresponds to the network system of the control center. Additionally, it is assumed that an attacker penetrates the LAN of PDC using a laptop. The network diagram is shown in Figure 2.18.

As introduced earlier, the PMU simulator allows users to specify the data source to be sent to the subscribing PDC. To simulate the behavior of whole phasor network during the power system contingencies, we set the data sources to be the rotor

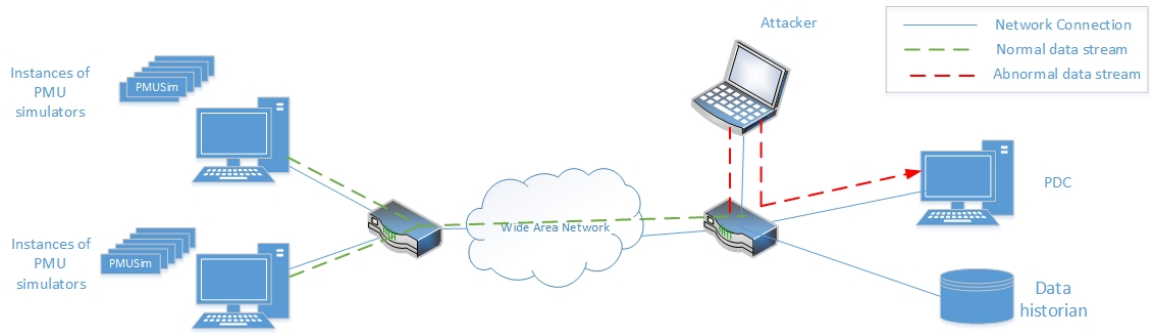


Figure 2.18: Demonstration of a data stream hijacking attack on a WECC 179-bus system with the small scaled phasor network prototype

angles obtained from a dynamic simulation of a sequence of contingencies using TSAT software of PowerTech Labs Power Tech Lab (2015). The applied faults include six three-phase faults at 0s, 40s, 80s, 120s, 160s and 200s near bus 83 (Malin substation) on lines 83# -172, 83# -170, 114# -124, 115# -130, 83# -94 and 83# -98 ("#" indicates the fault bus) to simulate a cascading failure. In other word, by replaying the rotor angles obtained from the dynamic simulation on the respective PMU simulators, we can completely reproduce the behavior of the phasor network during the cascading failure.

Figure 2.19 captures the angles differences on potential out of step (POOS) interfaces of 1-234, 14-23. As seen in the figure, the data received by the PDC provides the correct situational awareness: the angle difference between POOS's keeps climbing as the cascading failure takes place.

In this demonstration, we show that a data stream hijacking attack targeting a phasor network can change the result of situational awareness such that the contingencies vanish from the situational awareness and, thus, blinds the operator.

A data stream hijacking attack consists of following procedures.

1. Attacker eavesdrops the communication between PDC and PMU at the LAN of the PDC. During the eavesdrop, information about the command and data frame format, PMU sampling rate, the IP and MAC address are collected and analyzed for frame forging later.

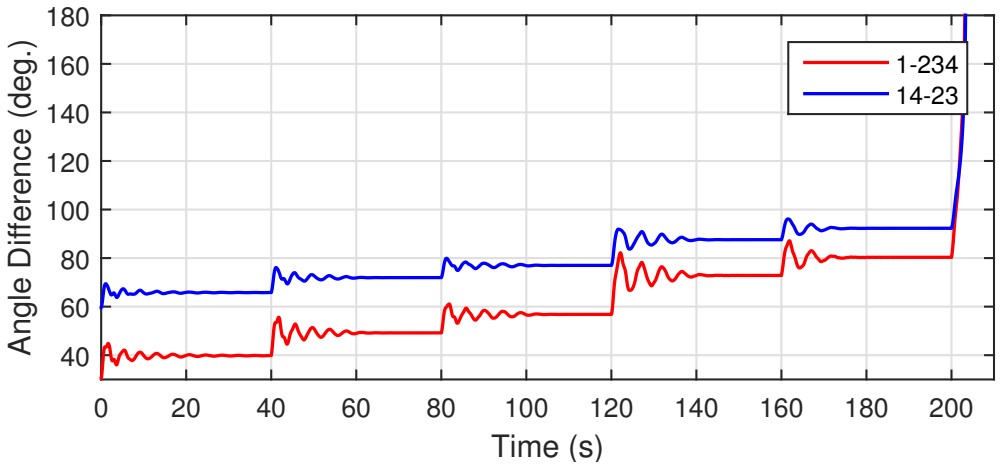


Figure 2.19: Normal angle difference over simulation time

2. Wait for the best timing such that the data stream hijacking attack can do the most damage to the power system (*e.g.* confuse the operator to the most extent) and inject a false command for the PMU to turn off the data frame.

3. Send preselected measurement data by fabricating and injecting forged data frames containing the falsified data, which may come from previously captured measurement data.

In our demonstration, the data stream hijacking attack was launched at the 35th second, which is just prior to the application of the first fault. The attacker hijacks the data stream transmission and keeps replaying to the PDC the rotor angle measurements of all the 12 PMUs between 30th to 35th second it obtained through eavesdropping.

Figure 2.20 illustrates the timeline for the exploit in an unprotected phasor network.

Figure 2.21 showed the Wireshark packet captured during a replay attack.

As shown in Figure 2.22, during the attack, the resulting visualization of the angles difference between interface 1-234 and 14-23 were manipulated such that the inter-area angle differences keep unchanged. Based on this manipulated situational

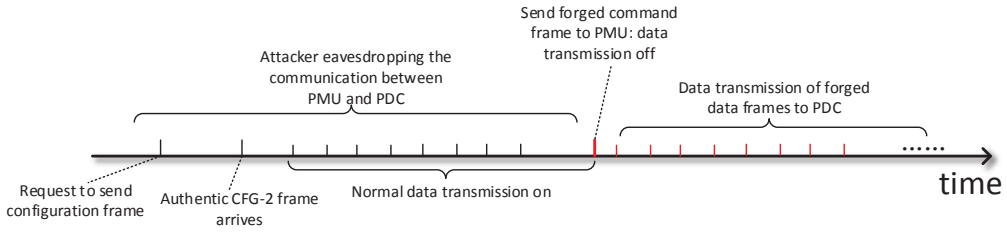


Figure 2.20: Time line for replaying attack in an IEEE C37.118 based phasor network

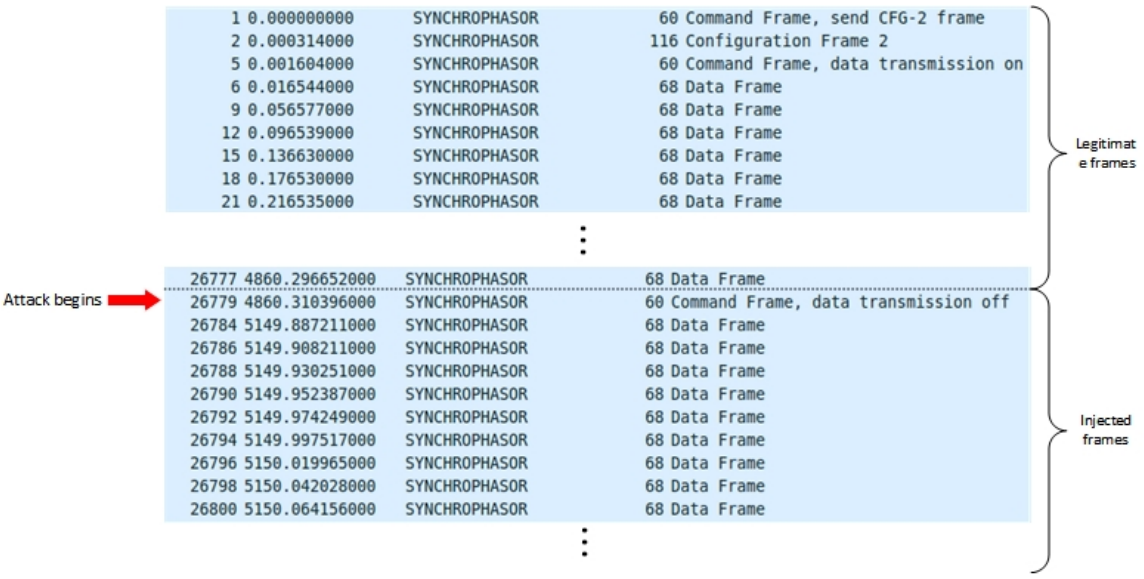


Figure 2.21: Wireshark capture showing the replay attack

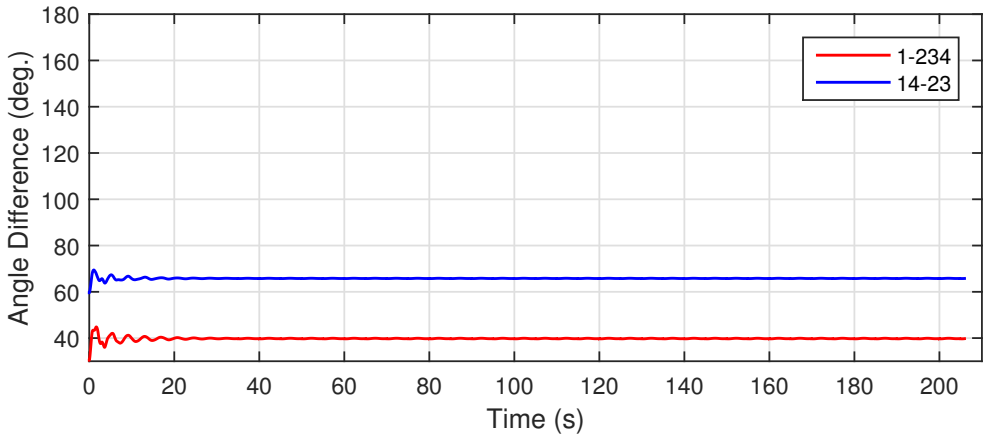


Figure 2.22: Manipulated angle difference by data stream hijacking over simulation time

awareness, the operator or automatic control is blinded and fails to apply preventative controls to mitigate the inter-area oscillation.

2.6 Countermeasures and Mitigations

In this section, we talk about solutions or best practices to dealing with the above vulnerabilities to prevent security breaches in reality.

All the manual penetration testing technique used in our assessment can also be used as an attack technique by adversaries in real cyber penetrations. It is, therefore, important to prevent them from happening in the first place.

Packet sniffing and packet injection can be greatly mitigated by employing SSL/TLS (Secure Socket Layer/Transport Layer Security) or IPSec (Internet Protocol Security) to encrypt the packets between electronic security perimeters (ESP) of PDC and PMU Stewart et al. (2010).

With SSL/TLS or IPSec, the information, including the header, configuration, command, and data frames which are transmitted between electronic security perimeters (ESP) in the phasor network, will be encrypted at the transport layer or the Internet layer, correspondingly. As such, attackers eavesdrop between ESP

can only get the encryption of that information. As a result, attackers gains no advantage through the eavesdropping. They even no longer can fingerprint the PDC and PMU as the SYNCHRONIZE word at the beginning of the C37.118 frames are also encrypted.

To counteract packet injections, we must authenticate packets' origin and ensure the integrity of packets on the fly. Both can be accomplished by SSL/TLS or IPSec, which utilize the public key infrastructure (X.509) certificates to authenticate a legitimate host on the phasor network, and use message authentication code (in SSL/TLS) or integrity check value (in IPSec) to make sure that the data are intact during their transmission. Also, for the same purpose, IEC 61850-90-5 also introduces keyed-hash message authentication code (HMAC) for message integrity verification and authentication of message's origin.

Notwithstanding, the protection provided by SSL/TLS or IPSec covers only the network traffic between ESPs. Network traffic within an ESP (*e.g.* the LAN of the PMU or the LAN of PDC) is not encrypted. Hence, packet sniffing and injection within an ESP is still possible. To overcome this, PMU and PDC should support SSL/TLS or IPSec, which can support truly end-to-end encryption between publishers and subscribers.

Thorough fuzzing test against the services and applications of phasor networks are very helpful to find the design and implementation flaws and errors. It is of great importance to eliminate the found flaws and errors before a PMU and PDC can be put into real use.

To prevent SQL injection attack, best practice and programming guidelines regarding the database related development such as The Open Web Application Security Project (2014) should be enforced in the developments of services and applications of phasor networks.

To defend against data spoofing attack, one should not solely rely on SSL/TLS or IPSec. For one reason, most of the commercial PMUs on the market does not support SSL/TLS or IPSec features; packet injection and data spoofing is still possible

within an ESP. For another, SSL/TLS and IPSec themselves are not always secure. There also exist attacks targeting SSL/TLS or IPSec, including DNS cache poisoning, man-in-the-middle attacks, TLS/SSL Certificate. Additionally, bugs or errors in the implementation of these security mechanisms can also undermine the security. Therefore, it would be more secure to employ multiple defenses in tandem to minimize the risk of data spoofing attacks when SSL/TLS falls short.

Two additional approaches are recommended. The first one is adopting an intrusion detection system (IDS) at the LANs of PMU or PDC. Instead of detecting data fabrication and spoofing with cryptographic tools, IDS tackles the data spoofing from the perspective of PMU or PDC's behavior and the content of frames. An IDS tries to differentiate abnormal communications from the normal based predefined or self-learned rules. For example, the IDS proposed by Sprabery et al. (2013) defines eleven rules, ranging from frame type check, protocol version check, to data frame size check. Any monitored frame that violates the rules will be deemed as a malformed frame, or a part of the potential attack. Therefore, the incident should be logged and an alert should be generated. Also, the IDS proposed in Yang et al. (2013) utilized the access control whitelists, protocol-based whitelists, statefulness of the C37.118 protocol, range, and length checkers to discriminate illegal frames.

The other way is using redundant devices to take the same measurement independently in separate phasor networks Beasley et al. (2014a). In that case, performing phasor data spoofing attack requires compromising more devices and networks service, which significantly increases the difficulty and cost to perform such an attack.

Chapter 3

Secure Outsourcing of Dynamic Simulations in Power Grid Leveraging Cloud Computing

3.1 Introduction

Dynamic simulations play critical roles in power system research and operations. It is a useful approach to accurately predict the dynamic behaviors of power systems under contingencies such as generator tripping, line switching, and short circuit Chow and Cheung (1992); Chow and Rogers (2000). In real practice, due to the heavy computational burden of simulating large scale power systems, dynamic simulation is currently conducted offline on hourly or daily basis, making it hardly useful in practice to respond to emergent contingencies. For example, for a system of comparable size to the Western Electricity Coordinating Council (WECC), commercial power system software takes approximately 60 seconds to simulate 30-second system dynamics following a single contingency using a local workstation with Intel Xeon(R) 3.2 GHz CPU and 12.0 GB memory. The fact that dynamic simulations are not fast enough

has unfortunately limited its applications in protecting modern power systems against a huge list of critical contingencies.

Recently, outsourcing the dynamic simulations to the cloud has emerged as a promising solution to the problem mentioned above. Pilot studies done by ISO New England ISO New England (2014)Litvinov (2014) showed that by outsourcing the heavy computational burden to the cloud, it is possible to conduct power system simulations, not only much faster but also with less cost. According to Litvinov (2014), an N-1-1 contingency analysis with 4,100 scenarios, which would have taken 1,700 hours at a commodity laptop, or 40 hours at the internal computing cluster of 40 cores, now needs a running time of only 1.5 hours with 150 Amazon EC2 nodes for a total monetary cost of about $60. Additionally, with a commodity PC, it takes around 10 hours to run one study with GE MARS, the Monte Carlo simulation-based tool for power system resources adequacy analysis. In contrast, with 32 C3.large type of Amazon EC2 instances, the running time comes down to 12 minutes and the cost $1.66.

Despite that cloud computing has demonstrated its tremendous potential, the fact that outsourcing requires dynamic simulation to take place in an external and a potentially malicious facility (the cloud) gives rise to concerns about the information security during the course of outsourcing, considering that both information required by and the results produced from the dynamic simulation are sensitive and private to the power grid owners. These concerns have restrained the utility companies from taking advantages of the aforementioned outsourcing paradigms. Instead, they would opt for conservative measures such as keeping all the computations local in exchange for the absolute data privacy assurance. The primary goal of this chapter is to explore the possibility to outsource the computation overhead without giving away data privacy.

3.2 Related Works

This work falls under the broad topic of non-interactive outsourcing of scientific computation. Some early research by Yao et al. (1982); Yao (1986); Chaum et al. (1988) showed that any function can be securely evaluated. More recently, with the breakthrough of fully homomorphic encryption, homomorphic encryption appears as the most straightforward solution to computation outsourcing. Informally, homomorphic encryption allows one (the cloud in our case) to execute encrypted software over encrypted data to generate an encrypted result, which can be decrypted by only outsourcers later. The problem is, however, even the state-of-the-art fully homomorphic encryption Gentry (2009) is too inefficient for practical uses. Relatively more efficient somewhat homomorphic encryption Naehrig et al. (2011) only supports a very limited number of additive or multiplicative operations. As a result, the prospect of directly applying homomorphic encryption to secure the outsourcing of dynamic simulation is remote. Aside from homomorphic encryption, which aims at protecting generic computing, cryptographic tools are also invented to realize the outsourcing of modular exponentiation Hohenberger and Lysyanskaya (2005), the outsourcing of ranked keyword search Wang et al. (2010a), the outsourcing of multiparty back-propagationYuan and Yu (2013), the outsourcing of ridge regression Nikolaenko et al. (2013), the outsourcing of decryptions of attribute-based encryption ciphertexts Green et al. (2011). However, they focus on specific applications, which are irrelevant to dynamic simulations. Additionally, these schemes more or less employ computationally heavy cryptographic primitives like public key cryptography pairing-based cryptography, which will introduce even more latency if they are used to secure the dynamic simulations.

There are also many works on outsourcing of computations that do not heavily rely on cryptographic tools. Wang et al. (2011b) investigates the problem of cloud-based outsourcing of linear programming for large-scale system. Their proposal, however, dealt with only linear programming while the computations in dynamic

simulations mostly involve nonlinear differential algebraic equations. Atallah et al. (2001) studies extensively the problem with respective to secure outsourcing of scientific computations such as sorting, template matching, string pattern matching, and differential equations. However, dynamic simulation is not considered.

3.3 Problem Statement

In this work, we address the problem of privacy-preserving outsourcing of power system dynamic simulations, with the goal of limiting the adversary's capability to glean private information from the computation tasks outsourced by the user.

3.3.1 System Model and Threat Model

3.3.1.1 System Model

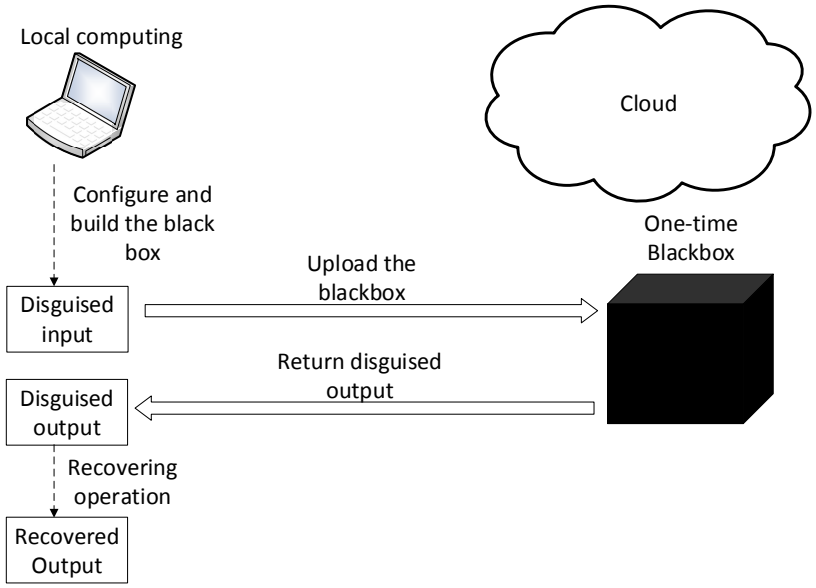


Figure 3.1: Ideal system model for secure computation outsourcing

Figure. 3.1 illustrates the conceptual system model, consisting of two entities: the user and the cloud. The user wishes to outsource to the cloud power system dynamic simulations with privacy preservation, by uploading a configured black box to the

cloud. An ideal black box is a piece of executable software that encapsulates all the information it requires, except for the external input, to complete the computation while looking "unintelligible" to the cloud. The cloud performs all computations through the black box. To preserve the privacy of both the input and output values, the black box is constructed in such as way that it takes protected inputs and produce protected outputs, which will be recovered by the user later.

As data confidentiality is of the primary interest of this work, we enumerate in the following the private data involved in a typical dynamic simulation.

- **Power System Models:** these are the mathematical models, usually nonlinear differential algebraic equations, of machines and control systems, such as generators, governors, motor load.

- **Model data:** the specific parameters and configurations with respective to power system models, including the bus connectivity and information about where and when to apply faults. Model data are usually measured, validated or defined by utility planning engineers.

- **Operating conditions of the system:** the values of system states at the beginning of the simulation. They are updated based on the real-time data from the SCADA (Supervisory Control and Data Acquisition)/EMS (Energy Management System) in the control room.

- **Simulation results:** the output of the contingency simulation in the form of trajectories of state variables over the simulated time.

Data are divided into two categories: input data and output data. Input data are defined as the data submitted to the cloud so that the simulation can be conducted on the cloud end. Output data refer to any information that the simulation produces as the result of the dynamic simulation. Naturally, all the kinds of data listed above except for the simulations results belong to the input data. Simulation results are the

only kind of output data. Both input data and output data are at risk of unauthorized disclosure if left unprotected.

3.3.1.2 Threat model and security goals

In the threat model, the adversary is semi-honest cloud service provider, who is not entirely trusted by the user in the sense that the cloud service provider will honestly carry out the delegated computation tasks and deliver the result of the computations, but it is curious about the computation happening in the cloud, and may try to compromise the user's data privacy by eavesdropping any data stream that flows in and out of the cloud. The security goal of our work is to limit the adversaries' ability to compromise the users' data confidentiality, which, in this particular context, is defined as not any private data listed in Section 3.3.1.1 can be gleaned from the data sent to and produced by the cloud.

3.3.2 Background on Power System Dynamic Simulations

In this section, we briefly review dynamic simulations and use Power System Toolbox (PST from now on) Chow and Rogers (2000); Chow and Cheung (1992), an MATLAB-based open-source simulator, as an example to explain how they work.

3.3.2.1 Dynamic simulations

Dynamic simulation refers to a computer-based approach to study a system's dynamic behavior as a function of time Zeigler et al. (2000). The key idea is to describe the system with a set of mathematical equations, where the variables are the time-varying states of the system. The dynamic states are obtained by solving the equations over the simulated time. As the equations are usually nonlinear, for example, ordinary differential equations or partial differential equations, solving them requires numeric methods such as Euler method, Runge-Kutta method, trapezoidal method, and so forth. In the power system research and operations, dynamic simulations are used to

simulate the time-varying trajectories of the state variables of a power system, such as machine speeds, rotor angles, in, under certain initial conditions and disturbances.

3.3.2.2 Power System Dynamic simulations with PST

PST is an open-sourced MATLAB-based software specialized in power system dynamic simulations. It is developed and maintained by Dr. Joe Chow's research group at Rensselaer Polytechnic Institute. PST was initially developed in 1990s, and the latest version is PST V.3.0. Designed and developed to be run on single PCs, PST does not take data privacy into consideration. In spite that, this open-sourced software serves as a convenient platform for us to apply and test our secure outsourcing scheme. It is worth noting, however, that our proposed secure outsourcing scheme is not unique to PST; it is applicable to any power system dynamic simulations regardless of what simulation software in use.

3.3.2.3 Variable naming convention in the source code

To better explain how to enable security protection by modifying the source code in later sections, we introduce the following variable naming conventions that are used by PST source code. First, the variable name of a state variable's derivative comes from prefixing a d to the name of the state variable. For example, mac_ang denotes the machine angles while $dmac_ang$ denotes their derivatives, *i.e.* the rates of the changes of machine angles. Second, variables in the source codes are matrices, as PST is MATLAB-based. So, we can access elements of a matrix by supplying the matrix with indices. For example, $mac_ang(i, j)$ refers to the element on the ith row and jth column in the matrix. We can also use $mac_ang(i, :)$ to refer the ith row. We say both $mac_ang(i, j)$ and $mac_ang(i, :)$ are references to mac_ang.

3.4 Secure Outsourcing Schemes

In this section, we explain how the proposed scheme protects the privacy of both the input and output data during the outsourcing of dynamic simulations.

3.4.1 Protecting the Simulation Results with the Disguising Technique

In dynamic simulations, simulation results are the trajectories of state variables. A trajectory is obtained by calculating the points on it step by step using numeric integration. The value of a single point is first been by the cloud once it is calculated even before the simulation concludes. So a trajectory must be protected in such a way that when a point is calculated out, its exposure to the cloud does not reveal the point's real values during and after the simulation to the cloud. To this end, considering the form of the output data, we adopt the disguising technique to mask the trajectories while the simulation is still in progress. That is, the original trajectories are transformed into another form so that their actual values remain secrete to the cloud. Later, only the one who knows how the transformation can undo the applied transformation and recover the original trajectories. We elaborate this idea as follows.

3.4.1.1 Using Injective Mappings to Disguise the DAE's solution

Dynamic simulations, in essence, solve a set of differential-algebraic equations (DAE from now on) in order to get the trajectories of state variables that describe the system's behavior. In order to disguise the trajectories, the original DAEs need to be transformed accordingly in the first place.

More specifically, DAEs can be generally represented as

$$\mathbf{F}(\dot{\mathbf{x}}(t), \mathbf{x}(t), t) = 0$$

where $\mathbf{x}$, *i.e.* , the solution of the DAEs, is a time-varying vector holding n state variables. To hide the original solution, *i.e.* , the trajectory $\mathbf{x}(t)$, we define an injective mapping $\mathbf{x_{disguised}}(t) \rightarrow \mathbf{x}(t)$ such that $\mathbf{x_{disguise}}(t)+\mathbf{g}(t) = \mathbf{x}(t)$, where $\mathbf{g}$ are randomly generated cubic spline functions and $\dot{\mathbf{g}}$ is the derivative of $\mathbf{g}$. The disguised DAEs will then be outsourced instead of the original ones as:

$$\mathbf{F_{disguised}}((\dot{\mathbf{x}}_{\mathbf{disguised}}(t), \mathbf{x_{disguised}}(t), t) = 0$$

which is equivalent to:

$$\mathbf{F}(\dot{\mathbf{x}}(t) - \dot{\mathbf{g}}(t), \mathbf{x}(t) - \mathbf{g}(t), t)) = 0$$

Since $\mathbf{g}$ are generated and kept secret by the user, after the cloud returns the disguised solution $\mathbf{x_{disguised}}(t)$, only the user can recover the original trajectories by computing $\mathbf{x}(t) = \mathbf{x_{disguised}}(t) + \mathbf{g}(t)$.

We sketch a proof of the correctness as follows.

Theorem 1. *For a pair of original DAE and its transformed version, denoted by* $F(\dot{\mathbf{x}}(t), \mathbf{x}(t), t) = 0$ *and* $F(\mathbf{sm}(\dot{\mathbf{x}'}(\mathbf{t})), \mathbf{sm}(\mathbf{x}'(\mathbf{t})), t) = 0$*, respectively, where* $\mathbf{sm}(\cdot)$ *is a secret injective mapping for the* $\mathbf{x}(\mathbf{t})$*, if* $\mathbf{x}'$ *is the solution of the transformed DAE, then* $\mathbf{sm}(\mathbf{x}')$ *is the solution of the original solution.*

Proof. If $\mathbf{x}'$ is the solution of the transformed DAE, then the following equation holds.

$$\mathbf{F}(\mathbf{sm}(\mathbf{x}'(\mathbf{t})), \mathbf{sm}(\mathbf{x}'(\mathbf{t})), t) = 0$$

Let $\mathbf{x}'' = \mathbf{sm}(\mathbf{x}')$, substituting $\mathbf{sm}(\mathbf{x}')$ with $\mathbf{x}''$, we have

$$\mathbf{F}(\dot{\mathbf{x}}'', \mathbf{x}'', t) = 0$$

As we also have the original DAEs $F(\dot{x}(t), x(t), t) = 0$, then we may conclude that $\mathbf{x}''$ is also the solution of the original DAEs. □

3.4.1.2 Choice of Injective Mappings

To achieve the best hiding without incurring too much computational overhead, we note that the secret injective mapping M should satisfy the a number of properties summarized as follows.

1. No ambiguity. This property requires that the mapping, $\mathbf{F}$, should be one-on-one. That is, given the disguised result, there is only one corresponding original result.

2. Derivable. The mapping, which intends to replace the initial state variable, needs to be derivable because it often requires computing a state variable's derivative in power system simulations. Otherwise, singularity may occur, leading to numerically unstable disguised DAE is not numerically stable. For this reason, the traditional XOR masking would not help in our case.

3. Random. If the mapping is deterministic or predictable, the adversary can guess the original result by merely looking at the disguised trajectories, and, thus, there is no effect of hiding at all. Examples that do not satisfy this property include adding to the state variable a constant number or a sinusoidal function of a constant frequency or multiplying the state variable with a deterministic value.

4. Efficient to evaluate. The outsourcing scheme should not incur much overhead by evaluating the extra mappings. Otherwise, the outsourcing may require too much effort on the computation that does not directly relate to the simulation, and the overall efficiency and economy of outsourcing will suffer.

Inspired by Atallah et al. (2001), we find that adding a well-chosen cubic spline functions to the original state variables in the DAEs is a suitable choice of secret injective mapping. Atallah et al. (2001) used cubic spline functions to disguise the solution of an ordinary differential equation. More specifically, the approach maps

the initial solution to the addition of the initial solution and a randomly generated spline function of time defined over the same domain of the initial solution. We adopt this idea in our design.

A cubic spline function Reinsch (1967), is defined as a piecewise cubic polynomial, which joins in the knots obeying continuity conditions. Knots refer to as points where the two neighboring pieces connect. A cubic spline function stipulates that neighboring parts are continuous at their sharing knot and have the same second and first order derivative. To obtain a random spline function, we first set a number of randomly generated knots and then use efficient cubic spline interpolation to connect these knots.

A well-chosen spline functions can satisfy all properties listed above.

First of all, adding a spline function to the original state variable is a one-on-one mapping, so one can always recover the original result by removing the effect of spline functions from the disguised result without ambiguity. Second, each piece is a cubic polynomial. Neighboring pieces must have the same second-order derivative, so, by definition, splines function are differentiable at any point in their domain. Thirdly, spline functions are based on a large number of randomly generated knots. A cubic spline function with m knots involves ($m + 1$ reginons) $\times$ (4 parameter per region) $-$ (m knots) $\times$ (3 constraints per knot) $= m + 4$ parameters. As a result, in order to determine a cubic spline function with m knots, one needs to guess all the $m + 4$ random values. If m is large enough, this will become extremely difficult. Fourthly, spline functions are highly configurable. We can configure them such that they have approximately the same range as the original trajectories to ensure the best disguising effect. Lastly, a spline function comprises of polynomials, which are efficient to evaluate In our design, by limiting the degree of the spline function to three, we can achieve small computation overhead.

To illustrate how spline functions are applied to disguise the output data, we take an exemplary two-area three-generator power system model that can be described with the following DAEs. Note that, however, in the example, the linear model is

chosen only for ease of illustration; the disguising technique using well-chosen spline functions as the injective mapping is also applicable to arbitrary non-linear models, as described in the last Section 3.4.1.1.

$$x_1' = x_4 \tag{3.1}$$

$$x_2' = x_5 \tag{3.2}$$

$$x_3' = x_6 \tag{3.3}$$

$$x_4' = \pi f/H_1 \cdot (P_{m1} - P_{e1}) \tag{3.4}$$

$$x_5' = \pi f/H_2 \cdot (P_{m2} - P_{e2}) \tag{3.5}$$

$$x_6' = \pi f/H_3 \cdot (P_{m3} - P_{e3}) \tag{3.6}$$

x_1 through x_6 are state variables. $P_{mi},\ H_i,\ P_{mi}$ are obtained from the power flow analysis and $P_{ei} = \sum_{j=1,2,3} E_i E_j Y_{i,j} cos(th_{i,j} - x_i + x_j)$, where $Y_{i,j}$ and $th_{i,j}$ are also results from power flow analysis. They are considered as invariant during one step of numerical integration.

First, apply the disguise technique to the state variables, *i.e.* let $x_{iD}(t) + g_i(t) = x_i(t)$, for $i = 1, 2, .., 6$, where the subscript D means 'disguised' and g_i are randomly generated cubic spline functions. These substitutions give the disguised version as follows.

$$x_{1D}' + g_1' = x_{4D} + g_4 \tag{3.7}$$

$$x_{2D}' + g_2' = x_{5D} + g_5 \tag{3.8}$$

$$x_{3D}' + g_3' = x_{6D} + g_6 \tag{3.9}$$

$$x_{4D}' + g_4' = \pi f/H_1 \cdot (P_{m1} - P_{e1}) \tag{3.10}$$

$$x_{5D}' + g_5' = \pi f/H_2 \cdot (P_{m2} - P_{e2}) \tag{3.11}$$

$$x_{6D}' + g_6' = \pi f/H_3 \cdot (P_{m3} - P_{e3}) \tag{3.12}$$

where g_i' is the derivative of the spline function g_i and the algorithm can easily calculate it given the expression of g_i. We then outsource the disguised DAEs to the cloud and let the cloud solve for the disguised trajectories of state variables. Upon receiving the disguised trajectory, we may obtain the original state variables' trajectories by applying the secret mapping, as

$$x_i(t) = x_{iD}(t) + g_i(t),\ i = 1, 2...6$$

3.4.1.3 Implementation of the Disguising Technique Using Spline Functions

To disguise a state variable, it requires replacing all references to the state variable and all references of state variable's derivative, with their respective injective mapping, *i.e.* adding the corresponding random spline function.

Take the state variable *mac_spd*, a n-by-t matrix denoting the speeds of all the n machines in the power system over t time instant. To disguise *mac_spd*, reference like $mac_spd(i, j)$, which indicates the n-th machine's speed at the jth instant, and $mac_spd(i, :)$, which refers to the ith machine's speed over the simulated time, are to be replaced by $mac_spd(i, j) + disg_mac_spd(i, j)$, and $mac_spd(i, :) + disg_mac_spd(i, :)$, respectively. In addition, reference of the state variable's derivative like $dmac_spd(i, j)$ should be replaced with $dmac_spd(i, j) + disg_dmac_spd(i, j)$. Here, *disg_mac_spd* denotes the spline functions used to disguise *mac_spd*, where the prefix *disg* stands for disguising, and *disg_dmac_spd* is the derivative of *disg_mac_spd*. In practice, *disg_mac_spd* is implemented as a function $disg_mac_spd(k, t)$ that takes three parameters: k the machine's number, and t the time instant. The function outputs the value of the k-th spline function at time instant t. Similarly, $disg_dmac_spd(k, t)$ outputs derivative of the k-th function in *disg_mac_spd* at time instant t.

More specifically, for example, we replace $mac_spd(i, :)$ with $mac_spd(i, :) + disg_mac_spd(i, :)$ by three steps: 1) extracting the name of the variable, "*mac_spd*"

and its associated indices, "$(i,:)$". 2) adding the disguising prefix to the variable name to get the name of the disguising term *disg_mac_spd*, 3) supplying the same indices to the disguising term $disg_mac_spd(i,:)$, and 4) replacing $mac_spd(i,:)$ with $mac_spd(i,:) + disg_mac_spd(i,:)$. The same procedures apply to replacing an arbitrary state variable and its derivative with their respective disguising terms.

When we try to implement the replacing method, we encounter two problems. The first is that nested references make it difficult to extract the index. For example, to extract the indexes in $mac_spd(get_fault_line_index(i,:),:)$, traditional approach of Regular Expression would fail. The reason is that Regular Expression only linearly scans the code and extracts anything between a left parenthesis and the first right parenthesis it scans afterward and ignores the fact that the two parentheses might not make a pair. In this example, the extracted index, thus, is, $get_fault_line_index(i,:$, which is incomplete and meaningless.

The second problem is that the resulted new source codes after applying the replacing method may no longer be valid. For example, the initial source code looks

```
mac_spd(:,j) = mac_spd(:,k)+h_sol * dmac_spd(:,k)
```

After disguising `mac_spd`, it becomes

```
mac_spd(:,j)+disg_mac_spd(:,j) = mac_spd(:,k) + disg_mac_spd(:,k) + h_sol
* ( dmac_spd(:,k) + disg_dmac_spd(:,k))
```

The newly generated code is no longer valid nor executable since the equal sign "=" means assigning the value of the right operand to the left operand. However, the left-hand side of the new code is no longer a variable. In this example, the problem could be solved by rewriting the new code to:

```
mac_spd(:,j) = -disg_mac_spd(:,j) + mac_spd(:,k) + disg_mac_spd(:,j) +
h_sol *(dmac_spd(:,k) + disg_dmac_spd(:,k)).
```

Taking both the problems into account, we design a source code modification algorithm based on parentheses counting and operator recognition. To solve the first problem, we let the algorithm keep track of matched parentheses and extracts the outermost pair of parentheses and the contents they enclose. To solve the second

problem, when an "=" is scanned in the source code, we move all terms but the original state variable from the left-hand side to the right-hand side. We illustrate our algorithm in Algorithm. 1, where the symbol "||" means concatenating.

3.4.2 Ensure Input Privacy Through Code Obfuscation

3.4.2.1 Code Obfuscation Basics

In the previous section, we use the injective mapping to mask the output to maintain the output confidentiality. However, there is yet nothing so far to prevent the adversary from peeking into the source codes and discovering where and how the disguise techniques. In that case, the adversary is free to remove the mask and recover the initial source code. More importantly, regardless of whether the disguised technique has been applied to disguise the output, the source code also contains all the input data discussed in Section 3.3. To overcome this, we propose to make use of code obfuscation to enhance the security protection of the input data.

Although Barak el al. have proved the theoretical impossibility of implementing general-purpose obfuscators in Barak et al. (2012), code obfuscators are widely used in practice for the intellectual property protection. Software vendors who do not want to share their source codes with others, will only deploy the obfuscated and compiled software to the end users and make it difficult for reverse engineering and analysis. The obfuscated software is equivalent to the original one in terms of functionality, but it deliberately hides implementation details.

Collberg et al. (1997) gives a definition of an obfuscating transformation as follows:

Definition (obfuscating transformation) Let $P \rightarrow P'$ be a transformation of a source program P into a target program P'. $P \rightarrow P'$ is an *obfuscating transformation*, if P and P' have the same observable behavior. More precisely, in order for $P \rightarrow P'$ to be a valid obfuscating transformation, the following conditions must hold.

- If P fails to terminate or terminates with an error condition, then P' may or may not terminate.

Data: The source code;
{**Vars**, **dVars**} names of state variables and their derivatives to be disguised
Result: Modified source codes
forall the S *in* {**Vars**, **dVars**} **do**
  forall the *occurrences of "S(" in the source code* **do**
    Initialize character *current_character* ← the character next to "$S($";
    Initialize integer *unmatched_parenthesis* ← 1;
    Initialize character *start_of_indices* ← *current_character*;
    while *unmatched_parenthesis is not 0* **do**
      if *current_character is "(" then*
        increment *unmatched_parenthesis*;
      if *current_character is ")" then*
        decrement *unmatched_parenthesis*
      move *current_character* to the next;
    end
    Initialize character *end_of_indices* ← *current_character* ;
    Initialize character string *indices* ← substring between *start_of_indice* and *end_of_indice* (inclusive);
    `/* determine if the current reference is on the left-hand side of a equals sign (''='') */`
    current_character ← the next character after *end_of_indices*;
    Initialize character *operator* ← *current_character*;
    `/* Initialize the string to replace the original reference */`
    Initialize character string *rep_str* ← an empty string;
    if *operator equals "=" then*
      rep_str ← $S||indices||" = -disg_''||S||indices$;
      replace the reference with *rep_str*
    else
      rep_str ← "("$||S||indices||" + disg_''||S|indices||$")";
      replace original $S||indices$ with *rep_str*;
    end
  end
end

Algorithm 1: Source code modification algorithm

- Otherwise, P' must terminate and produce the same output as P.

Various obfuscation techniques have been proposed and used in reality. These include layout transformation, control transformation, ordering transformation. They have different level of potency and can be applied independently. We refer the reader to Collberg et al. (1997) for more details on code obfuscation.

In our implementation, we use MATLAB's *pcode* functionality as a surrogate of the general-purpose code obfuscator. The *pcode* functionality is provided in MATLAB. It takes the MATLAB source code and transforms it into the preparsed and encoded version, so the resulting code becomes intelligible but works the same as the original code. Apparently, the *pcode* satisfies the definition of code obfuscation. Note that *pcode* is used here to represent the general obfuscation process. More sophisticated and secure obfuscations can be chosen depending on the level of protection. To our best knowledge, there is not yet any known active attack against the *pcode*.

3.4.3 Framework

By now, we can summarize our scheme by introducing as follows the framework of our proposed scheme.

- $\mathsf{SplineGen}(n, n_k, t_{start}, t_{end}, R_{estimate}) \rightarrow \mathbf{\Phi}$. This is a randomized algorithm that takes as input, n, the number of state variables to disguised, n_k the number of knots for each spline function, t_{start}, t_{end}, the starting and ending time of the simulation, and $R_{estimate}$ the estimate range of the state variables. It outputs the splines functions for the n state variables and stored spline functions in $\mathbf{\Phi}$. A spline function is obtained by first randomly generating the knots , and then connecting the knots by Lagrange interpolation.

- $\mathsf{SplineEva}(\Phi, \mathsf{k}, \mathsf{t})$ This function evaluates the k-th spline function in Φ at the time instant t, where $t \in [t_{start}, t_{end}]$, and returns the value.

- $\mathsf{ProbEnc}(\mathbf{\Phi}, \mathbf{\Psi}) \rightarrow \mathbf{\Psi}'$ This function takes as input Ψ the original problem in the form of simulation source code, as well as $\mathbf{\Phi}$, the spline functions generated by SplineGen. The function applies disguising technique to all the state variables to be protected. Afterward, it performs code obfuscation on the modified source code to generate the encrypted simulation source code, what is the output of $\mathbf{\Psi}'$.

- $\mathsf{SimulationExec}(\mathbf{\Psi}') \rightarrow \mathbf{S_{disguise}}$. This function executes the a feThis function takes as input an encrypted simulation, performs the encrypted simulation, and produces the disguised simulation result $\mathbf{S_{disguise}}$, by solving the disguised DAEs.

- $\mathsf{ResultRec}(\mathbf{S_{disguise}}, \mathbf{\Phi}) \rightarrow \mathbf{S}$ This function takes as input $S_{disguise}$ the disguised trajectories, $\mathbf{\Phi}$ the spline functions used to disguise the original state variables. It recovers the original trajectories by adding together the point on the disguised trajectories and its corresponding masking values, which are obtained by calling SplineEva.

In accordance with the system model illustrated in Section 3.3.1.1, The user performs SplineGen and ProbEnc to produce the encrypted simulator. It then uploads the encrypted problem to the cloud. The cloud performs $\mathsf{SimulationExec}(\mathbf{\Psi}')$ and returns the disguised trajectories to the user. Finally, the user calls ResultRec to recover the right trajectories from the disguise ones.

3.4.4 Experiment Results

We implemented our secure outsourcing scheme and tested whether the scheme would hide the simulation output data and whether we can recover the original trajectories from the disguised ones. As a cloud computing application, our implementation involves two parts: the cloud end and the client end. The cloud end is a service running in the cloud that receives and executes MATLAB codes sent from the client

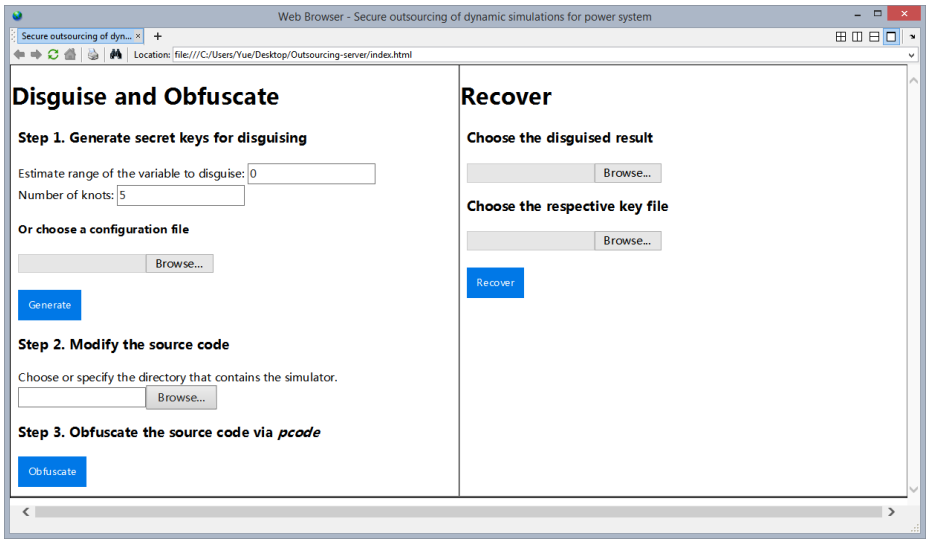
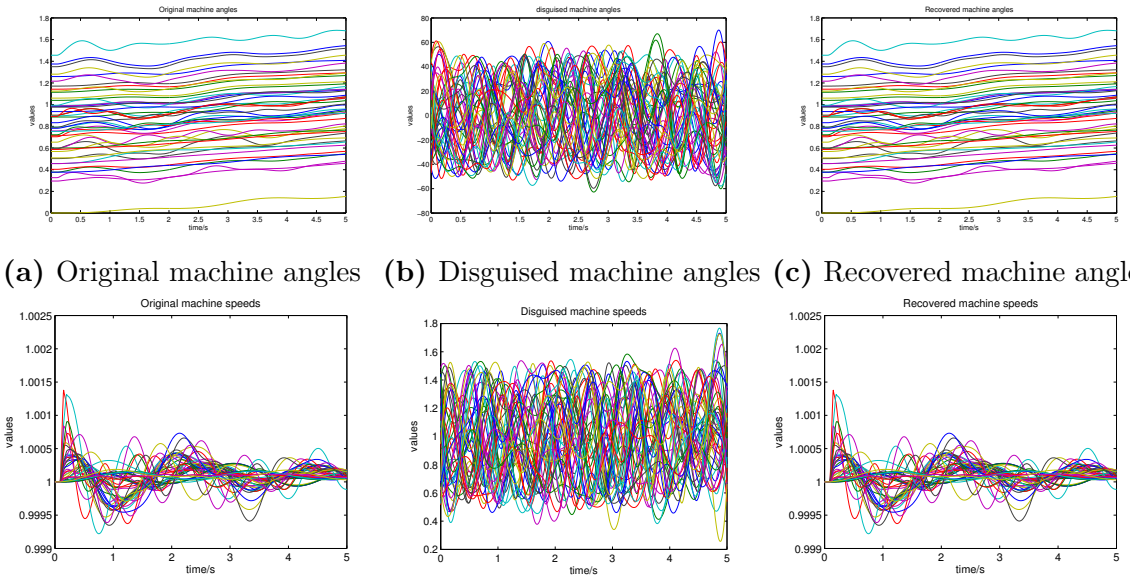


Figure 3.2: Client side UI for secure outsourcing of dynamic simulations Left half is the "Disguising and obfuscating" component, which is responsible for processing the source code to protect the input data confidentiality before uploading the source code. The right half is responsible for recovering the disguised trajectory received back from the cloud.

side and returns the result back upon completion. The client end consists of the source code modifier and code obfuscator. Source code modifier is a python program that first inserts the codes responsible for generating disguising spline functions to the original source code and then runs Algorithm.1. The code obfuscator makes use of the *pcode* function to obfuscate the modified source code. Fig. 3.2 illustrates the user interface of the client software.

To demonstrate the correctness of our scheme, we tested it on the NPCC 48-machine, 140-bus power system model. The model represents the backbone system of the northeast region of the North American East Interconnection.

Figure. 3.3 delineates the original (unmasked), disguised (masked), and recovered trajectories of the generators' angles and speeds. It is shown that the original and the recovered trajectories are identical. Taking the difference of the two, which is a zero vector, verifies our observation. By that, the outsourcing scheme can disguise the original result and also recover the disguised result without any error.



(a) Original machine angles **(b)** Disguised machine angles **(c)** Recovered machine angles

(d) Original speeds **(e)** Disguised speeds **(f)** Recovered speeds

Figure 3.3: Original, disguised, and recovered curves of machine angles and machine speeds

3.5 Performance analysis

We base our analysis on the following settings.

- The simulation to in involves N state variables and s time steps.
- The original simulator uses the fixed time step trapezoidal method for numerical integration, which is also commonly used by default in most commercial dynamic simulation software packages like PSS/E according to Fu (2011).
- Each spline function contains k random knots.
- Evaluating a spline function at a certain point is of constant complexity.
- Disguising one state variable in the source code is of constant complexity.
- Obfuscating a program with a code obfuscator is of constant complexity.

3.5.0.1 Local End Overhead

The local end performs SplineGen, ProbEnc, ResultRec. Based on our assumptions, in Big-O notations, the complexity of ProbEnc is $O(N)$ since it handle N state variables, with each taking constant time; SplineGen is also of $O(Nk)$ complexity since it generates a spine function for N state variables. Each requires Lagrange interpolation over k knots, which is of $O(k)$. ResultRec is $O(Ns)$, as it needs to recover N state variables' trajectories, and each of the trajectories is s long. Overall, the local overhead is of $O(Nk + N + Ns) = O(Ns)$ complexity, as $s >> k$. As seen, the complexity is independent of the complexities of power system models of the power system.

3.5.0.2 Cloud End Overhead

The computational cost that is pertaining to solving DAEs dominates the computational overhead in the cloud end. With fixed-step trapezoidal method, the per step complexity to calculate all the N state variables is given by $O(N\rho)$, where ρ indicates the average complexity of evaluating the functions of state variables in the power system model. Accordingly, the complexity required by solving for the N state variables' trajectories over s time steps is $O(Ns\rho)$. With massive and complex system models, *i.e.* a large ρ, the computation complexity of the cloud end is enormous and unpredictable.

Without outsourcing, the complexity $O(Ns\rho)$ will be undertaken by the user with their limited computation resources. In contrast, thanks to outsourcing, this computation burden will be migrated to the cloud, which possesses much more powerful computing capabilities to speed up the computation, leaving the local end with the much smaller and a predictable complexity of $O(Ns)$. As a result, users will not spend more time to outsource the simulation to the cloud than conducting the simulation on their own. By that, with the proposed secure outsourcing scheme,

users can expect to have savings on great local computation and time savings by outsourcing while preserving the privacy of both input and output data.

3.6 Discussions

We understand that code obfuscation used in our scheme has its limitation in that given enough time, and resources, a determined adversary will be able to reverse engineer any computer program protected by code obfuscation. However, we argue that, if properly used and with the advances in code obfuscation techniques, code obfuscation can still build up a strong defense against potential privacy breaches during the course of computation outsourcing. Also, the privacy protection can be further fortified by using the proposed secure outsourcing scheme in tandem with other higher-level strategies.

First, the software industry has been leveraging code obfuscation for the purpose of the intellectual property protection Collberg et al. (1997). As a typical tactic, if reverse engineering the current version of a software program takes even more efforts and time than developing a new and more advanced version, the value of reverse engineer is mostly diminished. Likewise, in power system dynamic simulation, information such as the transient state of the power system, although sensitive, expires in a short period after the contingency gets resolved. Hence, the same logic applies here. As long as code obfuscation technique can thwart the reverse engineering before the sensitive information expires, we consider that our scheme successfully defends against the attack. In fact, code obfuscation does show great usefulness when it comes to stalling the attacker. According to Collberg *et al.* Collberg et al. (1997), it is viable to apply obfuscating transformations in polynomial time but remove them in exponential time. Mariano *et al.* Ceccato and Penta (2009) reported that, by just using identifier renaming, it significantly decreased the efficiency of attackers, at least doubling the time needed to complete a successful attack even against the best attacker. Wroblewski (2002) conducted empirical studies that drew a similar

conclusion that it is possible to provide an empirically full protection of the source code by code obfuscation. Concretely, it takes 100 experienced crackers to work a year, 4 hours per day, to understand the meaning of a compiled and obfuscated program that contains 1,400,000 to 70,000,000 instructions and are protected by the opaque construct technique. Linn and Debray (2003) states that the best commercially available disassembly tool fails to disassemble correctly over 65% of the instructions, and 85% of the codes obfuscated with their obfuscation technique.

Secondly, as stated earlier, note that the code obfuscation we used in this work serves as a surrogate of the secure general-purpose software obfuscation. General-purpose software obfuscation aims at making any software under protection "unintelligible" while ensuring it is functionally equivalent to the original software. Recent researches on general-purpose obfuscation, such as Garg et al. (2013), Sahai and Waters (2013), and Barak et al. (2014), have achieved a series of progresses. With Garg et al. (2013), Sanjam *et al.* constructed a candidate indistinguishability obfuscator for all polynomial-size circuits, using both Multilinear Jigsaw Puzzles and fully homomorphic encryption Gentry (2009). As suggested by this work, this approach could finally evolve to a general-purpose black box obfuscation for an arbitrary program, by determining the classes of amenable functions to their scheme. When the general-purpose secure obfuscation becomes mature and ready for practical uses, we can quickly replace *pcode* employed above in the dissertation with provable-secure obfuscation without changing the system model illustrated in Fig. 3.1.

Last but not least, incorporating other non-cryptographic strategies is also helpful to increase the strength of protection. For example, to privately evaluate the product of two matrices without revealing the matrices, **Protocol 2** in Du and Atallah (2001) proposes to hide the authentic matrix in an enormous number of similar matrices. In a similar fashion, we can hide the authentic simulation among a batch of randomly generated bogus but similar simulations. In order for a successful attack, the attacker will need to analyze all of the outsourced simulations, which significantly raises the difficulty to launch an attack.

Chapter 4

Privacy-preserving Spectral Estimation of PMU Data for the Monitoring of Inter-area Oscillations

4.1 Introduction

Inter-area oscillations are detrimental to the stability of the power transmission grids. Poorly damped oscillations reduce the capability of the power transfers and even lead to widespread blackouts. Incidents such as Western Interconnection Blackout on Aug. 10 (1996), US Blackout on Aug. 14 (2003), Italian Blackout on Sept. 28 (2003) have demonstrated the serious damages that inter-area oscillation can make. Therefore, in order to keep the lights on, it would be very helpful to continuously track the oscillation conditions and provide the control room with early warnings before a potential oscillation develop into a real catastrophic inter-area oscillation.

Recently, using real-time measurement data collected by ambient phasor measurement units (PMU) to monitor and analyze the inter-area oscillations has been

shown effective. As one of the key enabling technology for the smart grid, PMU technology, a.k.a, synchrophasor technology makes real-time monitoring of power systems possible. The data collected by PMUs include instant voltage, current, rotor angle, and frequency. PMU measurement data have been utilized to 1) support real-time grid operations, 2) improve system planning and analysis, and, 3) response-based control applications North America Electric Reliability Corporation, NERC (2011). When it comes to the study of inter-area oscillations, or more broadly low-frequency oscillations, a host of research Sun et al. (2012); Vanfretti et al. (2013); Chakrabarti et al. (2009) use ambient PMU measurements and apply frequency domain approaches to monitor, detect, and analyze oscillations. The basic idea is to monitor the frequency domain of PMU measurements, and using methods such as Frequency Domain Decomposition (FDD) Brincker et al. (2000, 2001), computing risk scores Sun et al. (2012), or tracking the oscillation modes Vanfretti et al. (2013), so that useful information and more insights about the oscillations can be acquired.

For these frequency domain approaches, the spectral estimation is the starting point. To obtain spectral estimations, we note that existing frequency domain approaches all assume they have unrestricted access to all the measurement data of the ambient PMUs. Nonetheless, privacy concerns regarding the use of PMU measurement data exist in real practice. In the current deregulated power market, the transmission grid comprises competing utility companies running their own power systems under the coordination of independent system operator (ISO). The utility companies consider the PMU measurements taken at their own power system private and sensitive, as *North American Electricity Reliability Corporation Confidentiality Agreement for Electric System Operating Reliability Data* North American Electric Reliability Corperation (2009) points out that *"the operational reliability data may contain proprietary information and unequal access to operational reliability data may result in unfair advantages and disadvantages in the electricity market"*. In fact, according to EIOC (2013), *"Since deregulation, a competitive environment has arisen and complete information about the state of the power grid is seldom available*

to operators". Although legal agreements like North American Electric Reliability Corperation (2009) exist to try to ensure the privacy of sensitive data during their use in research and operations, there are no technical guarantees. As a consequence, in spite of benefits, the lack of privacy protection seriously discourages utility companies from sharing their data.

Our contributions In this work, we take the first step to investigate privacy-preserving multiparty spectral estimations to demonstrate the feasibility to conduct spectral estimation without losing data privacy. We address this open problem mainly leveraging the cloud computing and Paillier cryptosystem Paillier (1999). The main idea of the proposed scheme is to let each utility encrypt their PMU measurements; leveraging the additive homomorphism of the Paillier cryptosystem, the cloud performs Fourier transform directly over the encrypted measurement data without the need to decrypt for the original data. The cloud then sends the result of the privacy preserving spectral estimation, *i.e.* the encrypted spectrum estimation in the form of encrypted frequency coefficients, to the operator in the control room. The operator uses the private key to decrypt for all the encrypted coefficients.

Besides privacy losses caused by unauthorized disclosure of the measurement data in the time domain, there also exist data privacy concerns in the frequency domain. The spectrum in the frequency range of the inter-area oscillation should be disclosed to the operator for monitoring purpose. The spectrum beyond the inter-area oscillation frequency range may contain information that is irrelevant to the inter-area oscillations, but still useful to participating parties other than the operator. Accordingly, it is desirable to have a mechanism to enforce access control in the frequency domain to grant access to a particular frequency range to designated parties. Later, we will explain how the proposed scheme achieves this feature by utilizing random masks.

Note that while the proposed scheme in this chapter is based on the discrete Fourier transform (DFT), it can also be adapted to more sophisticated Fourier-based power spectral estimations such as Welch method, multitaper method, etc, in a similar

way. Neither the application of privacy preserving spectrum estimation is limited to monitoring the power grid; it can also be applied to any other areas involving frequency domain analyses of sensitive data such as health care, consumer behavior analysis, *etc.*

Our contribution is summarized as follows:

- To our best knowledge, for the first time, this work considers the privacy preserving spectral estimation in the multiparty scenario.
- The proposed scheme not only preserves the participants' privacy in the input data in the time domain but also achieves fine-grained access control in the frequency domain.
- Software prototype of the proposed scheme is developed and tested. Taking into account the characteristics of spectral estimations in the power grid, effective optimization are proposed to reduce the latency incurred by the extra cryptographic computations, and, thus, make the proposed method even more practical.

We organize the rest of this chapter as follows. Section.4.2 reviews the related works. Section.4.3 talks about the background knowledge. Section.4.4 first introduces settings and system model and formulates the problem we would like to solve. The section continues to presents the proposed scheme and discusses optimization techniques that make the proposed scheme more efficient. Section.4.5 carries out the security analysis. Section.4.6 covers the complexity analysis with respective to the communication and computation overheads. Section.4.7 presents the experiment results of our proposed scheme under stress testings and the a test with the a simplified model of WECC AC transmission system.

4.2 Related Works

Our work falls into the broad category of non-interactive privacy-preserving signal processing. Although, to our best knowledge, our proposed scheme is the first to deal with privacy-preserving spectral estimations involving multiple parties, we have drawn significant inspiration from the previous research. The most closely related work is Bianchi et al. (2009) by Bianchi *et al.* , which demonstrates the feasibility of carrying out Fourier transform in the encrypted domain. Lagendijk et al. (2013) survey existing privacy-preserving signal processing utilizing homomorphic encryption and secure multiparty computation. Both the works above, however, deal with only the minimal scenario where a single party wishes to offload the computation to an untrusted processing device. Additionally, no access control over the output data is considered.

Our scheme aims at addressing utility companies' privacy concerns in smart grid monitoring; previous researches on privacy preservation in smart grid, however, mainly focus on the privacy of customer data. Shi *et al.* Elaine Shi et al. (2011) proposes a scheme that enables privacy-preserving aggregation residential power consumption on a untrusted aggregator. This scheme only works for aggregation of encrypted data. Similar problems are also investigated regarding consumers' privacy protection in Borges et al. (2014); Thoma et al. (2013); Baharlouei and Hashemi (2014). A very few existing works consider the data privacy problem of operational data in the bulk power system. Tong *et al.* presents a secure data sharing scheme for situational awareness for the power grid in Tong et al. (2013). It is, however, limited to secure data archiving and access control for the power flow data; the processing and manipulation of the shared data still need to be conducted in a fully trusted environment.

4.3 Preliminaries

4.3.1 Fourier Transform

Fourier transform is a widely-used fundamental signal processing tool to obtain the spectral estimation of a signal. An M-point discrete Fourier transform (DFT) of a finite time-series signal $x(n)$ of length T is given by:

$$X(k) = \sum_{n=1}^{T} x(n)W^{nk}, k = 1, 2, ..., M$$

where $W = e^{-\frac{j2\pi}{M}}$. The $X(k)$ is a complex number that indicates the signal's magnitude and angle in the k-th frequency component. In the rest of the chapter, unless otherwise stated, we assume $M = T$, which is a typical choice without loss of generality.

4.3.2 Paillier Cryptosystem

Paillier cryptosystem Paillier (1999) is based on the decisional composite residuosity assumption. Paillier cryptosystem is widely known as an additive homomorphic cryptography. It enables one to compute the encryption of the sum of two value, say $m_1 + m_2$, given only the encryptions of m_1 and m_2, without the need to decrypt for the values. We briefly revisit Paillier cryptosystem in this subsection.

Key Generation Let p and q be big prime numbers and $N = pq$, such that N is hard to factorize. Let the private key be the least common multiple of $(p-1, q-1)$, *i.e.* ,$K_{priv} = lcm(p-1, q-1)$. Let $g \in \mathbb{Z}^*_{N^2}$ an element of order αN for some $\alpha \neq 0$. The public key is the tuple (N, g).

Encryption To encrypt a value m, where $m < N$, generate a random value $r < N$, and use the public key to encrypt the value. The ciphertext is constructed as: $c = Enc(K_{pub}, m) = g^m r^N \mod N^2$.

Decryption Given the ciphertext ct, one can decrypt for the plaintext with the private key, as

$$m = Dec(K_{priv}, ct) = \frac{L(c^{K_{priv}} \mod N^2)}{L(g^{K_{priv}} \mod N^2)} \mod N$$

where $L(x) = \frac{x-1}{N}$.

Additive homomorphism Given the ciphertext of m_1 and m_2, which are $g^{m_1} r_1^N$ and $g^{m_2} r_2^N$, respectively, one can compute the ciphertext of $m_1 + m_2$.

$$\begin{aligned} Enc(K_{pub}, m_1) Enc(K_{pub}, m_2) &= g^{m_1+m_2} (r_1 r_2)^N \\ &= Enc(K_{pub}, (m_1 + m_2)) \end{aligned} \quad (4.1)$$

$Enc(K_{pub}, m_1 + m_2)$ can be decrypted with the same private key that is used to encrypt m_1 and m_2.

Homomorphic multiplication of plaintext Given the ciphertext $g^m r^N$, one can also calculate the ciphertext of θm, where θ is a constant number, by computing

$$(g^m r^N)^\theta = g^{\beta m} (r^\theta)^N = Enc(K_{pub}, \theta m)$$

4.3.3 Monitoring Oscillation in Power System

Figure. 4.1 depicts the classical two area power system introduced in Kundur et al. (1994) with two generators in each area. Inter-area oscillations may develop at the interface 12-34. That is between the generator cluster $C1$, consisting of generators $G1$ and $G2$, and the generator cluster $C2$, consisting of generators $G3$ and $G4$. Assume that the PMUs installed at the respective generators monitor the generators' rotor angles. Then, potential oscillations can be monitored and analyzed by continuous tracking the spectra of the averaged angles of the two areas.

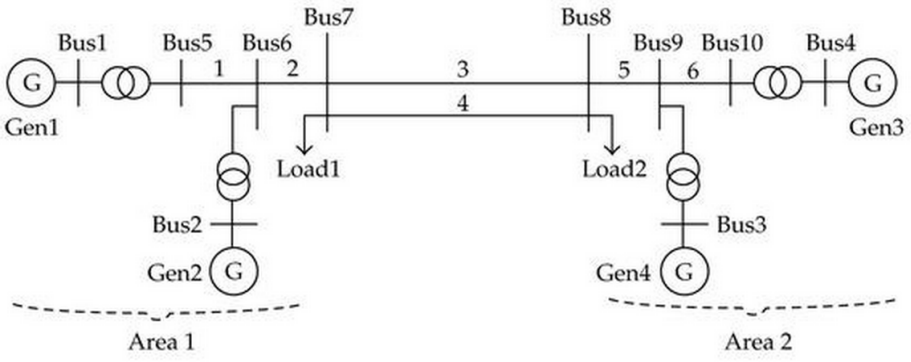


Figure 4.1: Two area four generator power system from Kundur et al. (1994)

We denote the time series of rotor angles of the four generator as δ_1, δ_2, δ_3, and δ_4, respectively. The angle of Area 1 is represented by the averaged rotor angle of the two generators in the area, *i.e.* $\delta_I = \frac{1}{2}(\delta_1 + \delta_2)$. Similarly, the angle of Area 2 is given by $\delta_{II} = \frac{1}{2}(\delta_3 + \delta_4)$. Thereby, what we would like to monitor is the spectrum of $\tilde{\delta}_I$ and $\tilde{\delta}_{II}$, where $\tilde{\delta} = \delta - mean(\delta)$ denote the mean centering δ.

4.4 Our Proposed Scheme

4.4.1 Problem Formulation

4.4.1.1 System Model and Entities

The system model of the proposed scheme is captured in Fig. 4.2. Each participating utility has several PMUs reporting their real-time measurements of rotor angles to the cloud servers. Since the PMU measurement data must be protected against unauthorized access of peer utility companies, cloud servers, and the control room, each PMU sends to the cloud servers the measurement data in their encrypted form. Next, the cloud servers receive the encrypted measurements and perform relevant computations to securely obtain the encrypted spectrum estimation without knowing the real values of the input data. The cloud then forwards the encrypted spectrum to the control room. Finally, the operator at the control room receives and decrypts the

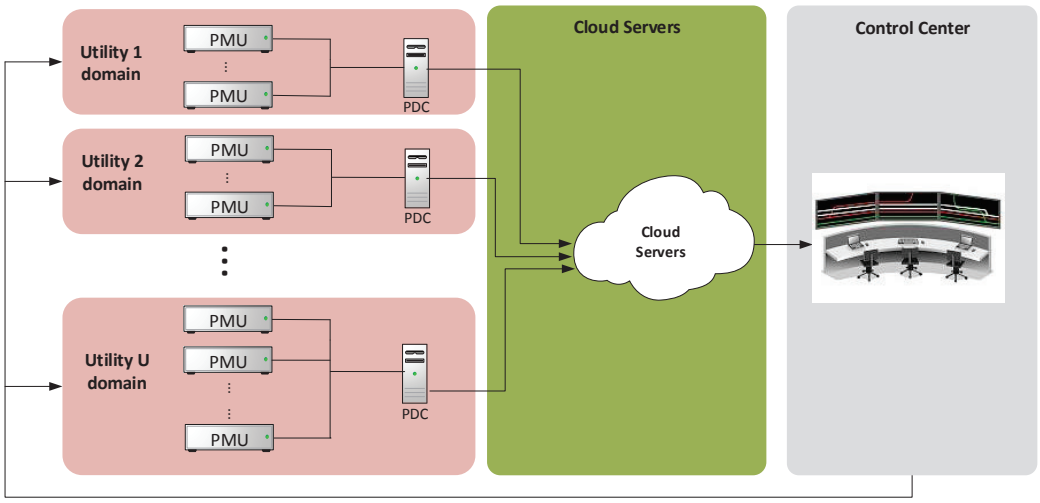


Figure 4.2: System Model of privacy-preserving processing of PMU data

spectrum. There also exists secure communication channels between utility companies and the control room for key assignments and other communications.

We justify the introduction of the public cloud with following arguments. First, it has been observed a remarkable trend for the power industry to take advantage of cloud computing to cope with the soaring volumes of data generated and the responding rapidly growing demand for computing capability. Using the service of Cloudera, the cloud computing company, TVA began to manage the large volume of data collected from 103 PMUs (as of 2009), for better reliability and scalability Bisciglia (2009). Additionally, ISO New England started to explore leveraging cloud computing to conduct dynamic simulations not only faster but also less expensive ISO New England (2014); Litvinov (2014). Second, the cloud has abundant computational resources so that it can handle the substantial computation incurred by the cryptographic operations in the privacy-preserving scheme within an acceptable latency. Finally, by letting the control room hold only the decryption key, but not the encrypted measurement data, which is only accessible to the cloud, the system model prevents the control room from abusing the privilege of decryption.

4.4.1.2 Trust Model and Security Goals

In the above system model, the public cloud servers are modeled as semi-honest. It means that the cloud is legally obligated to follow the pre-defined protocol specifications strictly and to conduct the computation correctly, but it will inevitably look at the information sent in and out.

It is also desirable to restrict the access privilege of the control room in the frequency domain. For the sole purpose of monitoring inter-area oscillations, the control room needs only the spectral estimation on a certain frequency range say $[0.1, 0.7]$ Hz, as inter-area oscillations happen in this range Kundur et al. (1994). The rest frequency ranges, especially, higher frequency components are more related to the local events and, thus, are supposed to be protected and released to only the concerned parties.

Our scheme also assumes that the cloud and the control room does not collude.

With the assumptions as mentioned above, the security goal of our system is two-fold:

1. to ensure none of the data sent by individual utility will be disclosed to any parties other than the data owner itself.

2. only the spectrum on the authorized frequency range is disclosed to the control room.

4.4.2 Monitoring Inter-area Oscillations

Our proposed scheme adopts the framework of Sun et al. (2012) and consists of two phases, namely, offline analysis and online monitoring.

The offline analysis is similar to that of Sun et al. (2012). This stage aims at simplifying the power system model by aggregating generators into clusters that may oscillate against each other. The aggregation can be done with generator coherency identification methods such as 1) Tolerance base method and 2) Weak Links Based

Method, according to Sun et al. (2012). The offline analysis does not require sensitive real-time measurements and, thus, can be conducted beforehand without privacy concerns.

The online monitoring phase estimates the spectrum of each cluster's angle. To this end, it is assumed that each cluster has several PMUs installed to monitor the rotor angles of the generators in the cluster. A cluster's angle is given by the average of the angles reported by PMUs belonging to that cluster.

To be more specific, we use δ_{C1} through δ_{CK} to denote the averaged rotor angles of Custer 1 through K; we also use δ_{Cki} to denote the rotor angles reported by the i-th PMU in the k-th cluster. Thus, $\delta_{Ck} = \frac{1}{|Ck|}\sum_{i=1}^{|Ck|}\delta_{Cki}$, where $|Ck|$ is the number of PMUs in Cluster k. Without having any security considerations, the control room can get the spectral estimation of the angle of cluster k at time t, denoted by $F_{\delta_{Ck}}(k_\omega)$, for $k_\omega = 0, 1, 2, ..., N$. by the following procedures. 1) first having all the PMUs in the k report their measurements of rotor angles denoted by δ_{Cki} for $\{i = 1, 2, ..., |Ck|\}$, 2) compute the clusters' angle as $\delta_{Ck} = \frac{1}{|Ck|}\sum_{i=1}^{|Ck|}\delta_{Cki}$, 3) applying Fourier transform to the mean centering cluster angle $\tilde{\delta}_{Ck} = \delta_{Ck} - \langle\delta_{Ck}\rangle$, where $\langle\delta_{Ck}\rangle$ is the iterative mean of δ_{Ck}. $\langle\delta_{Ck}\rangle$ is updated as $\langle\delta_{Ck}(t)\rangle = \frac{t-1}{t}\langle\delta_{Ck}(t-1)\rangle + \frac{1}{t}\delta_{Ck}(t)$, as the time advances.

In what follows, we present the proposed privacy preserving multiparty spectrum estimation that allows for secure calculation of $F_{\delta_{Ck}}(k_\omega)$ without the knowing the values of δ_{Cki}.

4.4.3 Privacy Preserving Multiparty Spectral Estimation

The basic idea is to secure each procedure of the above non-privacy-preserving approach using the homomorphic encryption.

Setup Once the online monitoring is activated, all the concerned parties make agreements on T, the length of the time window and, N, the number of points of the DFT ($T = N$ assumed), f_{PMU}, PMU's sampling rate, f_{spec}, the frequency to conduct spectral estimation, $[f_0, f_1]$, the frequency range authorized to the operator.

Upon successful agreement, the control room is notified to invoke procedures of cloud-assisted multiparty privacy-preserving spectrum estimation.

Key Generation The operator at the control room initializes the Paillier cryptosystem as described in Section 4.3.2. The operator keeps the private key to itself and assigns the public key to the cloud as well as all the participating utilities. In reality, a utility may have more than one PMUs contributing. However, for ease of presentation, we assume that each utility have only one PMU taking part in the multiparty spectral estimation.

Uploading Encrypted Input and Random Masks For every $\frac{1}{f_{spec}}$ second, each PMU encrypts their new measurement data using the public key. For example, the i-th utility in the Cluster k encrypts its measurements $\delta_{\mathbf{Cki}}$ as $E(\delta_{\mathbf{Cki}}) = \{E(\delta_{Cki}(t)\}$ for $i = 0, 1, 2, ..., T-1$, where $E(\cdot)$ is a shorthand of $E(K_{pub}, \cdot)$ hereafter, if there is no confusion.

Besides protecting the privacy in the input data, the proposed scheme also uses random masks to realize the access control in the frequency domain. Suppose according to the agreement in the setup phase, the $\{k_{\omega 1}, k_{\omega 2}, ..., k_{\omega n}\}$-th frequency coefficients are the frequency coefficients corresponding to $[f_0, f_1]$. For every frequency coefficient, each utility (PMU) generates a random complex number to obscure the coefficient jointly. For example, Utility A generates a mask array r_A where $r_A(k_\omega) = 0$ if $k \in \{k_{\omega 1}, k_{\omega 2}, ..., k_{\omega n}\}$, and $r_A(k_\omega)$ is a random value if otherwise. All utilities involved with cluster k generate and encrypt their random masks using Algorithm 4. The utilities then send the encrypted random masks along with the encrypted measurement data to the cloud in order to let the cloud obscure the spectrum in the way described in Algorithm 2.

Privacy Preserving Spectrum Estimation Now using the data received from PMUs, the cloud securely estimates the latest spectrum by applying privacy-preserving Fourier transform to the data of the most recent time window for each cluster. Algorithm 2 details the privacy-preserving Fourier transform. The algorithm first securely derives the averaged angle of a cluster in the time window. It then

securely calculates the corresponding iterative mean of the cluster's angle and mean centering cluster angle. Finally, the algorithm applies privacy-preserving discrete Fourier transform to the mean centering cluster angle gives the encrypted spectral estimation. Algorithm 2. calls three supporting sub-algorithm *i.e.* Algorithm 3, 4, and 5. Since all computations are carried out in the encrypted domain, so no information about the measurement data is leaked to the cloud.

Result Decryption The operator at control room receives encrypted frequency coefficients as a result of Algorithm 2 from the cloud. It then decrypts for the frequency coefficients using decryption method of Paillier cryptosystem with the private key. Upon decrypting all coefficients corresponding to $[f_{O1}, f_{O2}]$, the control room obtains the spectrum estimation of the authorized frequency range of the latest time window. The spectrum estimation is then used for visualization, monitoring, or other frequency domain approaches. For unauthorized frequency range outside $[f_{O1}, f_{O2}]$, the decryption results are frequency coefficients masked with random values, *i.e.* $S_{Ck}(k_\omega) = F_{Ck}(k_\omega) + r_{Ck}(k_\omega)$, where $r_{Ck} = \sum_{i=1}^{|Ck|} r_{Cki}$ is sum of random masks of all concerned utilities in Cluster k.

Here the underlying access control policy means that only frequency coefficients corresponding to $[f_{O1}, f_{O2}]$ is accessible to the control room, while all other obscured coefficients are useless without knowing all the added random masks, so that no party may access them. Nevertheless, this is only one of many access control policies that can be applied. In other cases, if the spectrum of a particular frequency range outside $[f_{O1}, f_{O2}]$ contains useful information to a utility, say Utility A, the proposed scheme enables disclosing that part of the spectrum to A. To do so, during the setup phase, Utility A claims the frequency range, if agreed, other utilities just turn off their respective random mask on that range such that $S_{Ck}(k_\omega) = F_{Ck}(k_\omega) + r_A(k_\omega)$ holds for the claimed frequency range. Consequently, after receiving from the control room the decrypted but still obscured $S_{Ck}(k_\omega)$, A can learn the real $F_{Ck}(k_\omega)$ over the claimed frequency range.

Data: $E(\delta_{Cki})$, $E(r_i(k_\omega))$ for $i = 1, 2, ..., |Ck|$, $t = 0, 1, 2, ..., T-1$,
$k_\omega \in \{0, 1, 2, ..., N-1\}$;
T Length of the time window, $N = T$;
$|Ck|$ Number of generator in the cluster;
τ the running counter of measurements in all past spectral estimations;
Result: Encrypted and masked frequency coefficients $E(S_{Ck})$ where
$S_{Ck}(k_\omega) = F_{Ck}(k_\omega) + r(k_\omega)$
Initialize **Spectrum** an zero-initialized vector of length N;
/* Homomorphically compute the averaged rotor angle of cluster k */
for $t = 0$ **to** $T-1$ **do**
  $E(\sum_{i=1}^{|Ck|} \delta_{Cki}(t)) = \prod_{i=1}^{|Ck|} E(\delta_{Cki}(t))$;
  $E(\delta_{Ck}(t)) = E(\frac{1}{|Ck|} * \sum_{i=1}^{Ck} \delta_{Cki}(t))$
end
/* Obtain the mean centering cluster angles */
for $t = 0$ **to** $T-1$ **do**
  /* Calculate the iterative means of δ_{Ck} within the past second. Note that $\frac{\tau+t-1}{t}$ and $\frac{1}{\tau}$ are constant and can be multiplied with ciphertexts */
  $E(\langle\delta_{Ck}(\tau+t)\rangle) = E(\frac{\tau+t-1}{t}\langle\delta_{Ck}(\tau+t-1)\rangle + \frac{1}{\tau}\delta_{Ck}(t))$; /* using Algorithm 3 and 5 */
  /* Get mean center δ_{Ck} */
  $E(\tilde{\delta}_{Ck}(t)) = E(\delta_{Ck}(t) - \langle\delta_{Ck}(t)\rangle)$; /* using Algorithm 3 */
end
/* Securely apply N-point DFT to $\tilde{\delta}_{Ck}$ */
for $k_\omega = 0$ **to** $N-1$ **do**
  /* Homomorphically sum up the encrypted random masks from all PMUs in cluster k */
  $E(r_{Ck}) = E(\sum_{i=1}^{|Ck|} r_{Cki})$; /* using Algorithm 3 */
  Spectrum$[k_\omega] \leftarrow E(r_{Ck})$;
  for $t = 0$ **to** $T-1$ **do**
    /* Calculate the respective element in the DFT matrix */
    $W_{tk_\omega} = W^{tk_\omega} = e^{-\frac{j2k_\omega t\pi}{M}}$;
    Securely calculate $E(\tilde{\delta}_{Ck}(t) * W_{tk_\omega})$; /* using Algorithm 5 */
    /* Securely accumulate the result to **Spectrum**$[k_\omega]$ */
    Spectrum$[k_\omega]$ = **Spectrum**$[k_\omega] \cdot E(\tilde{\delta}_{Ck}(t) * W_{tk_\omega})$
  end
end
return Spectrum

Algorithm 2: Privacy preserving spectral estimation of the angle of a generator cluster

Data: Encrypted complex numbers $(E(A_{real}), E(A_{img}))$ and $(E(B_{real}), E(B_{img}))$
Result: Encrypted sum or difference of A and B, $(E(A_{real} \pm B_{real}), E(A_{img} \pm B_{img})$
if *to return the addition* **then**
| **return** $(E(A_{real})E(B_{real}), E(A_{img})E(B_{img})$
end
else if *to return difference* **then**
| **return** $(E(A_{real})(E(B_{real})^{-1} \mod N), E(A_{img})(E(B_{img})^{-1} \mod N))$
end

Algorithm 3: Secure addition and subtraction of complex numbers

Data: a complex number $a + bi$;
Public key of the Paillier cryptosystem (g, N)
Result: Encryption of a complex number
`/* Use the encryption method in Paillier cryptosystem to encrypt the real and imaginary part, respectively */`
$E(a) \leftarrow g^a r_R^N \mod N$;
$E(b) \leftarrow g^b r_I^N \mod N$;
return $(E(a), E(b))$

Algorithm 4: Encryption of a complex number

Data: An encrypted complex number $(E(a), E(b))$;
An constant complex number $c + di$;
Result: Encryption of the complex number $(a + bi)(b + ci)$
$encProd_R = E(a)^b (E(b)^c)^{-1} \mod N^2)$; `/* get the real part */`
$encProd_I = E(a)^c (E(b)^d \mod N^2)$; `/* get the imaginary part */`
return $(encProd_R, encProd_I)$

Algorithm 5: Multiply a encrypted complex number with a constant complex number

4.4.4 Optimization

Algorithm 1. can be optimized in following ways to improve its efficiency.

Precompute DFT matrix Once parameters T and N are chosen, the DFT matrix, *i.e.* W_{tk_ω} for $t \in 1, 2, ..., T$ and $k_\omega \in 1, 2, ..., N$ are also fixed. Therefore, instead of computing the elements in the DFT matrix on the fly, precomputing the DFT matrix before the runtime and building a lookup table can reduce the execution time of the algorithm.

Compute only selected frequency coefficients To monitor inter-area oscillation, it suffices to perform the spectral estimation over only the frequency range of $[0.1, 0.7]$ Hz. Likewise, to monitor low-frequency oscillation, it suffices to only estimate the spectrum over the frequency range of $[0.1, 2]$ Hz. Computing only the necessary frequency coefficients, instead of the whole spectrum, will save computation efforts and reduce execution latency.

Parallelizing The encrypted frequency coefficients are independently calculated in Algorithm 2. As a result, it is possible to parallelize the computation by having a number of machines independently computing subsets of the frequency coefficients at the same time. For example, if ten machines are employed, each machine can compute tenth of the all encrypted coefficients In fact, with the good scalability, the cloud is very suitable and convenient to set up and carry out such parallel computations.

In Section 4.7, experiment results show that the optimization techniques improve the performance of the algorithm by orders of magnitude.

4.5 Security Analysis

We analyze the security of the proposed scheme in a bottom-up approach. To this end, we first analyze the security of Algorithm 4, Algorithm 3, and Algorithm 5, and, lastly, Algorithm 2.

Security of Algorithm 4 Algorithm 4 is semantically secure as long as the underlying Paillier cryptosystem is secure under decisional composite residuosity assumption Paillier (1999), as this algorithm only encrypts two independent integers.

Security of Algorithm 3 Algorithm 3 is semantically secure guaranteed by the security of additive homomorphic property of Paillier cryptosystem.

Security of Algorithm 5 Algorithm 5 is semantically secure as relies on the Paillier cryptosystem's homomorphic multiplication between a ciphertext and plaintext, which is known to be secure as well as Algorithm 3, which is also semantically secure.

Security of Algorithm 2 Algorithm 2 is semantically secure, because the algorithm only calls Algorithm 5 and Algorithm 3 and does not include other computations. Both the called algorithm are semantically secure by our analysis.

By now, the security goals of the scheme are met. First, for any participating utility company, its peer utility companies cannot get the original measurement data it shares for the monitoring purpose, because the measurement data are sent to the cloud while they are encrypted with the public key. Second, the cloud will learn neither the value of measurement data, any intermediate result nor the final spectrum because all computations are carried out in the encrypted domain. Finally, the access control in the frequency domain is effective because unauthorized frequency coefficients are securely obscured, so the control center can only access the spectrum of the authorized frequency ranges.

4.6 Complexity Analysis

For ease of presentation, we use notations Table 4.1 in the analyses.

4.6.1 Complexity Analysis

In this section, we analyze the computation and communication complexities of the proposed privacy-preserving scheme. Following a bottom-up approach, we

Table 4.1: Notations used in the the complexity analysis

b_c	Bit length of ciphertexts
f_{PMU}	Sampling rate of PMUs
f_{spec}	Frequency of conducting spectrum analysis
T	length of a time window (number of frequency coefficients)
N	length of DFT
$n_{cluster}$	number of generator clusters
n_{PMU}	average number of PMUs of a cluster
MUL	runtime for one multiplication on group G
EXP	runtime for one exponentiation on group G
D	runtime of decryption for a number in the Paillier cryptosystem.

first analyze the computation and communication overhead of the component sub-algorithms (Algorithm 2. through Algorithm 4), and then those of the main Algorithm. 2. As we assume that the execution time to perform modular multiplications and exponentiations on group G and the time required for decryption are the primary computation cost, we express the algorithms' computation overheads as functions of MUL and EXP.

4.6.1.1 Execution Time of Sub-algorithms

It is easy to get that Algorithm 3 takes $\mathbb{T}_3 = 2MUL$, since it requires one addition for the real part and another one for the imaginary part.

Similarly, Algorithm 5 requires three exponentiations and one multiplication to obtain the real part and two exponentiations and one multiplication to get the imaginary part. Thereby, Algorithm 5 take $\mathbb{T}_5 = 2MUL + 5EXP$ in total.

Algorithm 4 takes one modular exponentiations and one modular multiplication to encrypt the real part or the imaginary part of the complex number, so its execution time is given by $\mathbb{T}_4 = 2(EXP + MUL)$.

4.6.1.2 Execution Time for One Round of Spectral Estimation

Each of the $n_cluster$ clusters requires a spectral estimation, which takes T samples of the mean-centering cluster angle, and calculates N frequency coefficients. Assume

spectrum estimations are performed at the frequency of f_{spec} Hz. So, there are $\frac{f_{PMU}}{h_{spec}}$ new measurements sent from each PMU for one round of spectral estimation.

For a PMU, all it needs to do is to encrypt all the measurement collected during the last round of spectral estimation. Therefore, the total execution time for a PMU is $\frac{f_{PMU}}{h_{spec}}\mathbb{T}_4$.

The control room needs to decrypt for the frequency coefficients of $n_{cluster}$ clusters and each contains N complex numbers. As a result, the total running time for the control room is $2n_{cluster}ND$.

The most computation required by Algorithm 2 is migrated to the cloud servers. For each cluster, the algorithm first derives the averaged cluster angle, which takes $T \cdot n_{PMU}\mathbb{T}_3 + \mathbb{T}_5$. Calculating the iterative means takes $T(2\mathbb{T}_5 + \mathbb{T}_3)$. Calculating the mean-centering cluster angle takes $T\mathbb{T}_3$. The algorithm then performs privacy-preserving N point Fourier transform over the obtained mean centering cluster angle. To calculate one frequency coefficient, the algorithm calls Algorithm 5 T times, Algorithm 3 T times. Hence, in total, the overall execution time to conduct spectral estimations for $n_{cluster}$ clusters using Algorithm 2 is given by $\mathbb{T}_2 = n_{cluster}T((n_{PMU} + N + T + 1)\mathbb{T}_3 + (3 + N)\mathbb{T}_5))$

4.6.1.3 Communication Overhead

For one round of spectrum analysis, the communication overhead for each PMU is the encrypted measurement data from the last spectrum analysis, which is given by $2b_c\frac{f_{PMU}}{h_{spec}}$ bits.

Cloud servers receive all bits from all the PMUs, and it also sends the encrypted frequency coefficients to the control room. So the communication overhead for the cloud is $2b_c\frac{f_{PMU}}{h_{spec}} * n_{PMU} + 2n_{cluster}Nb_c$ bits.

The control center receives the encrypted frequency coefficients of the $n_{cluster}$ clusters, which takes $2n_{cluster}Nb_c$ bits.

4.7 Performance Evaluations

4.7.1 Stress Testing

4.7.1.1 Set Up

We fully implemented the scheme with Java's BigInteger library and measured the run time to perform one round of privacy preserving spectral estimation. We use the key length of 512 bits to initialize the Paillier cryptosystem. All experiments are conducted on Linux servers with 8-core Intel Core i7 processor and 16GB RAM.

In the stress testing, we test the latency of Algorithm 2 as we gradually increase the length of the time window. PMU measurements are simulated by random generated time series of rotor angle measurements at a frequency of 50 Hz. Each measurement is a decimal number with six decimal places. Because Paillier cryptosystem would work on Z_N, the raw measurement are mapped to Z_N with all the decimal places preserved by multiplying a scaling factor $Q = 1,000,000$,

We measured the execution times of Algorithm 1 before and after the optimization. The optimization being applied are as follows:

1 precompute the DFT matrix and build an in-memory lookup table before runtime.

2 compute only the frequency coefficients corresponding to the $[0, 1]$ Hz.

3 parallelize the computation with five servers in the same computer cluster.

Without optimization, Algorithm 1 calculates the whole spectrum and is executed on a single machine with the hardware specifications mentioned above.

4.7.1.2 Experiment Result

As Table.4.2 shows, the applied optimization reduces the computation overhead by three orders of magnitude and reduce the communication overhead by one order of

Table 4.2: Time for execution of Algorithm 1 before and after optimizations

Time window	Algorithm 1		Algorithm 1 optimized	
	Comm.(kb)	Comp.(seconds)	Comm.(kb)	Comp.(seconds)
500	75	50.5	13.125	0.06
1000	137.5	209.6	13.75	0.24
1500	200	460.5	14.375	0.53
2000	262.5	816.8	15	0.93
2500	325	1280	15.625	1.44
3000	387.5	1858.2	16.25	2.06
3500	450	2507.4	16.875	2.79
4000	512.5	3305.6	17.5	3.74

magnitude. The communication overhead after optimization grows much slower than that of the original algorithm. Additionally, the latency can be further reduced by using more machines working in parallel. In fact, as long as the latency is less than the period of conducting spectral estimations, the proposed scheme can be used for online monitoring.

4.7.2 Test with WECC 179-bus Power System

We also tested the proposed scheme with a simplified model of WECC AC transmission system with 29 generators and 179 buses Sun et al. (2012). As illustrated in Figure 4.3, offline analysis indicates that generators in the transmission grid is split into four coherent clusters. Each cluster is monitored by three PMUs, which are circled in the figure. The sampling rate of the PMUs is 100 Hz. Every second, privacy-preserving spectral estimation is applied to update the power spectrum of the cluster angles over the latest 30 second window (containing 3,000 measurements), with a 29-second overlap with the last time window. As we are monitoring inter-area oscillations, only coefficients corresponding to $[0.1, 0.9]$ Hz, *i.e.* , the 3rd through 21th frequency coefficients, are calculated.

For demonstration purpose, four successive three-phase faults is simulated at 40s, 80s, 120s, and 160s.

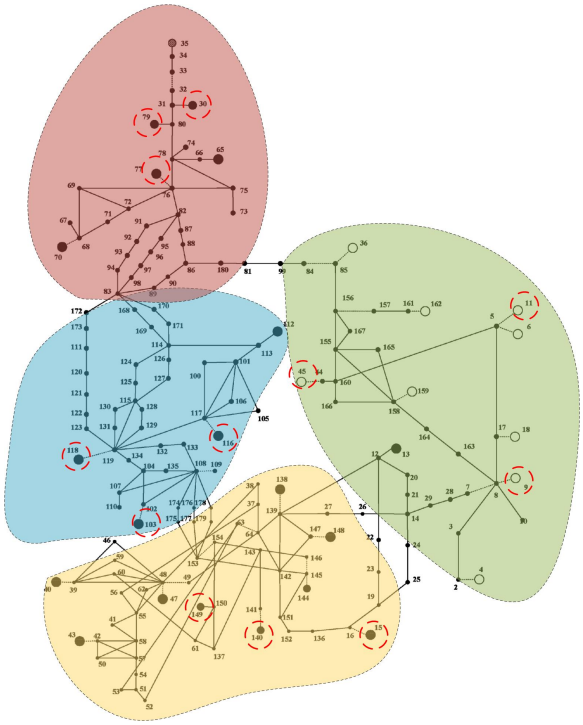


Figure 4.3: Coherent generator clusters in WECC 179 bus system

With the proposed scheme, the control room can obtain the spectral estimations of four clusters' averaged rotor angles without the need for the original PMU measurements. With 15 servers working in parallel, the averaged latency to obtain spectral estimations for all the four clusters is 0.95 second, which is less than the period of conducting spectral estimations (1 second).

The encrypted and decrypted spectral estimations over the simulated time are visualized using spectrograms in Figure. 4.4 and Figure.4.5, respectively. As seen in the spectrograms, frequency responses emerge and die down as the faults are applied and cleared. Obvious oscillation modes are seen around 0.2Hz, 0.5Hz, and 0.75 Hz, with the 0.2 Hz mode being the dominant one. The results of spectral estimations are in accordance with the result of small-signal analysis. The obtained spectral estimations can also be used for further research and analysis, which is beyond the scope of this work.

Figure. 4.6 further illustrates the access control in the frequency domain. The left panel of the figure depicts the full spectrogram of the angle of Cluster 2, while the right panel captures the same spectrogram but with its $[0.6, 0.9]$ Hz range obscured with a random mask by one of the utility companies in Cluster 2.

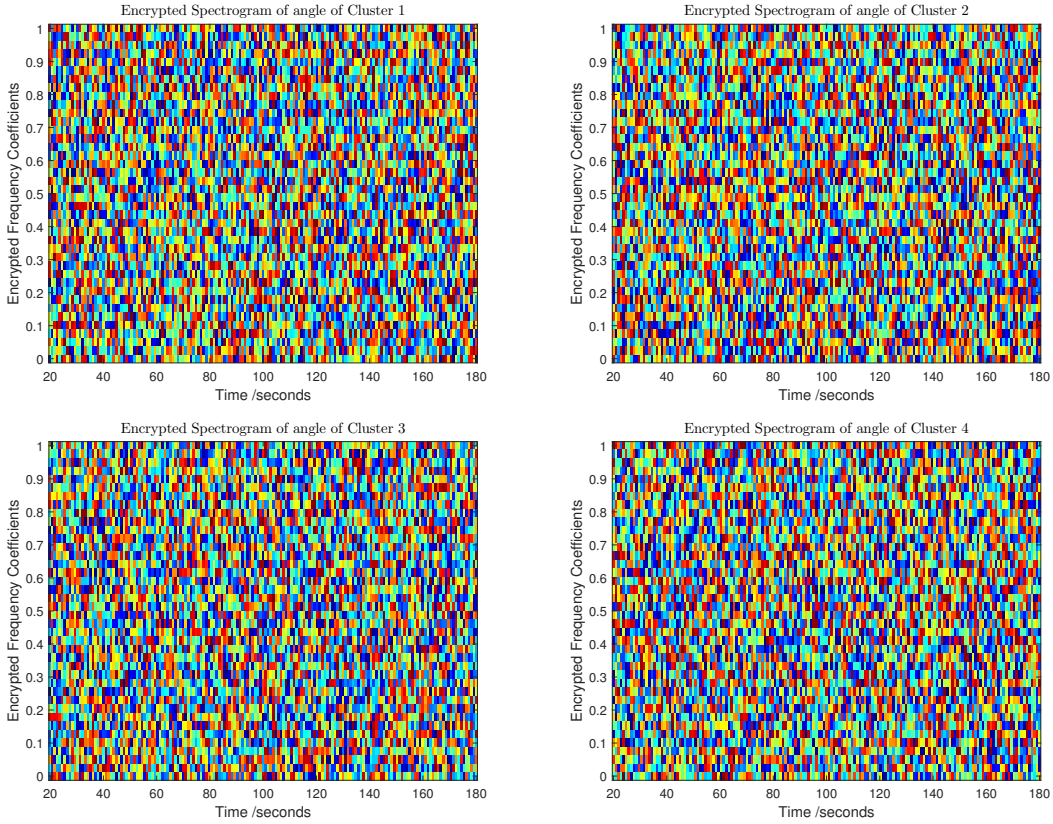


Figure 4.4: Encrypted spectrograms of clusters' angle

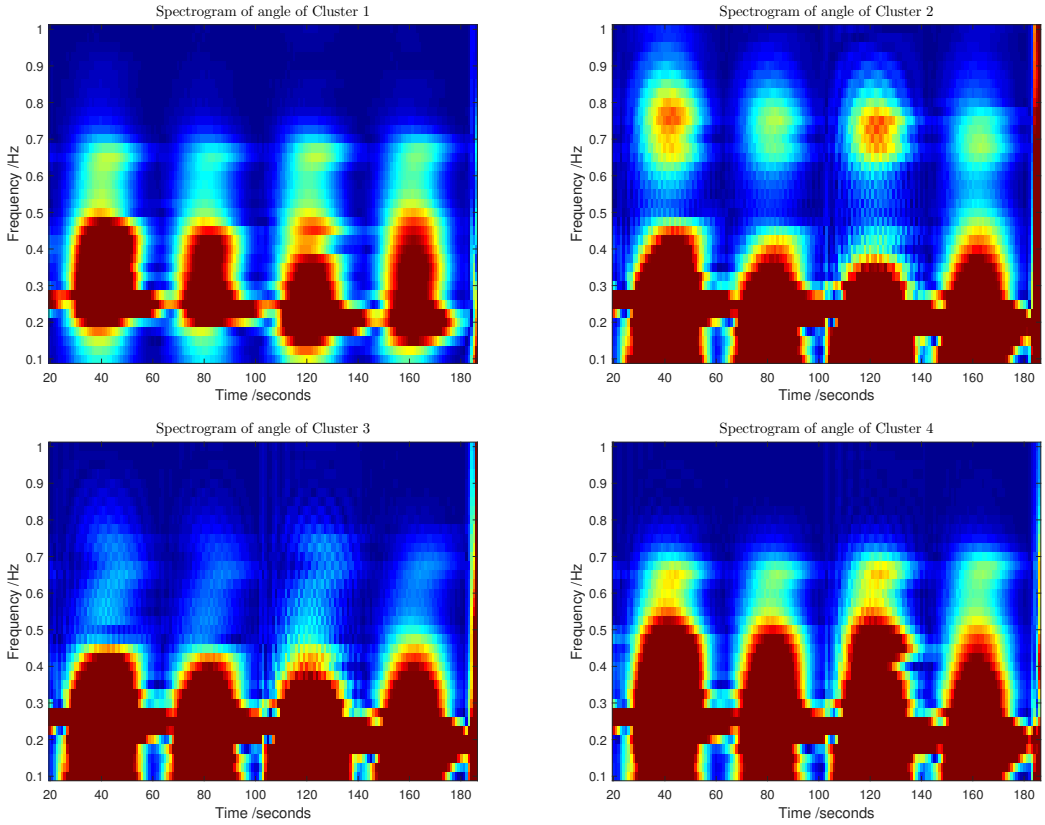


Figure 4.5: Decrypted spectrograms of clusters' angle

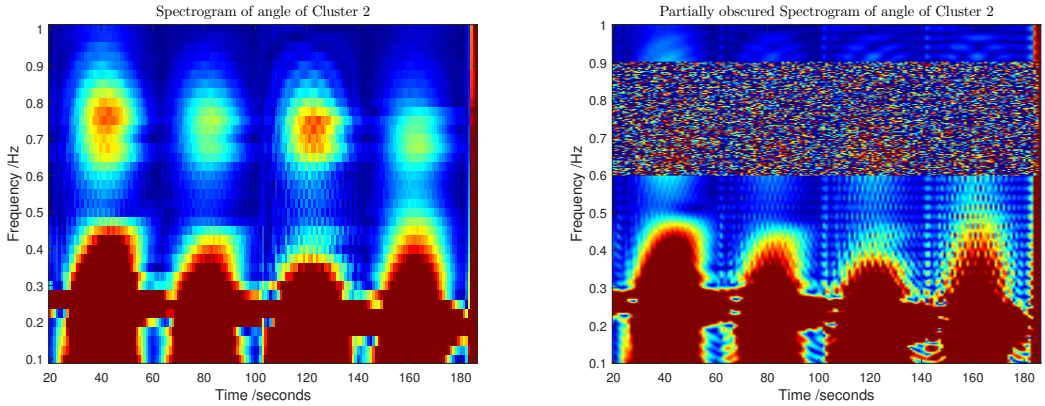


Figure 4.6: Decrypted spectrograms of clusters' angle

Chapter 5

Secure Data Sharing for Situational Awareness

5.1 Introduction

Power grids are critical infrastructure, whose reliability plays a significant role in people's everyday life and work. For instance, the 2003 Northeast blackout affected 55 million people in eight states in the U.S. and one province in Canada, led to 256 power plants going offline and $6 billion of economic loss Electricity Consumers Resource Council (2003). Among the many ways to improve the reliability of the power grid, situational awareness emerges as a promising and efficient high-level solution. According to EIOC (2013), *"The analyses of recent blackouts have demonstrated that a major challenge to keeping the lights on is maintaining situational awareness"*. Situational awareness is the practice to capture the changes in the environment or the subject. It renders the information to human operators for their perception and comprehension such that they will take the best actions in response to the changes EIOC (2013). In power system operations, system faults, inclement environment conditions, or inside/outside agent attackers can all cause the same grave consequence — a widespread blackout. In such a case, global situational awareness

is the starting point for any subsequent measures to adequately mitigate system operation risks, to prevent the spread of malfunction and facilitate fast recovery.

One would imagine that the difficulty to maintain situational awareness stems from the need to process large amount of data, or from the demand for effective visualizations to exact and render the information hidden in the data. However, an even more fundamental reason comes from the unavailability of the data, which must be collected from multiple utility companies that are potentially competing and mutually untrusted. For example, the North American power grid's high voltage transmission system has more than 80,000 nodes (i.e., busbars) at 69kV. Those nodes include both substations and power plants. They collectively exist within a hierarchy consisting of more than 2,000 independent utility companies, over 100 balancing authorities and eight reliability councils. As pointed out by EIOC (2013), "*Since deregulation, a competitive environment has arisen and complete information about the state of the power grid is seldom available to operators*". For this reason, to truly enable *global* situational awareness, an efficient data sharing mechanism that also preserves the concerned parties' privacy is in urgent need.

The challenge to design such a mechanism is threefold. First, the scheme must guarantee the data privacy of all the participating parties. Second, the mechanism should attend to the sophisticated structure of the data to be shared. For one thing, the volume of data is steadily increasing as the time goes by. For the other, the data required by situational awareness comes in a wide variety. Third, the data sharing mechanism should be able to adapt to different situations during the operation of the power grid. More specifically, during normal operations, where data sharing activities are among different utility companies, privacy preservation is preferred over performance. As a contrast, when an emergency takes place, the data sharing is conducted between Independent System Operation (ISO) and affected utility companies to facilitate real-time situational aware. In such cases, performance, in terms of low latency and high throughput, becomes the top concern. An ideal design should satisfy all the three requirements simultaneously.

In this chapter, we present a novel secure power flow data sharing scheme for practical power grid situational awareness.

5.2 Related Works

Klump and Weber (2002) indicates that data sharing within the same utility company is widely adopted by using standard databases with APIs for remote access. Additionally, that real-time monitoring of the most recent snapshot with 3 to 5 seconds of latency is typical. However, there are strong reservations regarding sharing operation data across utility companies because of privacy concerns.

Although, to our best knowledge, we are the first to investigate the secure data sharing in the context of the smart grid, we draw significant inspirations from previous works of cyber security.

The cybersecurity community has studied the problem of secure storage and keyword search Sun et al. (2011, 2010); Goh (2003); Curtmola et al. (2006); Liesdonk et al. (2010); Wang et al. (2012b, 2011a, 2012a,c); Cao et al. (2011b,a); Wang et al. (2011c); Li et al. (2010a,b). However, a fundamental difference between these works and our proposed schemes is that existing works do not handle large volumes of data in real time. As a result, the computational efficiency is not a primary concern for them, and, thus, their schemes is inefficient to meet the stringent time constraints required for the time-critical situational awareness.

This work utilizes the evolving key scheme for key management. Evolving key schemes were previously proposed by Tzeng and Tzeng (2002); Itkis and Xie (2003); Franklin (2006) to primarily facilitate public key based encryption and signature, and offer forward secrecy and intrusion resilience and However, the evolving key scheme proposed in this work is unique in the sense that the hierarchical structure of the key generation and management is derived from the unique model of the power flow data being investigated.

Finally, various security issues in the power grid by Qi et al. (2011); Bobba et al. (2009); Khurana et al. (2010a); Rogers et al. (2010); Zhang and Gunter (2010); Wang et al. (2010b); Kosut et al. (2010); Bakken et al. (2007), but none considered the challenging issues around secure data sharing.

5.3 System Model

5.3.1 Network Model

In the network model of the proposed scheme, grid operators of participating utility companies store their real-time power flow data in a data center with massive storage that is hosted either by the regional ISO or a trusted authority. The data would be used by the regional ISO or other operators in response to critical events. The centralized approach facilitates *in-situ* data processing to eliminate latency incurred by transferring data to the ISO from multiple utility companies during an emergency, which could be invaluable for quick remedial measures. However, the data stored in the data center must be protected such that the data owners, instead of the data center, have the ultimate access control of its data, such that the data owner can specify a particular piece of data to be accessed by a certain party. We will elaborate on this in Section 5.3.3.

5.3.2 Data Model

We derived the data model from the power flow data and is designed to facilitate data visualizations for situational awareness, which differentiates itself from previous works designed for general data models. Figure 5.1 captures the model of the power flow data. In the figure, $T_t S^m$ denotes the different time instants (indexed by m) within the time period $(T_{t-1}, T_t]$, or simply $(t-1, t]$, and t is the ending time of a period, *e.g.* , 20:00:00 11/11/11. We model the data with three dimensions, namely, data category, time scale, and spatial scale. Data category indicates the type of collected

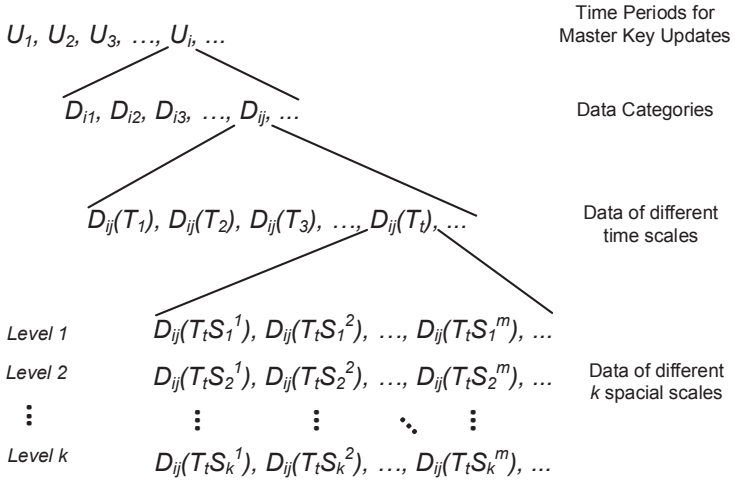


Figure 5.1: Data Model.

power flow data, such as overload index (*e.g.* , instantaneous voltage, current, power), voltage stability, frequency stability, and transient stability. Time scale represents data collected in different time periods. Spatial scale indicates the location where the data is collected. The data model is of a hierarchical structure. The entirety of the data is first divided into slices according to their categories. Each slice is further divided into smaller slices based on the time the data is collected. Finally, each smaller slice contains all the data of the category, obtained at the respective period at different spatial scales.

In Figure. 5.1, consider the category D_{ij}, say, the overload index, collected during the master key update period i. It follows that $D_{ij}(T_t)$ denotes the overload index values collected in time period t. $D_{ij}(T_t)$ can be expressed by a table, where each row corresponds to a spatial scale, and an element in a row indicates the overload index value at a particular spatial scale at a specific time instance during time period t. Data on a same spacial scale within a same time period, *e.g.* , $D_{ijk}(TS_1^t), \cdots, D_{ijk}(TS_m^t), \cdots$, are stored in a data file.

5.3.3 Threat Model

In the threat model, the ISO or the third trusted party, who administrates the data center, are fully trusted. ISO only requests the concerning utility companies for accesses to their data when situational awareness is needed for critical events.

Adversaries considered by this work include peer grid operators and attackers the system. Peer operators are assumed to be honest-but-curious in the sense that once they are authorized to obtain each other's data, they will only use the data for legitimate purposes and will not leak the data to unauthorized parties or outside attackers. However, they will be curious to learn each other's sensitive data for their own advantage, *e.g.* , economic data for marketing, if there is chance that the data is unprotected. Outside attackers are malicious as they try to gain access to operators' data by launching both passive (*e.g.* , eavesdropping) and active (*e.g.* , hacking into the data center) attacks against the data center.

5.3.4 Security Goals

Given the application scenario we are concerned with, our system strives to achieve three security goals: data privacy, access control, and integrity. *Data privacy* has two further aspects: *data confidentiality* and *unlinkability*. *Data confidentiality* requires that unauthorized entities (*e.g.* , peer utility companies, outside attackers) must not learn the data stored in the data center. *Unlinkability* dictates that unauthorized entities must not be able to link multiple data files to profile the operator, which requires that the data should be stored scrambled, appear random, and leak no useful information. *Access control* requires that the operator has control over who may access which portions of the data. The desired access control should support fine-grained access control, meaning that the data owner are enabled to control the access of the smallest data unit, *i.e.* , a single data file, in this case. Additionally, it should also be efficiently adjustable. Specifically, it requires that if the data owner is to grant access to a large set of files, the data owner does not have to permit access to

every single file, which is time-consuming and inefficient. Last but not least, *Data integrity* requires that the operators' data stored in the data center should remain intact, and any tampering with the stored data should be detected. This requirement is particularly crucial for situational awareness since falsified data may cause wrong decision-making and devastating consequences.

5.4 S^3A Security Architecture

5.4.1 Scheme Overview

The data sharing mechanism for situational awareness consists of two components: *secure and unlinkable data storage* and *evolving key scheme based key management*.

The *secure and unlinkable data storage* serves as an "encrypted database" that facilitates searching over encrypted data. The *evolving key scheme based key management* is used to achieve an efficient key management tailored for the particular power flow data model and the flexible-grained access control. It is worth noting the two components do not work independently, as later discussions reveal. An overview of the proposed secure data sharing mechanism for situational awareness follows.

Consider a scenario where the ISO, requests the overload index values collected in time period t, denoted by $D_{ij}(T_t)$ by another utility company, called A. The request is first sent to the data center. The data center then sends the keywords "overload index" and "t" to A who owns the data. With the keywords, the operator at A computes two trapdoors, one for each keyword, and returns the trapdoors to the data center. Using Secure Index, with the two trapdoors, the data center can then obtain the file identifiers, which are the pseudonyms for the respective files, and use them to retrieve the encrypted data files.

Since the retrieved data files are encrypted, the data center still needs to know the keys to decrypt the files. The operator at Utility A computes the keys with **evolving key scheme based key management**, when he or she receives the two keywords

above, he or she can derive the decryption key given the keywords using the proposed hierarchical evolving key scheme detailed in Section.5.4.3. Thus, the operator also returns the keys to the data center. In our example, the keys used to decrypt the overload index data files collected in the period t are the encryption keys sitting at the bottom K levels in the key hierarchy. In fact, since all requested files belong to $D_{ij}(T_t)$, the operator needs to return only one key, the time scale key corresponding to $D_{ij}(T_t)$. With the time scale key and (i, j, t, k), which is shown in the figure, ISO itself may derive all the file encryption keys in the lower hierarchy. This approach may be more desirable in situations like a crisis, where transmitting many keys over a potentially unreliable communication network may not be possible. In either case, the ISO now obtains all the data it needs and can perform feature extraction according to the five spatial scales ($K = 5$) for situational awareness using the decrypted power flow data. The method to derive the encryption keys will be covered in Section. 5.4.3.

5.4.2 Secure and Unlinkable Data Storage

The secure and unlinkable data storage ensures the security of the data and privacy of the data owners who stored their data in the remote central data warehouse. It consists of two parts: 1) encrypted files, each of which is identified by a identifier that does not reveal any information about the content of the file, and 2) secure indexes for their respective keywords. Each secure index is associated with a keyword, which enables searching over the encrypted data. A secure index takes the form of a linked list, and a node in the linked list represents a encrypted file that contains that keyword.

The secure indexes, while facilitating searching over encrypted data, should also guarantee data confidentiality and unlinkability as follows: 1) keywords submitted for queries should not be exposed to the data center, as it is also part of the private information, and 2) the linkage from a keyword to the set of files that contain that keyword should also remain unexposed to the data center without a legitimate request

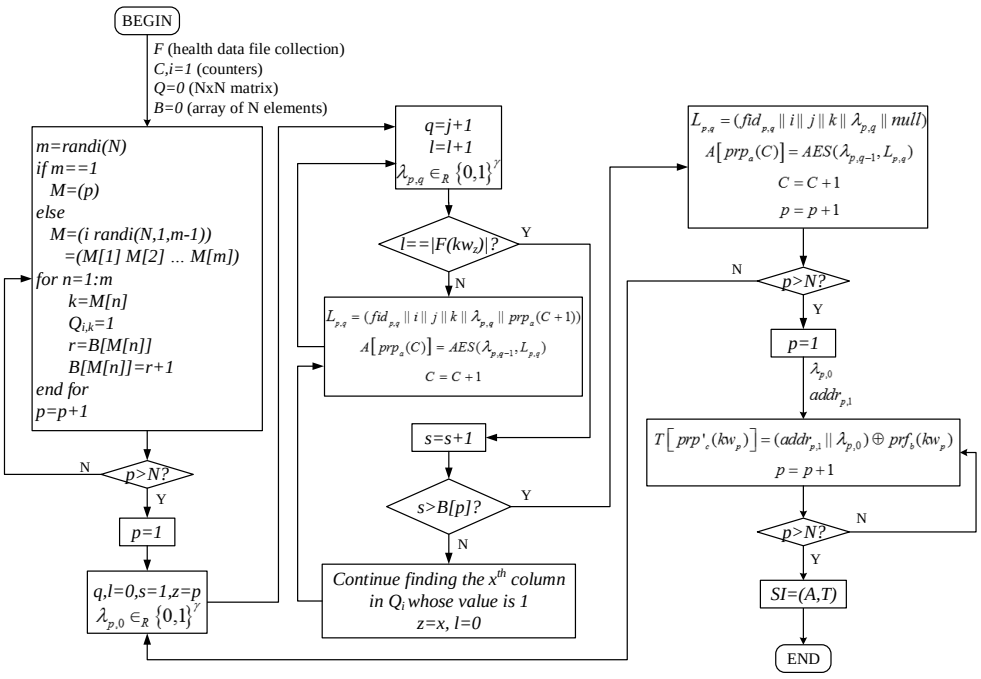


Figure 5.2: Construction of Secure Index

to the data owner. That is, by examining the a secure index, one should not be able to link the encrypted files that belong to that secure index to the respective keyword of the secure index, even though both the keyword and the files are encrypted.

These security features cannot be achieved by only using random numbers as the indexing method. Section 5.4.2.1 presents the details about how to satisfy the security requirement in details.

5.4.2.1 Secure Data Storage

The operator P_n at a particular utility company constructs a secure index for the data files using the algorithm. Figure.5.2 shows the procedures. The input to the algorithm is the plaintext data file collection F and counters C and x (both starting from 1). The notations in Figure.5.2 are defined as follows. SI consists of an array A and a lookup table T. $A[*] = z$ (and similarly $T[*]$) denotes the value z is stored in $A[*]$. The collection of linked lists $L = \{L_x | x = 1, \cdots, |kw|\}$ (where $|kw|$ denotes the size of the

keyword space) is encrypted and stored in A. Further, each linked list L_x is a collection of nodes $L_{x,y}$ such that $L_x = \{L_{x,y}|y = 1, \cdots, |F(kw)|\}$ (where $|F(kw)|$ denotes the number of data files containing kw). Each linked list node contains three fields in Curtmola's construction Curtmola et al. (2006), *i.e.* , $L_{x,y} = (fid_{x,y} \parallel \lambda_{x,y} \parallel ptr)$ where $\parallel$ denotes concatenation and $fid_{x,y}$ is a unique file identifier, $\lambda_{x,y}$ of length ξ is the secret key used to encrypt the next node $L_{x,y+1}$ in the linked list L_x, and ptr contains the address of the encrypted $L_{x,y+1}$ (*i.e.* , $E_{\lambda_{x,y}}(L_{x,y+1})$). The first node $L_{x,1}$ is addressed by $addr_{x,1}$. The pointer ptr is the output of a pseudorandom permutation prp computed from the data owner secret a. Similarly, prp' is another pseudorandom permutation computed from the secret c. The keyword is encrypted by a pseudorandom function prf computed from the secret b. We added a new field $(i \parallel j \parallel t \parallel k)$ to a linked list node for identifying and regenerating the right encryption key. We will provide justification for this design shortly in Section 5.4.2.2.

The operator P_n then encrypts the data files with the encryption keys (δ_k's) generated from the evolving key scheme (Section. 5.4.3) and stores the encryptions in the data center. Fig.5.3 shows an example that includes data files containing different keywords, the construction of their linked lists, a search based on the keyword "transient stability" and the data retrieval.

Finally, a question remains — how often do the operators of different utility companies update their power flow data at the data center. The update frequency has a significant impact on both the efficiency of the data sharing mechanism and the accuracy of the resulted situational awareness. When an operator updates its respective power flow data, the operator sends a new set of data files to be stored in the data center, and the data center needs to construct a new secure index over this set. In general, Δ updates will result in Δ different data files collections and associated secure indexes. As a result, at least Δ trapdoors and Δ searches are needed to retrieve all data files (in different collections) containing a keyword since a trapdoor computed based on one secure index cannot be used to search over other data collections. If the update period is too small, it is prohibitively expensive for

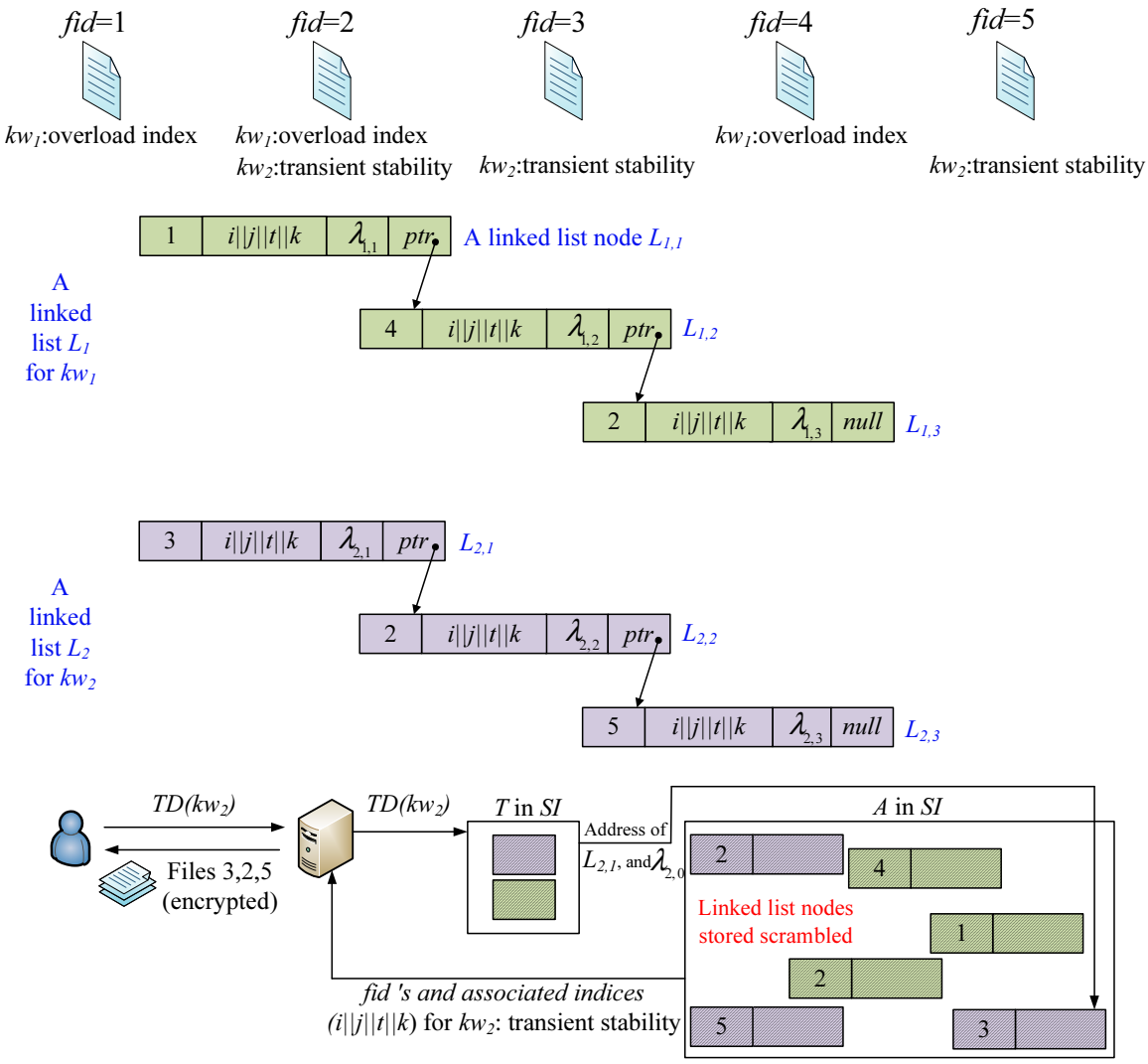


Figure 5.3: An example of the access-pattern-preserving SI construction process.

The construction is for five files involving two keywords. Two linked lists each created for a keyword and a search based on the keyword "transient stability". T in SI is used to find the address of the first linked list node $L_{2,1}$ stored in A. The symmetric key $\lambda_{2,0}$ is used to decrypt this node. Shaded nodes are encrypted.

our data-intensive application. In contrast, if the update period is too large, the situational awareness does not reflect the up-to-date situation about the power grid. In fact, secure and efficient updates of secure indexes is an open problem in the literature. Fortunately, since power flow data is used in critical events, our modeling of data provides a solution to the problem in the particular context. A few minutes

worth of data before the events are most important for situational awareness. For this reason, if we let the update period of associated data files and their secure index be twice as long as this interval, we can ensure that the most critical data will fall into a single data collection.

5.4.2.2 Secure Data Search

Besides the usage in critical situational awareness, the data center can also serve as the modern storage-as-a-service model to archive power flow data from different utility companies. It is straightforward for the operator P_n to search and retrieve its data files stored in the data center. In this case, P_n just needs to compute the trapdoors for the keywords it wants to search, *i.e.* , $TD_{kw} =< prp'_c(kw), prf_b(kw) >$ for each keyword, and send the trapdoors to the data center. The data center will decrypt the linked lists using the trapdoors and return the encrypted data files to P_n as shown in Fig. 5.3. This is the search mechanism provided by the original construction of secure index. However, it bears the drawback of lacking guarantee of hiding the access pattern and search pattern, which is also considered as a side-channel private information during private information retrieval. We give a solution to address the problem as follows.

The basic idea is to extend a linked list to contain other keywords in addition to the intended one. For example, linked list L_i contains only nodes $L_{i,j},\ \forall j \in [1, |Kw_i|]$. In the proposed pattern hiding scheme, each linked list will hold multiple (but not the same number of) keywords and each keyword will appear in many (but not the same number of) linked lists, e.g., L_i is now constructed to include two other keywords kw_g and kw_h. Then L_i should contain all nodes for the three keywords, *i.e.* , $L_{i,j},\ \forall j \in [1, |kw_i|],\ L_{g,j},\ \forall j \in [1, |kw_g|]$ and $L_{h,j},\ \forall j \in [1, |kw_h|]$. To search for kw_h-related files, the operators can deliberately submit a trapdoor calculated from kw_i. As the file identifiers associated with all the three keywords will be returned, the operator can select the ones containing the desired keyword kw_h. Similarly, since kw_i is included in the other linked lists, say L_o, the operator can submit a search based on kw_o to

disguise the actual search for kw_i-related files. The pattern hiding scheme is described as follows for each keyword i in the keyword space:

1. Randomly select an integer m between 1 and $N = |kw|$ where N is also the number of linked list to be constructed. The integer m determines how many different linked lists will contain kw_i.

2. Then randomly generate an array of $m-1$ integers between 1 and N, indicating which linked lists will include kw_i besides L_i. We can keep running this process until we have m distinct integers. Suppose $i = 1, N = 8, m = 4$, and the array of integers $M = (1\ 8\ 4\ 5)$. The array M shows the positions of kw_i, *i.e.* , kw_i is contained in the ith, 4th, 5th and 8th linked lists.

3. Record the positions of kw_i in a matrix Q by setting the corresponding elements to 1 (otherwise 0), e.g., $Q_{i,k} = 1$ represents the ith keyword is contained in the kth linked list. Sum the columns of Q and record the results in an array B, which indicates how many different keywords are contained in the corresponding linked lists. Using the above example, the first row of Q is $Q_1 = (1\ 0\ 0\ 1\ 1\ 0\ 0\ 1)$. Suppose as the process continues, $Q_2 = (0\ 1\ 0\ 1\ 0\ 0\ 1)$, then $B = (1\ 1\ 0\ 2\ 1\ 0\ 0\ 2)$ indicating the 1st, 2nd and 5th linked lists contain 1 keyword, the 4th and 8th linked lists include 2 keywords, and so on.

4. The actual construction of linked list is based on array B and matrix Q. Suppose the 1st linked list L_1 contains three keywords kw_1, kw_7 and kw_8. We start the construction by linking all nodes for kw_1 first. The last node for kw_1, $L_{1,|kw_1|}$ will be linked to the first node for kw_7(or kw_8), $L_{7,1}$(or $L_{8,1}$), etc.

5.4.3 Evolving Key Based Flexible-Grained Access Control

5.4.3.1 Hierarchical Evolving Keys

In previous discussions, every file stored in the data center is associated with three attributes, *i.e.* the three dimensions of data model: the data category, time scale,

and spatial scale. As such, attribute-based encryption Goyal et al. (2006) seems a good fit to enforce the access control mechanism of the data. However, ABE is not the optimal solution in this context for two considerations. First, attribute-based encryption and other public-key cryptosystem are not suitable for encrypting large chunks of data because they necessitate an enormous amount of computationally expensive modular exponentiation (*e.g.* , RSA Rivest et al. (1978)) and pairing (*e.g.* , identity-based encryption Boneh and Franklin (2003), Goyal et al. (2006)), which make the encryption and decryption very slow. Second, even though we may use ABE to encrypt only the symmetric keys that are used to encrypt the files to avoid the problem, as the time constantly elapses, the universe of attributes of the ABE keeps growing, making the solution inefficient and unscalable Goyal et al. (2006).

As a solution, we propose a hierarchical evolving key scheme employing only the secret key encryption. This evolving key scheme is designed based on the data model in Fig. 5.1. Assuming there are N grid operators, without loss of generality, the figure shows the data model of operator P_n. The secret keys used by P_n are generated as follows:

α: α is the root master key.

β: $\beta_i = E_\alpha(i)$ $(i \in [1, I])$ is the temporary master key for the period U_i.

γ: $\gamma_j = E_{\beta_i}(j)$ $(j \in [1, J])$ is the category key for data category D_{ij} and the time scale seed key for D_{ij}'s data collected in different time periods.

θ: $\theta_t = E_{\gamma_j}(t)$ is the time scale key for the data in category D_{ij} collected in time period $(t-1, t]$.

δ: $\delta_k = E_{\delta_{k-1}}(k)$ $(k \in [1, K])$ is the encryption key for the data on spacial scale k, D_{ijk}, within the time period $(t-1, t]$, and $\delta_0 = \theta_t$. Data on a same spacial scale within a same time period, *e.g.* , $D_{ijk}(TS_1^t), \cdots, D_{ijk}(TS_m^t), \cdots$, constitute a data file that is encrypted by δ_k.

The root master key α is used to generate all other keys and, thus, is the only key that needs to be securely stored. Referring to Fig. 5.1, β_i is generated for each time period U_i for master key update. It is, in general, a bad security practice to use a secret key for long either to generate other keys or to encrypt data, because keys "wear out" after certain uses. Thus, it is desirable to change the master key once in a while. The more frequently a temporary master key is used, the more often we should update it, or in other words, the shorter we should let the period U_i be. γ_j is generated for each data category, which includes all data of this category collected in different time periods T's within the period U_i ($U_i \gg T_t$). θ_t is generated for the collected data in each time period, *e.g.* , $D_{ij}(T_t)$. The collected data further include individual datums (*e.g.* , $D_{ijk}(TS_m^t)$) obtained at different time instances TS's within the time period T_t. Each datum has different spatial scales for visualization purpose and δ_k is generated for each scale (*e.g.* , $D_{ijk}(TS_1^t), \cdots, D_{ijk}(TS_m^t), \cdots$). The key for higher spatial scale is used to generate the keys for lower scales, *e.g.* , δ_{k-1} is used to generate δ_k and below. This is because if access to a particular spatial scale is allowed, then access to all lower scales is also allowed.

In essence, the δ_k's are the actual keys encrypting/decrypting the data files. All the upper-level keys in the key hierarchy, *i.e.* , β_i's, γ_j's, and θ_t's, are intermediate keys that are used to create an efficient key management scheme and re-generate the δ_k's, and are never used to encrypt a file.

Note that the evolving key scheme also offers efficient data integrity protection. Data integrity can be ensured by a keyed hash (*e.g.* , HMAC, CBC residue) computed over and appended to each data file. The secret key used by the keyed hash can be the same as the encryption/decryption key for the data file. However, a better security practice would be to use different keys for encryption and integrity protection. For example, our encryption keys are generated as $\delta_k = E_{\delta_{k-1}}(k)$ ($k \in [1, K]$), $\delta_0 = \theta_t$. We can add an extra branch under θ_t in the key hierarchy for the integrity keys ϕ_k as $\phi_k = E_{\phi_{k-1}}(k)$ ($k \in [1, K]$), $\phi_0 = E_{\theta_t}(2)$, and modify δ_0 to be $\delta_0 = E_{\theta_t}(1)$. This is in

accordance with the fact that the encryption and integrity protection keys for a data file always come in pairs.

5.4.3.2 Flexible-Grained Access Control

Flexible-grained access control takes advantage of the most important property of the evolving keys: keys are constructed in a hierarchical fashion. On one hand, the lower-level keys can be efficiently derived from the upper-level key given the auxiliary information *i.e.* the files' categories, time scales, location scale. For instance, given a temporary master key α_i and the indices (j, t, k), it is trivial to find the encryption key δ_k that is used to encrypt the corresponding data file. On the other hand, the reverse cannot be done *i.e.* it is impossible to derive a upper-level key from a lower-level key.

By granting the data requester a high-level key, the data provider grants coarse-grained access, as the auxiliary information can be found in the secure index. By giving only the keys used to encrypt individual files, the data provider limits the data requester's access to only the set of files corresponding tot he encryption keys.

Coarse-grained access is advantageous in situations where efficiency and requirement for low-latency take priority over the data security and privacy. When the data center needs data from the grid operators for situational awareness, it will send requests to the operators including the needed keywords. In fact, the operators are likely to know roughly what data are useful for this purpose. For example, 5 to 10 minutes worth of data prior to the event is usually needed for situational awareness. The operators will send the corresponding trapdoors and key(s) to the data center upon detecting the event, without the data center requesting it. The data center will first use the trapdoors to decrypt the linked lists and then extract the indices $(i \parallel j \parallel t \parallel k)$ and file identifiers from the linked list nodes. Coarse-grained access control is desirable in this case because the timely generation of results for situational awareness is of top priority. Accordingly, the operators may choose to reveal more data than necessary, for example, the data of an entire category D_{ij}, thus trading security for efficiency. The adjustability is critical because it may not be feasible for

the operators to select the encryption keys for the particular data files needed by the data center during the course of situational awareness. Using the above example, all information needed from the operators is a single data category key γ_j and the data center enjoys the freedom to select the most useful files at a later time. Since the data center has the category key and the indices $(i \parallel j \parallel t \parallel k)$, it can generate any encryption key by first generating the appropriate time scale key. The method for deriving the keys are discussed in the evolving key scheme (Section. 5.4.3). Finally, the derived encryption keys will be used to decrypt the data files indexed by the file identifiers.

In contrast, fine-grained access is more desirable in the scenarios where the data provider want specific files to be shared. The following scenario illustrates such scenarios. When grid operators, say P_n and P_{n+1}, need to share data in order to respond to some local events, they will send each other trapdoors associated with the keywords allowed for search. Fine-grained access control is more desirable in this case because it helps the data provider to expose only the data that is necessary for this incident to its competitors. For example, P_n only allows P_{n+1} to see the data on the bottom two spacial scales, *e.g.* , $D_{ij(K-1)}$ and D_{ijK}, within a strict time period, *e.g.* , $(t-1, t]$, for a certain event. Then P_n only sends the encryption key δ_{K-1} to P_{n+1}. If more data are needed, P_{n+1} must make additional requests. Upon receiving the trapdoors and keys, P_{n+1} will submit the trapdoors to the data center that returns the associated encrypted files. These files will then be decrypted using the encryption keys received from P_n. When the stored data are retrieved for the data owner operator's use or for other operators, the data center serves merely as a storage server that is not authorized to know what keywords are being searched. The strictness of the access control can be adjusted by the evolving key scheme.

5.5 Security Analysis

In this section, we show that our proposed schemes meet the security goals put forth in Section 3.3.

Privacy: As stated in section 3.3, data privacy in this work involves two aspects: data confidentiality and unlinkability. For data confidentiality, we consider two scenarios that may potentially compromise data confidentiality, that is, when the data is stored in the data center and when the operator searches over its own data. There are two kinds of data stored in the data center, *i.e.* the encrypted files and Secure Index. The confidentiality of files depends on the strength of the algorithm for data encryption, which is assumed to be secure against cryptographic attacks. In addition, considering that keys "wear out" with frequent applications, the security is further fortified by the practice of updating the master key periodically. When we consider the confidentiality of Secure Index, we want to make sure that the sensitive information, including keywords, file identifiers, and decryption keys in the Secure Index will not be leaked. First, in the lookup table of Secure Index, keywords are stored in the ciphertext of $prpt'_c(kW)$ instead of in plaintext. Second, also in the lookup table, the address and decryption key of the first node of a linked list are masked by the output of a pseudorandom function as it is shown in $T[prpt'_c(kw)] = (addr_{x,1} \| \gamma_{x,0}) \oplus prf_b(kw)$. Since the decryption of a linked list node requires the decryption key and ptr stored in the previous node, all the subsequent nodes would not be revealed to anyone who does not know the secret b. We refer the reader to *Thm. 3* in Goh (2003) for the mathematical proof. Another scenario that has the potential to cause data confidentiality breaches is when the operator executes searching and retrieving. When the operator wants to fetch the files containing a keyword kW, it would compute a trapdoor $TD(kw) = prpt_c(kw)$ and submits the trapdoor to the data center. By submitting the trapdoor instead of the kw, the data center will not be able to know the content of the keyword. We next discuss the second aspect of privacy, *i.e.* , unlinkability, which requires that unauthorized entities

must not be able to link multiple data files to profile the operator. We realize this property by storing the files to the addresses that are decided by the pseudorandom permutation $ptr = prp_a(C)$ Curtmola et al. (2006). As a result, the files are stored scrambled and appear randomly, and physically adjacent nodes do not have logical relations, for instance, sharing the same keywords or belonging to the same operator.

Fine-grained access control: This property requires that the operator who owns the data has control over who can access which portion of the data. The major challenge is to make sure that the authorized party can only access the portion it is authorized to access. Note that the encryption key is derived from its immediate upper-level key with AES. For instance, the encryption key for the data in category Dij is given by $\theta_t = E_{\gamma j}(k)$, where γ_j is the key for category D_{ij} and the upper-level key in this case and E denotes the encryption algorithm in use, *i.e.* , AES in our schemes. The use of AES ensures that it is computationally infeasible to reverse the encryption and obtain the key of higher hierarchy from that of a low one.

Data integrity: Data integrity is guaranteed by a keyed hash (e.g., HMAC, CBC residue). The keyed hashes are computed from the files and are appended to data files, as we described earlier. The real challenge lies in the key management is difficult to manage so many keys associated with a huge number of files. Fortunately, this issue has been solved by adding extra branches in the evolving key scheme.

5.6 Performance Evaluation

We implemented a prototype of the proposed data sharing mechanism in Java programming language. and evaluate the performance of the prototype in terms of latency of the proposed scheme. To evaluate the performance, we generate our test files by vertically partitioning the real power flow data from the 2,736 bus Polish system test case provided by MATPOWER Zimmerman et al. (2011). Each test file contains a JSON object that holds a list of keywords and a matrix of the power flow data about a number of entities in the power system and the file's keywords. The

Table 5.1: Runtime of common operations of the proposed data sharing mechanism for situational awareness

Operations	**Average runtime**	**Note**
Generating a trapdoor	0.001 ms	
Querying one keyword	0.004 ms	retrieve the responsive encrypted files
Deriving a encryption key	0.004 ms	in a four hierarchy data model
Encrypt one file	0.035 ms	throughput is 275 MB/s
Decrypt one file	0.035 ms	throughput is 273 MB/s

rows of the matrix correspond to buses, generators, or branches. A test file typically contains one hundred rows, which partitions all power flow data in the test cases into seventy files. The columns of the matrix are the parameters about the each entity (bus, generator, or branch). These include real power demand, reactive power deman, voltage magnitude, *etc.* . A list of parameters of the test cases can be found in Zimmerman et al. (2011). The keywords of a files includes the related bus numbers and the names of the parameters. The average size of these files is 0.017MB.

In the experiment, we first build the secure index with all the test files and their associated keywords. We then outsource the generated secure index and their AES encrypted test files to the S3A server, where the evaluations are performed. Our evaluation results are presented in Table 5.1. The measured runtimes listed in the table are the average of 100 measurements taken on a workstation with Intel Core i7 CPU and 8 GB RAM running Java Virtual Machine (JVM) Version 1.8 and Linux operating system. The key size of the AES encryption is 128 bits.

Chapter 6

Conclusions and Future Work

In this study, we investigate data security and privacy problems that span the whole life cycle of the data in the smart grid.

First, in Chapter 2, we conducted a vulnerability assessment of the phasor network by the mean of penetration tests. The penetration test consists of three phases: reconnaissance, exploration, and exploit development. By testing techniques, including packet sniffing, packet injection, fuzzing, we have identified vulnerabilities associated with the IEEE C37.118 and the specific software implementations of the PMU and PDC under test. We also showed that the found vulnerabilities can impact phasor network based Wide-area monitoring system (WAMS) by demonstrating how can an attacker exploit the found vulnerabilities to perform a data stream hijacking attack to manipulate the situational awareness result based on the real-time phasor measurement data. We also present security recommendations to mitigate the vulnerabilities and enhance the data security of the phasor network. Future work will consider more complex, *e.g.* multilayered large scale phasor networks and attempt to identify relevant vulnerabilities.

In Chapter 3, we presented a novel approach, which combines the disguising technique and the code obfuscation, to secure the outsourcing of non-linear power system dynamic simulations against potential privacy breaches. We implemented

and tested the proposed schemes. It is shown that the trade-off between feasibility, efficiency, and security has been made. Future work will focus on improving the security protection by replacing code obfuscation with provably secure homomorphic encryption and garbled circuit.

In Chapter 4, we presented a cloud-assisted multiparty spectrum estimation for inter-area oscillation monitoring. The proposed scheme leverages the additive homomorphic Paillier cryptosystem to enable computations directly over the encrypted data so as to ensure the participants' privacy. The proposed scheme also enables access control in the frequency domain. Security and complexity analysis are carried out, which showed that our proposed scheme is scalable, efficient, and secure. Future work will investigate method to build auditability, by, for example, enable the control room to validate the cloud server's work in zero knowledge, so we can relax the assumption that the cloud servers are semi-honest. Another research direction worth considering is to extend the idea of using homomorphic cryptographies to build privacy protection into more PMU data. applications.

In Chapter 5, we proposed S^3A, a secure data sharing mechanism for *global* situational awareness of the power grid. The contributions of our proposed schemes are listed as follows:

1. design of secure and unlinkable data sharing mechanisms that enable peer operators to outsource and share their data without compromising their privacy using light-weighted private key cryptography encryption. To our best knowledge, our proposed schemes are the first to consider the data sharing in critical events of the power grid.

2. design of novel key generation and management mechanisms that are specific to the unique model of power flow data to achieve fine-grained access control.

The proposed architecture achieved desired security goals and have a satisfactory performance with respective to low-latency and high-throughput.

Bibliography

Atallah, M. J., Pantazopoulos, K. N., Rice, J. R., and Spafford, E. H. (2001). Secure outsourcing of scientific computations. *Advances in Computers*, 54:215–272. 51, 57

Baharlouei, Z. and Hashemi, M. (2014). Efficiency-fairness trade-off in privacy-preserving autonomous demand side management. *Smart Grid, IEEE Transactions on*, 5(2):799–808. 2, 5, 75

Bakken, D., Hauser, C., Gjermundrod, H., and Bose, A. (2007). Towards more flexible and robust data delivery for monitoring and control of the electric power grid. *Technical Report EECS-GS-009, Washington State University*. 98

Barak, B., Garg, S., and Kalai, Y. (2014). Protecting obfuscation against algebraic attacks. *Advances in Cryptology ...*. 70

Barak, B., Goldreich, O., Impagliazzo, R., Rudich, S., Sahai, A., Vadhan, S., and Yang, K. (2012). On the (im)possibility of obfuscating programs. *J. ACM*, 59(2):6:1–6:48. 62

Baumeister, T. (2010). Literature review on smart grid cyber security. *University of Hawaii at Manoa, Tech. Rep.* 15

Beasley, C. (2014). Electric power synchrophasor network cyber security vulnerabilities. 15

Beasley, C., Zhong, X., Deng, J., Brooks, R., and Kumar Venayagamoorthy, G. (2014a). A survey of electric power synchrophasor network cyber security. In *Innovative Smart Grid Technologies Conference Europe (ISGT-Europe), 2014 IEEE PES*, pages 1–5. 47

Beasley, C., Zhong, X., Deng, J., Brooks, R., and Venayagamoorthy, G. K. (2014b). A survey of electric power synchrophasor network cyber security. In *Innovative Smart Grid Technologies Conference Europe (ISGT-Europe), 2014 IEEE PES*, pages 1–5. 15

Bianchi, T., Piva, A., and Barni, M. (2009). On the implementation of the discrete fourier transform in the encrypted domain. *Information Forensics and Security, IEEE Transactions on*, 4(1):86–97. 75

Biondi, P. (2014). Scapy. http://www.secdev.org/projects/scapy/. Retrieval date: Oct 30th, 2014. 21, 23

Bisciglia, C. (2009). The smart grid: Hadoop at the tennessee valley authority (tva). http://blog.cloudera.com/blog/2009/06/smart-grid-hadoop-tennessee-valley-authority-tva/. Accessed: 2014-09-28. 79

Bobba, R., Heine, E., Khurana, H., and Yardley, T. (2010). Exploring a tiered architecture for naspinet. In *Innovative Smart Grid Technologies (ISGT), 2010*, pages 1–8. IEEE. 16

Bobba, R., Khurana, H., AlTurki, M., and Ashraf, F. (2009). PBES: A policy based encryption system with application to data sharing in the power grid. *ASIAN ACM Symposium on Information, Computer and Communications Security (ASIACCS'09)*, pages 262–275. 98

Boneh, D. and Franklin, M. (2003). Identity-based encryption from the weil pairing. extended abstract in CRYPTO 2001. *SIAM J. of Computing*, 32(3):586–615. 108

Borges, F., Demirel, D., Bock, L., Buchmann, J., and Muhlhauser, M. (2014). A privacy-enhancing protocol that provides in-network data aggregation and verifiable smart meter billing. In *Computers and Communication (ISCC), 2014 IEEE Symposium on*, pages 1–6. 2, 5, 75

Bou-Harb, E., Fachkha, C., Pourzandi, M., Debbabi, M., and Assi, C. (2013). Communication security for smart grid distribution networks. *Communications Magazine, IEEE*, 51(1):42–49. 15

Brincker, R., Ventura, C., and Andersen, P. (2001). Damping estimation by frequency domain decomposition. *IMAC XIX, Kissimmee, USA*. 9, 72

Brincker, R., Zhang, L., and Andersen, P. (2000). Modal identification from ambient responses using frequency domain decomposition. In *Proc. of the 18*International Modal Analysis Conference (IMAC), San Antonio, Texas*. 72

Campbell, R. J. (2011). The smart grid and cybersecurity: Regulatory policy and issues. Congressional Research Service, Library of Congress. 15

Cao, N., Wang, C., Li, M., Ren, K., and Lou, W. (2011a). Privacy-preserving multi-keyword ranked search over encrypted cloud data. *IEEE Conf. on Computer Communications (INFOCOM)*. 97

Cao, N., Yang, Z., Wang, C., Ren, K., and Lou, W. (2011b). Privacy-preserving query over encrypted graph-structured data in cloud computing. *IEEE Intl. Conf. on Distributed Computing Systems (ICDCS'11)*. 97

Ceccato, M. and Penta, M. D. (2009). The effectiveness of source code obfuscation: an experimental assessment. *..., 2009. ICPC'09. ...*, pages 178–187. 69

Centre for the Protection of National Infrasctructure and Department of Homeland Security (2010). Cyber Security Assessments of Industrial Control Systems. https://ics-cert.us-cert.gov/sites/default/files/documents/Cyber_Security_Assessments_of_Industrial_Control_Systems. pdf. Retrieval date: Oct 30th, 2014. 7, 21

Chakrabarti, S., Kyriakides, E., Bi, T., Cai, D., and Terzija, V. (2009). Measurements get together. *Power and Energy Magazine, IEEE*, 7(1):41–49. 9, 72

Chaum, D., Crépeau, C., and Damgard, I. (1988). Multiparty unconditionally secure protocols. In *Proceedings of the Twentieth Annual ACM Symposium on Theory of Computing*, STOC '88, pages 11–19, New York, NY, USA. ACM. 50

Chow, J. and Cheung, K. (1992). A toolbox for power system dynamics and control engineering education and research. *Power Systems, IEEE Transactions on*, 7(4):1559–1564. 48, 53

Chow, J. and Rogers, G. (2000). Power system toolbox. *Cherry Tree Scientific Software,[Online] Available: http://www. ecse. rpi. edu/pst/PST. html.* 48, 53

Collberg, C., Thomborson, C., and Low, D. (1997). A taxonomy of obfuscating transformations. 62, 64, 69

Curtmola, R., Garay, J., Kamara, S., and Ostrovsky, R. (2006). Searchable symmetric encryption: improved definitions and efficient constructions. *in ACM Conference on Computer and Communications Security (CCS), Alexandria, Virginia.* 97, 104, 113

CVE official site (2014). Common vulnerabilities and exposures. https://ipdc.codeplex.com. Retrieval date: Oct 30th, 2014. 22

D'Antonio, S., Coppolino, L., Elia, I. A., and Formicola, V. (2011). Security issues of a phasor data concentrator for smart grid infrastructure. In *Proceedings of the 13th European Workshop on Dependable Computing*, pages 3–8. ACM. 15

Du, W. and Atallah, M. J. (2001). Privacy-preserving cooperative scientific computations. In *Proceedings of the 14th IEEE Computer Security Foundations Workshop*, pages 273–282, Washington, DC, USA. IEEE Computer Society. 70

EIOC, P. (2013). Situational awareness. url: http://eioc.pnnl.gov/research/sitawareness.stm. accessed Oct 8, 2014. 10, 72, 95, 96

Elaine Shi, R. C., h. Hubert Chan, T., Song, D., and Rieffel, E. (2011). Privacy-preserving aggregation of time-series data. In *In NDSS*. 2, 5, 75

Electricity Consumers Resource Council, E. (2003). The economic impacts of the august 2003 blackout. url: http://www.elcon.org/Documents/ EconomicImpact-sOfAugust2003Blackout.pdf. accessed Oct 8, 2014. 95

Franklin, M. (2006). A survey of key evolving cryptosystems. *Int. J. of Security and Networks (IJSN)*, 1(1-2). 97

Fu, C. (2011). High-speed extended-term time-domain simulation for online cascading analysis of power systemalysis of power system. 67

Garg, S., Gentry, C., and Halevi, S. (2013). Candidate indistinguishability obfuscation and functional encryption for all circuits. *... of Computer Science ....* 70

Gentry, C. (2009). *A fully homomorphic encryption scheme*. PhD thesis, Stanford University. 50, 70

Goh, E.-J. (2003). *Secure indexes*. in Cryptology ePrint Archive: Report 2003/216, available at http://eprint.iacr.org/2003/216/. 97, 112

Goyal, V., Pandey, O., Sahai, A., and Waters, B. (2006). Attributed-based encryption for fine-grained access control of encrypted data. *in ACM Conference on Computer and Communications Security (CCS)*, pages 89–98. 108

Green, M., Hohenberger, S., and Waters, B. (2011). Outsourcing the decryption of abe ciphertexts. In *In Proceedings of the USENIX Security Symposium*. 50

Grid Protection Alliance (2014). Open pdc project. http://openpdc.codeplex.com/. Retrieval date: Oct 30th, 2014. 17, 20

Halfond, W., Viegas, J., and Orso, A. (2006). A classification of sql-injection attacks and countermeasures. In *Proceedings of the IEEE International Symposium on Secure Software Engineering*, volume 1, pages 13–15. IEEE. 39

Hohenberger, S. and Lysyanskaya, A. (2005). How to securely outsource cryptographic computations. In *Proceedings of the Second International Conference on Theory of Cryptography*, TCC'05, pages 264–282, Berlin, Heidelberg. Springer-Verlag. 50

Huang, H. and Guttromson, R. (2005). Archiving and management of power systems data for real-time performance monitoring platform. 6

IEEE Standards Association (2014). C37.118.1-2011-ieee standard for synchrophasor measurements for power systems. http://standards.ieee.org/findstds/standard/C37.118.1-2011.html. Retrieval date: Oct 30th, 2014. 19

International Electrotechnical Commission (2014). Core iec standards. http://www.iec.ch/smartgrid/standards/. Retrieval date: Oct 30th, 2014. 11

ISO New England (2014). Regional electricity outlook. 8, 49, 79

Itkis, G. and Xie, P. (2003). Generalized key-evolving signature schemes or how to foil an armed adversary. *in Applied Cryptography and Network Security Conference*. 97

Kaufman, C., Perlman, R., and Speciner, M. (2002). *Network Security: Private Communication in a Public World, Second Edition*. Prentice Hall Press, Upper Saddle River, NJ, USA, second edition. 3, 7, 26

Khandeparkar, K. and Pandit, N. (2014). ipdc - free phasor data concentrator. https://ipdc.codeplex.com/. Retrieval date: Oct 30th, 2014. 20

Khurana, H., Bobba, R., Yardley, T., Agarwal, P., and Heine, E. (2010a). Design principles for power grid cyber-infrastructure authentication protocols. *in Proc. Hawaii International Conference on System Sciences*. 98

Khurana, H., Hadley, M., Lu, N., and Frincke, D. (2010b). Smart-grid security issues. *Security Privacy, IEEE*, 8(1):81–85. 5

Klump, R. and Weber, J. (2002). Real-time data retrieval and new visualization techniques for the energy industry. In *Proc. of the Annual Hawaii Int. Conference on System Sciences*, pages 712–717. 97

Kosut, O., Jia, L., Thomas, R., and Tong, L. (2010). Malicious data attacks on smart grid state estimation: Attack strategies and countermeasures. *in Proc. IEEE Int'l Conf. on Smart Grid Communications*, pages 220–225. 98

Kundur, P., Balu, N. J., and Lauby, M. G. (1994). *Power system stability and control*, volume 7. McGraw-hill New York. xi, 77, 78, 80

Lagendijk, R., Erkin, Z., and Barni, M. (2013). Encrypted signal processing for privacy protection: Conveying the utility of homomorphic encryption and multiparty computation. *Signal Processing Magazine, IEEE*, 30(1):82–105. 75

Li, J., Wang, Q., Wang, C., Cao, N., Ren, K., and Lou, W. (2010a). Fuzzy keyword search over encrypted data in cloud computing. *IEEE Conf. on Computer Communications (INFOCOM'10), San Diego, California*, pages 1–5. 97

Li, M., Yu, S., Ren, K., and Lou, W. (2010b). Securing personal health records in cloud computing: Patient-centric and fine-grained data access control in multi-owner settings. *Proc. 6th Int'l Conf. on Security and Privacy in Communication Networks (SecureComm'10)*. 97

Liesdonk, P., Sedghi, S., Doumen, J., Hartel, P., and Jonker, W. (2010). Computationally efficient searchable symmetric encryption. *in Proc. 7th Intl. Workshop on Secure Data Management (SDM'10). LNCS Vol. 6358, Springer Verlag*, pages 87–100. 97

Linn, C. and Debray, S. (2003). Obfuscation of executable code to improve resistance to static disassembly. *Proceedings of the 10th ACM conference on Computer and communication security - CCS '03*, page 290. 70

Litvinov, E. (2014). Early experience with cloud computing at iso new england. 8, 49, 79

Liu, Y., Ning, P., and Reiter, M. K. (2009). False data injection attacks against state estimation in electric power grids. In *Proceedings of the 16th ACM Conference on Computer and Communications Security*, CCS '09, pages 21–32, New York, NY, USA. ACM. 3, 4, 14

Lu, Z., Lu, X., Wang, W., and Wang, C. (2010). Review and evaluation of security threats on the communication networks in the smart grid. In *MILITARY COMMUNICATIONS CONFERENCE, 2010 - MILCOM 2010*, pages 1830–1835. 3, 5

Lyon, G. F. (2009). *Nmap Network Scanning: The Official Nmap Project Guide to Network Discovery and Security Scanning*. Insecure. 21

Martin, K., Brunello, G., Adamiak, M., Antonova, G., Begovic, M., Benmouyal, G., Bui, P., Falk, H., Gharpure, V., Goldstein, A., Hu, Y., Huntley, C., Kase, T., Kezunovic, M., Kulshrestha, A., Lu, Y., Midence, R., Murphy, J., Patel, M., Rahmatian, F., Skendzic, V., Vandiver, B., and Zahid, A. (2014). An overview of the ieee standard c37.118.2 — synchrophasor data transfer for power systems. *Smart Grid, IEEE Transactions on*, 5(4):1980–1984. 17

Maynor, D. (2011). *Metasploit Toolkit for Penetration Testing, Exploit Development, and Vulnerability Research*. Elsevier. 21

McClure, S., Scambray, J., Kurtz, G., and Kurtz (2009). *Hacking exposed: network security secrets and solutions*. McGraw-Hill. 26

Metke, A. and Ekl, R. (2010). Security technology for smart grid networks. *Smart Grid, IEEE Transactions on*, 1(1):99–107. 3

Miller, B. and Rowe, D. (2012). A survey SCADA of and critical infrastructure incidents. In *Proceedings of the 1st Annual conference on Research in information technology - RIIT '12*, page 51, New York, New York, USA. ACM Press. 2, 13

Miller, B. P., Fredriksen, L., and So, B. (1990). An empirical study of the reliability of unix utilities. *Communications of the ACM*, 33(12):32–44. 23

Mo, Y., Kim, T.-H., Brancik, K., Dickinson, D., Lee, H., Perrig, A., and Sinopoli, B. (2012). Cyber-physical security of a smart grid infrastructure. *Proceedings of the IEEE*, 100(1):195–209. 3

Morris, T., Pan, S., Lewis, J., Moorhead, J., Younan, N., King, R., Freund, M., and Madani, V. (2011). Cybersecurity risk testing of substation phasor measurement units and phasor data concentrators. In *Proceedings of the Seventh Annual Workshop on Cyber Security and Information Intelligence Research*, page 24. ACM. 14, 36

Naehrig, M., Lauter, K., and Vaikuntanathan, V. (2011). Can homomorphic encryption be practical? *Proceedings of the 3rd ACM workshop on Cloud computing security workshop - CCSW '11*, page 113. 50

NETL Modern Grid Strategy (2008). Advanced metering infrastructure. *US Department of Energy, National Energy Technology Laboratory, Mag*, 1. 11

Nikolaenko, V., Weinsberg, U., Ioannidis, S., Joye, M., Boneh, D., and Taft, N. (2013). Privacy-preserving ridge regression on hundreds of millions of records. In *Security and Privacy (SP), 2013 IEEE Symposium on*, pages 334–348. 50

NIST (2014). National vulnerability database. https://nvd.nist.gov. Retrieval date: Oct 30th, 2014. 22

North America Electric Reliability Corporation, NERC (2011). Real-time application of synchrophasors for improving reliability. http://www.nerc.com/docs/oc/rapirtf/RAPIR retrieval data: 10/12/2014. 15, 72

North American Electric Reliability Corperation (2009). Operating reliability data confidentiality agreement, version 3. http://www.nerc.com/comm/OC/Pages/OperatingReliability DataConfidentialityAgreement.aspx. Retrieval date: Nov. 20th, 2014. 4, 72, 73

Office of Electicity Delivery and Energy Reliability, Department of Energy (2009). Smart grid: An introduction. http://energy.gov/oe/services/technology-development/smart-grid. retrieval date: Oct. 23rd, 2014. 1, 15

Office of the National Coordinator for Smart Grid Interoperability (2010). Nist framework and roadmap for smart grid interoperability standards, release 1.0. http://www.nist.gov/public_affairs/ releases/upload/smartgrid_interoperability _final.pdf. retrieval date: Oct. 23rd, 2014. x, 2

Paillier, P. (1999). Public-key cryptosystems based on composite degree residuosity classes. In *Advances in cryptologyEUROCRYPT99*, pages 223–238. Springer. 73, 76, 87

Power Tech Lab (2015). Tsat - transient security assessment tool. http://www.powertechlabs.com/areas-of-focus/smart-utility/dsatools-software/transient-security-assessment-tool/. Retrieval date: March 28th, 2014. 42

Qi, H., Liu, Y., Li, F., Luo, J., He, L., Tomsovic, K., Tolbert, L., and Cao, Q. (2011). Increasing the resolution of wide-area situational awareness of the power grid through event unmixing. *in Proc. Hawaii International Conference on System Sciences*, pages 1–8. 98

Reinsch, C. H. (1967). Smoothing by spline functions. *Numerische mathematik*, 10(3):177–183. 58

Rivest, R., Shamir, A., and Adleman, L. (1978). A method for obtaining digital signatures and public-key cryptosystems. *Communications of the ACM*, 21(2):120–126. 108

Rogers, K., Klump, R., Khurana, H., Aquino-Lugo, A., and Overbye, T. (2010). An authenticated control framework for distributed voltage support on the smart grid. *IEEE Trans. Smart Grid*, 1(1):40–47. 98

Sahai, A. and Waters, B. (2013). How to Use Indistinguishability Obfuscation: Deniable Encryption, and More. *IACR Cryptology ePrint Archive*. 70

Shepard, D. P., Humphreys, T. E., and Fansler, A. A. (2012). Evaluation of the vulnerability of phasor measurement units to gps spoofing attacks. *International Journal of Critical Infrastructure Protection*, 5(3):146–153. 15

Silver-Greenberg, J., GOLDSTEIN, M., and PERLROTH, N. (2014). Jpmorgan chase hacking affects 76 million households. http://dealbook.nytimes.com/2014/10/02/jpmorgan-discovers-further-cyber-security-issues/?_r=0. Retrieval date: Oct 30th, 2014. 2, 13

Sprabery, R., Morris, T. H., Pan, S., Adhikari, U., and Madani, V. (2013). Protocol mutation intrusion detection for synchrophasor communications. In *Proceedings of the Eighth Annual Cyber Security and Information Intelligence Research Workshop*, page 41. ACM. 15, 47

Stallings, W. (1995). *Network and internetwork security: principles and practice*, volume 1. Prentice Hall Upper Saddle River, NJ. 7

Stewart, J., Maufer, T., Smith, R., Anderson, C., and Ersonmez, E. (2010). Synchrophasor security practices. *Schweitzer Engineering Laboratories, Pullman, Washington (¡ www. selinc. com/WorkArea/DownloadAsset. aspx.* 14, 15, 20, 45

Sun, J., Zhu, X., and Fang, Y. (2010). A privacy-preserving scheme for online social networks with efficient revocation. *IEEE Conf. on Computer Communications (INFOCOM)*, pages 2516–2524. 97

Sun, J., Zhu, X., Zhang, C., and Fang, Y. (2011). HCPP: Cryptography based secure EHR system for patient privacy and emergency healthcare. *IEEE Intl. Conf. on Distributed Computing Systems (ICDCS'11)*. 97

Sun, K., Luo, X., and Wong, J. (2012). Early warning of wide-area angular stability problems using synchrophasors. In *Power and Energy Society General Meeting, 2012 IEEE*, pages 1–6. 9, 40, 72, 80, 81, 91

The Open Web Application Security Project (2014). Sql injection prevention cheat sheet. https://www.owasp.org/index.php/SQL_Injection_Prevention_Cheat_Sheet. Retrieval date: May 30th, 2015. 27, 46

Thoma, C., Cui, T., and Franchetti, F. (2013). Privacy preserving smart metering system based retail level electricity market. In *Power and Energy Society General Meeting (PES), 2013 IEEE*, pages 1–5. 2, 5, 75

Tong, Y., Deyton, J., Sun, J., and Li, F. (2013). s^3a : A secure data sharing mechanism for situational awareness in the power grid. *Smart Grid, IEEE Transactions on*, 4(4):1751–1759. 75

Tzeng, W.-G. and Tzeng, Z.-J. (2002). Robust key-evolving public key encryption schemes. *in Proc. 4th Int'l Conf. on Information and Communications Security*. 97

United States Government Accountability Office (2004). Critical infrastructure protection challenges and efforts to secure control systems. x, 12

University of Washington (2014). Power systems test case archive. http://www.ee.washington.edu/research/pstca/. Retrieval date: November 30th, 2014. 10

Vanfretti, L., Bengtsson, S., Peric, V., and Gjerde, J. (2013). Spectral estimation of low-frequency oscillations in the nordic grid using ambient synchrophasor data under the presence of forced oscillations. In *PowerTech (POWERTECH), 2013 IEEE Grenoble*, pages 1–6. 9, 72

Wang, C., Cao, N., and Li, J. (2010a). Secure ranked keyword search over encrypted cloud data. *2010 IEEE 30th International Conference on Distributed Computing Systems*, pages 253–262. 50

Wang, C., Cao, N., Ren, K., and Lou, W. (2012a). Enabling secure and efficient ranked keyword search over outsourced cloud data. *to appear in IEEE Trans. Parallel and Distributed Systems*. 97

Wang, C., Chow, S., Wang, Q., Ren, K., and Lou, W. (2011a). Privacy-preserving public auditing for secure cloud storage. *to appear in IEEE Trans. Computers*. 97

Wang, C., Ren, K., and Wang, J. (2011b). Secure and practical outsourcing of linear programming in cloud computing. In *2011 Proceedings IEEE INFOCOM*, pages 820–828. IEEE. 50

Wang, C., Ren, K., Yu, S., and Urs, K. (2012b). Achieving usable and privacy-assured similarity search over outsourced cloud data. *IEEE Conf. on Computer Communications (INFOCOM)*. 97

Wang, C., Wang, Q., Ren, K., Cao, N., and Lou, W. (2012c). Toward secure and dependable storage services in cloud computing. *Services Computing, IEEE Transactions on*, 5(2):220–232. 97

Wang, Q., Wang, C., Ren, K., Lou, W., and Li, J. (2011c). Enabling public verifiability and data dynamics for storage security in cloud computing. *IEEE Trans. Parallel and Distributed Systems*, 22(5):847–859. 97

Wang, W. and Lu, Z. (2013). Cyber security in the Smart Grid: Survey and challenges. *Computer Networks*, 57(5):1344–1371. 3

Wang, Z., Scaglione, A., and Thomas, R. (2010b). The node degree distribution in power grid and its topology robustness under random and selective node removals. *in Proc. IEEE Int'l Workshop on Smart Grid Communications*. 98

Wireshark Team (2014). Ethernet capture setup. https://wiki.wireshark.org/ CaptureSetup /Ethernet #Capture_using_a_monitor _mode_of_the_switch. Retrieval date: Apr 3rd, 2014. 21, 23

wireshark.org (2014). Wireshark documentations. https://www.wireshark.org/docs/. Retrieval date: Oct 30th, 2014. 23

Wroblewski, G. (2002). *General Method of Program Code Obfuscation (draft)*. PhD thesis, Citeseer. 69

Yan, Y., Qian, Y., Sharif, H., and Tipper, D. (2012a). A survey on cyber security for smart grid communications. *Communications Surveys Tutorials, IEEE*, 14(4):998–1010. 3

Yan, Y., Qian, Y., Sharif, H., and Tipper, D. (2012b). A survey on cyber security for smart grid communications. *Communications Surveys Tutorials, IEEE*, 14(4):998–1010. 15

Yang, Y., McLaughlin, K., Sezer, S., Littler, T., Pranggono, B., Brogan, P., and Wang, H. (2013). Intrusion detection system for network security in synchrophasor systems. In *Information and Communications Technologies (IETICT 2013), IET International Conference on*, pages 246–252. 47

Yao, A. C., Yao, A. C., Yao, A. C., and Yao, A. C. (1982). Protocols for secure computations. In *Foundations of Computer Science, 1982. SFCS '08. 23rd Annual Symposium on*, pages 160–164. 50

Yao, A. C.-C. (1986). How to generate and exchange secrets. In *Foundations of Computer Science, 1986., 27th Annual Symposium on*, pages 162–167. 50

Young, E. . (2014a). Cyber security how safe is your smart grid? http://www.ey.com/GL/en/Industries/Power—Utilities/Plug-in—Cybersecurity. Retrieval date: Oct 30th, 2014. 12

Young, E. . (2014b). Oil and gas cybersecurity - penetration testing techniques. http://www.ey.com/Publication/vwLUAssets/Oil_and_gas_cybersecurity_-_Penetration_testing_techniques/$FILE/EY-O&G_cybersecurity-penetration_testing_techniques.pdf. Retrieval date: Oct 30th, 2014. 12

Yuan, J. and Yu, S. (2013). Privacy Preserving Back-Propagation Neural Network Learning Made Practical with Cloud Computing. *IEEE Transactions on Parallel and Distributed Systems*, 99(1):1–1. 50

Zeigler, B. P., Praehofer, H., and Kim, T. G. (2000). *Theory of modeling and simulation: integrating discrete event and continuous complex dynamic systems*. Academic press. 53

Zhang, J. and Gunter, C. (2010). Application-aware secure multicast for power grid communications. *in Proc. IEEE Int'l Conf. on Smart Grid Communications*, pages 339–344. 98

Zimmerman, R., Murillo-Sanchez, C., and Thomas, R. (2011). Matpower: Steady-state operations, planning, and analysis tools for power systems research and education. *Power Systems, IEEE Transactions on*, 26(1):12–19. 113, 114

Vita

Yue Tong is a PhD student at the Department of Electrical Engineering and Computer Science of the University of Tennessee. His research interest are in the data security and privacy, applied cryptography, and their applications in the cloud computing, electronic health records (EHR) system and power delivery system. He has joined the University of Tennessee working as a teaching and research assistant under the supervision of Dr. Jinyuan Sun since 2011. Prior to that, he obtained an Bachelor of Engineering from Xidian University, Xian, China.