Fall 2015

Quantification of information flow in cyber physical systems

Li Feng

Department: Computer Science

QUANTIFICATION OF INFORMATION FLOW IN CYBER PHYSICAL SYSTEMS

by

LI FENG

A DISSERTATION

Presented to the Faculty of the Graduate School of the

MISSOURI UNIVERSITY OF SCIENCE AND TECHNOLOGY

in Partial Fulfillment of the Requirements for the Degree

DOCTOR OF PHILOSOPHY

in

COMPUTER SCIENCE

2015

Dr. Bruce McMillin, Advisor
Dr. Ali Hurson
Dr. Wei Jiang
Dr. Dan Lin
Dr. Jonathan Kimball

ABSTRACT

In Cyber Physical Systems (CPSs), traditional security mechanisms such as cryptography and access control are not enough to ensure the security of the system since complex interactions between the cyber portion and physical portion happen frequently. In particular, the physical infrastructure is inherently observable; aggregated physical observations can lead to unintended cyber information leakage. Information flow analysis, which aims to control the way information flows among different entities, is better suited for CPSs than the access control security mechanism. However, quantifying information leakage in CPSs can be challenging due to the flow of implicit information between the cyber portion, the physical portion, and the outside world. Within algorithmic theory, the online problem considers inputs that arrive one by one and deals with extracting the algorithmic solution through an advice tape without knowing some parts of the input. This dissertation focuses on statistical methods to quantify information leakage in CPSs due to algorithmic leakages, especially CPSs that allocate constrained resources. The proposed framework is based on the advice tape concept of algorithmically quantifying information leakage and statistical analysis. With aggregated physical observations, the amount of information leakage of the constrained resource due to the cyber algorithm can be quantified through the proposed algorithms. An electric smart grid has been used as an example to develop confidence intervals of information leakage within a real CPS. The characteristic of the physical system, which is represented as an invariant, is also considered and influences the information quantification results. The impact of this work is that it allows the user to express an observer's uncertainty about a secret as a function of the revealed part. Thus, it can be used as an algorithmic design in a CPS to allocate resources while maximizing the uncertainty of the information flow to an observer.

ACKNOWLEDGMENT

I am always feeling so lucky to be able to reach here when I am looking back on my footprints. My sincere and foremost gratitude goes out to my advisor, Dr. Bruce McMillin, for accepting me as his student, for giving me the opportunity to work with him, for supporting me financially, for guiding me with extreme patience and continuous encouraging me.

I am also grateful to Dr. Ali Hurson, Dr. Wei Jiang, Dr. Dan Lin and Dr. Jonathan Kimball for serving as my committee members. Their feedback, comments, and suggestions were always helpful.

I am fortunate to work with so many smart and wonderfully labmates: Thomas Roth, Stephen Jackson, Michael Catanzaro, graduated labmates Ravi Akella and Thoshitha Gamage. They provide me with the best working environment. Especially, Thomas and Stephen, they both inspire me with their creative ideas, insightful thoughts and make excellent contributions to our projects.

I greatly acknowledge that my work was supported by the National Science Foundation under award EEC-0812121, the Future Renewable Electric Energy Delivery and Management Center (FREEDM). Also acknowledged is computer science department and all the staffs for their helps during my stay at Missouri S&T.

Last but not the least, I am extremely thankful to my dear family: my parents, Dezhong Feng and Xiufang Zhang, for their unconditional love and raising me to be the person I want to be; my parents-in-law, Erlie Wang and Congli Li, for their countless sacrifices and support; my husband, Lei for his limitless love and always believing in me; my sister, Juan for supporting me going through the hardships in my lives, and my loving son, Ryan for bring me happiness everyday. Without all of you, this would be meaningless. I love you all!

TABLE OF CONTENTS

Page

ABSTRACT iii

ACKNOWLEDGMENT iv

LIST OF ILLUSTRATIONS viii

LIST OF TABLES x

SECTION

1. INTRODUCTION 1
   1.1. CHALLENGES 1
   1.2. CONTRIBUTIONS 2
   1.3. ORGANIZATION 3
2. RELATED WORK 4
   2.1. INFORMATION FLOW SECURITY PROPERTIES 5
      2.1.1. Noninterference Property 5
      2.1.2. Noninference Property 6
      2.1.3. Nondeducibility Property 6
   2.2. QUANTITATIVE INFORMATION FLOW 6
      2.2.1. Shannon Entropy Based Information Theory 7
      2.2.2. Min-entropy Based Information Theory 8
      2.2.3. Generalization of Min-entropy Leakage 9
3. THE UNCERTAINTY OF THE OBSERVABILITY PROBLEM 12
   3.1. MODEL PROBLEM 12
   3.2. INFORMATION FLOW ANALYSIS THROUGH OBSERVABILITY 16
      3.2.1. External Observer Obtaining One Reading 16
      3.2.2. Multiple External Observers without DGI 17
   3.3. ADVICE COMPLEXITY FOR ONLINE KNAPSACK 22
      3.3.1. Online Problem 22
      3.3.2. Competitive Analysis 22
      3.3.3. Advice Complexity 23
      3.3.4. Online Knapsack with Advice Complexity 24
4. PRELIMINARY EXPERIMENT RESULTS 28
   4.1. INFORMATION LEAKAGE MODEL 28
   4.2. EXPERIMENT RESULTS 32

4.2.1. Case 1: Six Items as Solution to a Knapsack Problem 33
4.2.2. Case 2: Split the Light Items in Case 1 34
4.2.3. Case 3: Split the Heavy Items in Case 1 35
5. THE INFORMATION FLOW QUANTIFICATION FRAMEWORK 39
5.1. CONFIDENCE INTERVAL 39
5.2. FRAMEWORK 40
5.3. CASE STUDY AND EVALUATION RESULTS IN FREEDM 46
5.3.1. Case 1: The Heavy Items are Seen Before the Light Items 47
5.3.2. Case 2: The Light Items are Seen Before the Heavy Items 48
5.3.3. FREEDM Example and Evaluation Result 49
5.4. ALGORITHM 51
5.5. EXAMPLES BY APPLYING THE ALGORITHMS 53
5.5.1. Case 1:Heavy Items and Light Items Satisfy Heavy $> \frac{\texttt{Light}}{\delta}$ 53
5.5.2. Case 2:Heavy Items and Light Items Satisfy Heavy $< \frac{\texttt{Light}}{\delta}$ 54
6. INTEGRATING WITH PHYSICAL SYSTEM 59
6.1. INVARIANT FOR THREE NODES SWITCHED SYSTEM 59
6.2. INVARIANT FOR TWO-GENERATORS SEVEN NODES SYSTEM . 61
6.3. CONFIDENCE INTERVAL ANALYSIS WITH INTEGRATION OF PHYSICAL SYSTEM INVARIANT 63
7. CONCLUSIONS AND FUTURE WORK 70
7.1. CONCLUSIONS AND CONTRIBUTIONS 70
7.2. FUTURE WORK 71
8. ZERO KNOWLEDGE PROOFS 72
8.1. BACKGROUND AND MOTIVATION 72
8.2. INTRODUCTION 73
8.2.1. Interactive Proof and Its Properties 73
8.2.2. A Simple Example 74
8.2.3. Computational Indistinguishability 75
8.2.4. One-way Function 76
8.2.5. Simulation Paradigm 76
8.2.6. Definition of ZKP 77
8.2.7. Witness Indistinguishability 78
8.2.8. Honest Verifier VS General Cheating Verifier 79
8.2.9. Black-box Simulator VS Non-black-box Simulator 79
8.2.10.Quality of ZKPs 81
8.3. NP PROBLEM AND ZKPS 84
8.4. ZKPS APPLICATIONS 85

8.4.1. ZKP for Graph Three-Colorability 86
8.4.2. ZKP for Feige-Fiat-Shamir Identification Scheme 88
8.4.3. ZKP for Graph Isomorphism 90
8.4.4. ZKP for Hamiltonian Cycle [12] 92
8.4.5. Another ZKP for Graph Three-Colorability [12] 94
8.4.6. ZKP Sketch for SAT 95
8.4.7. ZKP for Circuit Computations 98
8.4.8. ZKP for Exact Cover 99
8.4.9. ZKP for 0-1 Knapsack 103
8.5. ADVANCED TOPICS IN COMPOSING ZKPS 106
8.5.1. Composing ZKPs 106
8.5.2. The Richardson-Kilian Concurrent ZKPs Protocol 111
8.5.3. The Improved Concurrent ZKPs Protocol 112
8.5.4. Efficiency Considerations 116
8.5.5. Knowledge Complexity 117
8.5.6. Non-interactive Zero-Knowledge 118
8.6. CONCLUSION 118

BIBLIOGRAPHY 120

VITA 128

LIST OF ILLUSTRATIONS

Figure Page

3.1 Load Balance Algorithm 15

3.2 FREEDM with Attackers A_1, A_2 and A_n 16

3.3 Advice Complexity Model 24

4.1 Advice Tape Modelfor DGI 28

4.2 Uncertainty Portion with 6 items and $\delta = 0.2$ 33

4.3 Uncertainty Portion with 6 items and $\delta = 0.08$ 34

4.4 Uncertainty Portion with 10 Items and $\delta = 0.2$ by Splitting the Light Items 35

4.5 Uncertainty Portion with 10 Items and $\delta = 0.08$ by Splitting the Light Items 36

4.6 Uncertainty Portion with 10 Items and $\delta = 0.08$ by Splitting the Heavy Items 37

5.1 Observation Based Information Leakage Analysis Framework 40

5.2 Confidence Interval with Observations 44

5.3 Observation to Confidence Interval for Case 1 47

5.4 Observation to Confidence Interval for Case 2 48

5.5 Power Migration in FREEDM for the Specific Example 49

5.6 Observation to Confidence Interval in FREEDM 50

5.7 Observation to Maximal Confidence Interval for Case 1 54

5.8 Observation to Maximal Confidence Interval for Case 2 55

6.1 Three Nodes Switched System Topology 60

6.2 Simulated microgrid performance in response to two Power Balance Algorithms. Case (a) indicates instability due to excessive power migrations, whereas in case (b) the red arrow indicates that Power Balance Algorithm finally guards the migration, preventing instability. 61

6.3 Two Generators Seven Nodes System Topology 62

6.4 Observation to Confidence Interval for (a) Line Overloading Case without ATC (b) ATC Invariant Guarded Case 65

6.5 Line Overloading Case Without ATC 66
6.6 ATC Invariant Guarded Case 67
6.7 ATC Invariant Guarded Case with Splitting 68
8.1 Ali Baba's Cave 75
8.2 Simulator 77
8.3 3Colored Graph 86
8.4 ZKP for G3C 87
8.5 ZKP for Feige-Fiat-Shamir Identification Scheme 89
8.6 Graph Isomorphism 90
8.7 ZKP for Graph Isomorphism 91
8.8 Hamiltonian Cycle 92
8.9 ZKP for Hamiltonian Cycle 94
8.10 ZKP for Boolean Computation 97
8.11 ZKP for Circuit Computation 100
8.12 ZKP for Exact Cover 101
8.13 ZKP for 0-1 Simple Knapsack 105
8.14 Sequential Composition of ZKP 108
8.15 Parallel Composition of ZKP 108
8.16 Concurrent Composition of ZKP 111

LIST OF TABLES

Table		Page
3.1	Smart Grid Load Table: Power Management	13
3.2	Load Balance Algorithm for DGI	14
3.3	Advice Tape for Simple Knapsack	27
4.1	Load Table of Node3	29
4.2	First Bit	31
4.3	Information Leakage	32
4.4	Revealed Item and Deduced Solution	38
4.5	Another Example of Load Table	38
5.1	Observations with Confidence Interval for Case 1	47
5.2	Observations with Confidence Interval for Case 2	48
5.3	Observations with Confidence Interval from FREEDM	50
5.4	Observations to Maximal Confidence Interval for Case 1	58
5.5	Observations to Maximal Confidence Interval for Case 2	58
6.1	Load Balance Algorithm for DGI with Invariant Check	63
6.2	Results of Reduced Confidence Interval due to ATC	65
8.1	Black-Boxes for Index and Product Operation	103
8.2	Protocol Not Closed Under Parallel Composition	109
8.3	Parallel Composition Leads to Information Leakage	109
8.4	RK Protocol	112
8.5	PRS Protocol	113
8.6	Basic ZKP for 0-1 Simple Knapsack	114
8.7	PRS Concurrent ZKP for 0-1 Simple Knapsack	115
8.8	$2k^2$ Randomly Generated Strings	115
8.9	Constraints on Strings	115
8.10	Preamble Messages	116

1. INTRODUCTION

Cyber Physical Systems (CPSs) consist of both distributed computations (as a cyber portion) and physical entities that provide better utilization, control, and performance [35] to the overall system. These systems operates by transmitting information between the cyber components and the physical entities. Examples of CPSs include water and gas distribution systems, power generation and distribution systems, transportation systems, and advanced communication systems [69].

In addition to the extensive functional capabilities provided by CPSs, they pose a series of new security challenges due to frequent complex interactions between the cyber portion and physical portion. For example, failures and attacks occur on the physical infrastructure, cyber attacks occur on the data management and communication protocol, and aggregated physical manifestations lead to unintended information leaks. Information leakage can have significant consequences, ranging from violation of confidentiality of process decision making [35] to loss of privacy, if the resource is shared. Indirect information transmissions usually happen in covert channels [77] that are not easy to detect.

Information flow properties, aiming at controlling the way information flows among different entities, are more suitable for CPSs. Generally, they will generate a binary solution for the system to indicate whether there is any information leakage or not. On the other hand, it is quite common for a system to leak a small amount of information through observability. The upcoming question is to quantify the information leakage based on physical observations. In a resource-constrained CPS (one in which a finite set of resources are allocated by the CPS to itself), these aggregated observations from the physical system form incomplete information about sensitive information, including the amount of the resource and the allocation. Different subsets of observations give different levels of reduction to the uncertainty of sensitive information.

1.1. CHALLENGES

The confidentiality violation describes unintended and unwanted information disclosures between two domain user groups, namely, a High-level user and a Low-

level user. A High-level user usually has a secret to preserve that a Low-level user tries to obtain. Security considerations within a CPS include the interactions that occur between cyber events and physically observable behaviors [65]. Different levels of confidentiality violations are defined by the observer so that the information flow can be analyzed completely. An observer who is outside a CPS can partially deduce some information when exposed to an observable physical flow. An observer who is inside a CPS, however, knows more about the system as the decisions in the cyber domain are manifested on the physical domain. Furthermore, this observer can inflict greater damage because he can manipulate the cyber processes.

Systems, in general, are not intended to be fully secure. A small amount of information leakage is sometimes necessary. For example, a password checking system provides a message when a password is entered incorrectly. Quantifying the system's information leakage can, however, be challenging. The information theory is often used to calculate the leakage of secret information received by a program that produces a public output [82], [81], [14], [61], [59], [20], [58].

A number of factors need to be considered when attempting to secure a CPS. For example, the distribution of the secret information and properties of the problem aid in revealing additional information. Consider a password that is stored in a k bit variable. If it is assumed that a uniform probability distribution exists for the variable, uncovering the password in a single check is quite difficult because the probability for guessing correctly is $\frac{1}{2^k}$. If the distribution of possible values is less uniform (some values are more significant than others), uncovering the password in a single check becomes much easier. Quantifying information leakage in CPSs can, however, be challenging because information leakage can happen explicitly or implicitly in the cyber portion and the physical portion.

1.2. CONTRIBUTIONS

Inspired by differential privacy [24], a collection of multiple, observable, physical changes or states could help with the interpretation of sensitive cyber information. This dissertation focuses on quantifying the information flow in CPSs from a security perspective, especially CPSs that allocate constrained resources.

Theoretically, a preliminary model for observer to quantify the information leakage that uses an advice complexity analysis has been provided, and a method that

can quantify the observer's uncertainty about a secret has been illustrated. A general framework is proposed, and algorithms are illustrated to obtain the quantified information. Both are based on the advice tape concept of algorithmically quantifying information and statistical analysis. An electric smart grid has been used as an example to develop information leakage quantification within a real CPS. The characteristic of the physical system, which is represented as an invariant, is also considered and integrated in the result of the information quantification.

1.3. ORGANIZATION

This dissertation is organized as follows:

Section 2 presents backgrounds of security and related work on quantifying information flow.

Section 3 introduces the problem statement as well as preliminary results analyzing information leakage.

Section 4 discusses the experimental evidence gathered for the verification of the preliminary result.

Section 5 presents a framework to analyze the information flow quantification and the algorithms, in detail for the calculating leaked information with the increased number of observations; results from case studies in a real CPS show the preference of the specific distribution.

Section 6 illustrates the integration method of the proposed framework and algorithm with the physical system by means of an invariant approach. Through considering the characteristics of the physical system as an invariant, the cyber algorithm can be updated to generate more uncertainty to the observer in a real CPS.

Section 7 is the conclusion as well as the proposal for the future work.

Section 8 is a survey on Zero Knowledge Proof.

2. RELATED WORK

Security has three principal categories: confidentiality, integrity and availability. Confidentiality prohibits unauthorized reading information, availability prohibits the unauthorized withholding of information and integrity prohibits the unauthorized modification of information.

Standard security mechanisms include access control, firewalls, antivirus software and encryption. Access control is an important part of the current security infrastructure. It tries to define different security levels in the system. A High-level user usually has a secret; while a Low-level user tries to acquire the secret. Access control imposes restrictions to prevent secret information sending from the High-lever user to the Low-level user. It was first formulated by Lampson and later refined by Graham and Denning [62]. This model describes each individual machine in a system. Each state in the system is a triple (S, O, M), where S is a set of subjects, O is a set of objects. M is an access matrix which has one row for each subject and one column for each object. A cell in M matrix contains the access right the subject has for an object. Some famous traditional security model such as Bell-La Padula (BLP) Model [10], Biba Model [11], Take-Grant [60] and HRU Model [47] apply an access control based approach. They usually employ subject, object, rights and an access control matrix and enforce security by specifying a particular rule.

However, it is not trivial to determine a system's subjects, objects, read access and write access from all aspects. Sometimes it is difficult to map an access control model's primitive to a computer system. And access control does not control how the data is used after it is got read access. For example, consider a Trojan Horse, a malicious program performing an internal attack on the system. It is insufficient to stop the action of a Trojan Horse through an access control model. The malicious user could easily bypass the access restriction by sending out a Trojan Horse program to obtain the secret.

A firewall protects secret information by preventing its communication with outside. Antivirus softwares use some known patterns to detect malicious behavior in the software. Encryption makes sure that only the communication end could access the secret. However, those mechanisms do not provide end-to-end security concerning

information flow inside a system [77]. For example, antivirus softwares offer very limited protection to new attacks and firewalls will permit some communication in both directions if it is outside the scope of firewall mechanism. They are not adequate to protect the security of the CPSs [69], neither.

Indirect information transmissions usually happen in covert channels in [77] that are not easy to detect and usually reveal information from a High-level user to a Low-level user. Information flow properties aim at controlling the way information flows among different entities. The information flow based security is better suited for the CPSs that has inherently observable changes between the cyber portion and the physical portion. Thus, besides ensuring the accesses of subjects to objects, information flow rules have to be applied for the purpose of the confidentiality. Some of the most prominent information flow properties are Noninterference, Noninference and Nondeducibility.

2.1. INFORMATION FLOW SECURITY PROPERTIES

2.1.1. Noninterference Property. Noninterference was first proposed by Goguen and Meseguer in [36], which attempted to capture the idea that a user in a High-level should not interfere with a user in Low-level. Usually, the method for showing noninterference holds is to demonstrate the Low-level user cannot observe any differences between two executions that is different only in High-level inputs. But noninterference is too strong to be satisfied in reality. Due to highly integrated cyber and physical interactions in the CPSs, it is inevitable for the CPSs to leak information [67].

The idea of indistinguishability is used to describe that there is no information flow between two objects A and B [72]. Noninterference can be expressed that for any pair of behaviors of the system that differ only in A's behavior, B's observations cannot distinguish these two behaviors. Approximate noninterference in [72] is suggested to relax noninterference within a threshold ϵ. It replaces notion of indistinguishability with the idea of similarity and allows for some exactly quantified leakage of information. The quantity measurement for the similarity is to calculate the supremum norm over the difference of the probability distributions representing different observations.

In [34], Bisimulation Non Deducibility on Compositions (BNDC) is proposed with the same idea of noninterference. In short, a system is BNDC if what a Low-

level user sees of the system is not modified by composing any High-level process to the system. Security Process Algebra (SPA) [76] is used to the express information flow of the system and protocol.

2.1.2. Noninference Property. Noninference is equivalent to Noninterference for deterministic system when the High-level output cannot be generated without the High-level input [64]. Unlike noninterference, noninference is more general and can be applied to nondeterministic systems.

2.1.3. Nondeducibility Property. Sutherland's nondeducibility was based on the idea that security is maintained if High-level and Low-level states are independent in the sense that a Low-level user can never totally rule out any High-level state based on his own local state [21].

In cryptography, Shannon's probabilistic definition of secrecy is similar to Sutherland's nondeducibility. It requires High-level and Low-level events to be probabilistically independent. In another words, the Low-level's posterior probability of a High-level event should be the same as his prior probability of that event before interacting with the system [21].

Nondeducibility on strategies was first introduced by [85] to deal with the problem of feedback. It observes that if a program is run multiple times and feedback between runs is allowed to be seen, information can be leaked. In general, if the system has the property of nondeducibility on strategies, there are no noiseless communication channels between High-level input and Low-level output.

2.2. QUANTITATIVE INFORMATION FLOW

Because it is often not possible to achieve noninterference and some information needs to be revealed, quantitative information flow is another direction [19]. There are some fields that apply the theory of quantitative information flow to check the confidentiality of sensitive information. In secure information flow analysis, the program will be examined to see if a secret input is leaked when an attacker sees some outputs of the program. In anonymity protocols, network traffic is checked to see if an attacker could know who is communicating [81]. Side-channel attacks try to break cryptography by taking advantage of information from an algorithm's physical execution such as running time, cache behavior, power consumption and electromagnetic radiation [59] [58]. Those information could be used to recover a secret from cryp-

tographic algorithms. Using the process algebra CSP, information flow quantity has been defined as the number of behaviors of High-level user that are distinguishable from a Low-level user's point of view [61].

2.2.1. Shannon Entropy Based Information Theory. The foundations of quantitative information are information theoretic measures such as Shannon entropy and mutual information. The basic scenario is a program or a protocol that receives some High-level secret inputs and produces some public Low-level outputs. The attack model is what an adversary will deduce about secret inputs if he sees the public outputs.

Let X be a random variable whose value belongs to a set χ. The Shannon entropy $SE(X)$ is the expected number of bits required to transmit X optimally and defined as the uncertainty about X:

$$SE(X) = \sum_{x \in \chi} P[X = x] log \frac{1}{P[X = x]} \tag{1}$$

Given two jointly distributed random variables X and Y, the conditional entropy $SE(X|Y)$ is defined as the uncertainty about X given Y:

$$SE(X|Y) = \sum_{y \in Y} P[Y = y] SE(X|Y = y) \tag{2}$$

where

$$SE(X|Y = y) = \sum_{x \in \chi} P[X = x|Y = y] log \frac{1}{P[X = x|Y = y]} \tag{3}$$

The mutual information $I(X;Y)$ is defined as the amount of information shared between X and Y:

$$I(X;Y) = SE(X) - SE(X|Y) \tag{4}$$

In order to quantify information leakage, there is a statement describing:

Initial uncertainty = information leakage + remaining uncertainty

If we define H as secret input and L as public output, the entropy $SE(\mathrm{H})$ is the initial uncertainty about H and the conditional entropy $SE(\mathrm{H}|L)$ is the remaining

uncertainty about H. After rearranging the statement, Information leakage = initial uncertainty - remaining uncertainty = $SE(\mathrm{H})$ - $SE(\mathrm{H}|L)$.

2.2.2. Min-entropy Based Information Theory. However, the Shannon Entropy based method doesn't support good security guarantees about the probability that H could be guessed from an example [81]. Based on Rényi's min-entropy, a number of researchers have developed a min-entropy theory to quantify information flow.

Min-entropy is based on a concept of vulnerability. Let X be a random variable whose value belongs to a set χ. The vulnerability of X is the worst-case probability that an adversary could guess the value of X in one try and is denoted as $V(X)$:

$$V(X) = max_{x \in \chi} P[X = x] \tag{5}$$

The min-entropy of X is proposed as measure of initial uncertainty and denoted as $SE_\infty(X)$:

$$SE_\infty(X) = log \frac{1}{V(X)} \tag{6}$$

The conditional vulnerability $V(X|Y)$ is calculated given jointly distributed random variables X and Y:

$$V(X|Y) = \sum_{y \in Y} P[Y = y] V(X|Y = y) \tag{7}$$

where

$$V(X|Y = y) = max_{x \in \chi} P[X = x|Y = y] \tag{8}$$

The conditional min-entropy $SE_\infty(X|Y)$ is proposed as the measure of remaining uncertainty:

$$SE_\infty(X|Y) = log \frac{1}{V(X|Y)} \tag{9}$$

The min-entropy information leaked is calculated based on above definition: Information leakage = initial uncertainty - remaining uncertainty = $SE_\infty(X)$ - $SE_\infty(X|Y)$.

There is another important concept in information theory named channel: a channel represents relationships between systems inputs and outputs [82]. A channel is a triple (S, O, C_{SO}), where S is a finite set of secret input values, O is a finite set of observable output values and C_{SO} is an $|S| \times |O|$ matrix. C_{SO} is called the channel matrix where each cell $C_{SO}[s, o]$ represents the conditional probability of obtaining output o given that the input is s.

The attack model is that an adversary A tries to find out the value of S through observing O. In order to quantify the amount of information that flows from S to O, it is natural to make a comparison of A's uncertainty about S before and after seeing the value of O. That is:

Information leakage = initial uncertainty - remaining uncertainty

The min-entropy leakage from S to O is denoted as $Leakage_{SO}$:

$$Leakage_{SO} = SE_{\infty}(S) - SE_{\infty}(S|O) \tag{10}$$

The above approaches try to measure information flow through the reduction in uncertainty about secret data. But this uncertainty-based measurement is inadequate because accuracy isn't taken into account [20]. The attacker's probability distribution over the secret is subjective and is regarded as a belief. Belief will change due to the attacker's observations of the execution of a program. This new metric measurement for quantitative information flow is developed and applied in several examples.

2.2.3. Generalization of Min-entropy Leakage. The min-entropy leakage measures the amount of information by which a channel increases the vulnerability of a secret being guessed correctly in one try by an adversary. However, in reality, there are some other scenarios that need considering. For example, what if the adversary could make multiple guesses about the secret? Or what if the adversary guesses the secret partially or approximately? In [17], a gain function g is proposed to model the benefits that the adversary could gain by guessing a value close to the secret, guessing a part of the secret, guessing a property of the secret, or guessing the secret within some number of tries.

Assuming the adversary makes a guess w when the secret's actual value is x, then the gain function $g(w, x)$ will model the benefits that the adversary gains from this guess, ranging from 0 to 1. Given a gain function g, g-vulnerability is defined as

the maximum expected gain over all guesses. The $g-leakage$ is the generalization of min-entropy leakage based on g-vulnerability.

Given a set χ of possible secret values and a finite, nonempty set W of allowable guesses, a gain function is a function g: $W \times \chi \to [0,1]$. Usually, a gain function g can be represented as a $|W| \times |\chi|$ matrix G, where $G[w,x] = g(w,x)$.

The prior g-vulnerability is derived from the definition of vulnerability to take account of the gain function g and a prior distribution π on χ:

$$V_g(\pi) = max_{w \in W} \sum_{x \in \chi} \pi[x] g(w,x) \tag{11}$$

The posterior g-vulnerability is based on gain function g, prior π and channel C:

$$V_g(\pi, C) = \sum_{y \in Y} max_{w \in W} \pi[x] C[x,y] g(w,x) \tag{12}$$

The g-entropy and g-leakage are defined in the same way as min-entropy and min-leakage:

$$SE_g(\pi) = log \frac{1}{V_g(\pi)} \tag{13}$$

$$SE_g(\pi, C) = log \frac{1}{v_g(\pi, C)} \tag{14}$$

$$Leakage_g(\pi, C) = SE_g(\pi) - SE_g(\pi, C) = log \frac{V_g(\pi, C)}{V_g(\pi)} \tag{15}$$

In [17], gain functions can be induced from metrics and distance functions. The notion of distance measures the difference between the secrets in χ. The metric d on χ, is a function: $d : \chi \times \chi \to [0, \infty)$, satisfying the properties:

- Identity: $d(x_1, x_2) = 0 \ iff \ x_1 = x_2$
- Symmetry: $d(x_1, x_2) = d(x_2, x_1)$
- Triangle inequality: $d(x_1, x_3) \leq d(x_1, x_2) + d(x_2, x_3)$

The normalized metric $\overline{d}$, for example, is dividing all distances by the maximum value of d. Even binary gain functions could be in various formula by considering different choices of W. For example, $W = \{W, \chi | W\}$ is called a 2-block gain function by

considering W as a property that the secret χ might satisfy W or might not satisfy $\chi|W$. Actually W could be any partition of χ into one or more disjoint blocks, the attacker just needs to determine which block the secret belongs to. The partition gain function is derived from dividing W into more than two blocks.

In [14], quantifying information flow is extended to use views, which in fact are partitions of the secrets from the attacker's perspective. Give a view and n independent observations of the system, the probability that an attacker wrongly predicts the secret is plotted. It shows that behavior of this error probability is a function of the number of observations. In particular, as $n \to \infty$, the error probability decreases exponentially.

However, as the Shannon entropy only considers whole secret as input, it is not appropriate for quantifying information leakage to the observability problem that considers incremental observations of the secret [31]. Moreover, information theory based methods do not address all aspects of the CPSs' system background, in particular they do not address the algorithmic aspects of CPS computation as well as the physical characteristics. In a CPS, implicit information will happen in covert channel between the interactions of the cyber component and physical component and aggregation of physical observations would also leak certain amount of information in the cyber algorithm. As a result, in a CPSs where heavy interactions happen between the cyber part and the physical part, the information quantification method should be able to be adjusted to be useful.

In [31], a method has been proposed to quantify information flow in a smart grid applying advice complexity analysis for an abstract algorithm. In this dissertation, a more general way has been proposed to quantify information flow in CPSs with constrained resources. The algorithms are developed to quantify the information and results are verified in the smart grid.

3. THE UNCERTAINTY OF THE OBSERVABILITY PROBLEM

All the papers in the previous section try to quantify what the attacker will learn about a secret through the output of the system based on information theory. They only consider a secret as a whole that has a public distribution but not the incremental observations of the secret as the input. What this dissertation focuses on is if an observer has already seen a part of secret, how much does this part of secret reveal the remaining part? Based on the revealed information, how to decide the remaining information? There must be given some relationship between the revealed one with the unrevealed one. If this relationship between the revealed part with the unrevealed one is given by the observable physical system, it is possible to quantify information flow. Exploring that relationship is the key to quantify information leakage. In particular, as the observer sees more and more pieces of information, he will be able to quantify the information leakage based on different relationships that are manifested from the revealed part.

3.1. MODEL PROBLEM

FREEDM is a smart grid that uses intelligent transmission and distribution networks to deliver electricity. Numerous vulnerabilities and challenges will arise as it involves many factors such as the cyber part, the physical part, integration of cyber and physical systems, and human behavior and commercial interests [57]. For example, as the smart grid uses supervisory control and data acquisition (SCADA) systems to obtain fine grained monitor data, the protection of electricity usage patterns for the customers should be considered.

Distributed Grid Intelligence (DGI) is a collection of cyber processes in the smart grid to perform intelligent and distributed computation and management. They implement balancing power flow and optimal distribution of energy through a distributed load balancing algorithm [3]. Each node in the smart grid consists of a Solid State Transformer (SST), a Distributed Renewable Energy Resource (DRER), a Distributed Energy Storage Device (DESD), and a LOAD which is the consumption of power at the household. Each node computes its local SST that represents the amount of excess or deficit generation. That is $SST = LOAD - DRER$.

DGIs use the distributed load balancing algorithm in [3] to manage power in the smart grid. Nodes in the system first form a group with a group leader based on a leader election algorithm. In the load balancing algorithm, each node in the smart grid will receive a Normal value from the group leader. The node will be classified as Supply if its SST's actual value is larger than the Normal value and as Demand if its SST's actual value is less than the Normal value. The nodes participating in the load balancing algorithm communicate their SST changes with each other in order to migrate power from a Supply node to a Demand node. The nodes normalize their SST values and achieve a roughly balanced status after power migrations.

Each DGI maintains a Load Table as shown in Table 3.1, which stores information of its local grid organization and information from other nodes in the system. A node in the Supply state will advertise a draft request message to the Demand nodes in its load table. On receiving the draft request from Supply node, the Demand node responds with a message called draft age indicating request power. The Supply node will select one Demand node based on the greedy algorithm to fractional knapsack problem [3] and send out a draft select message to the Demand node. Upon receiving the draft select message, the Demand node will initiate power migration by increasing its local SST and send back an accept message to the Supply node. Then the Supply node will finish the power migration by decreasing its local SST. The algorithm continues until all the nodes are in the Normal state. The following summarizes the steps for power migration in the DGI and Figure 3.1 shows a sequence diagram of the load balancing algorithm. The communications protocol between the Supply node and the Demand node is shown in Table 3.2 and continues until all the nodes are in the Normal state.

Table 3.1: Smart Grid Load Table: Power Management

Net DRER: __	Net DESD: __
Net LOAD: __	SST Gateway: __
Normal: __	
Node	**State**
Node 1	Demand
Node 2	Supply
...	...
Node n	Normal

Table 3.2: Load Balance Algorithm for DGI

1. A node in the Supply state will advertise a draft request message to the Demand nodes in its load table. 2. On receiving the draft request from Supply node, the Demand node responds with a message called draft age indicating requested power. 3. The Supply node will select one Demand node based on the greedy solution to the knapsack problem and send out a draft select message to the Demand node. 4. Upon receiving the draft select message, the Demand node will initiate power migration by increasing its local SST and send back an accept message to the Supply node. 5. Then the Supply node will finish the power migration by decreasing its local SST.

Definition 3.1 (Knapsack problem)**.** *There are n kinds of items, 1 through n. Each item i has a value v_i and a weight w_i. The maximum weight that a bag can carry is W. Mathematically the knapsack problem can be formulated as:*

- *maximize* $\sum_{i=1}^{n} v_i x_i$
- *subject to* $\sum_{i=1}^{n} w_i x_i \leq W$

The 0-1 knapsack problem requires each object must be either in the knapsack completely or not at all, that is $x_i \in \{0, 1\}$. The 0-1 knapsack problem is NP-hard [55] and can be solved by the dynamic programming. The fractional knapsack would allow partial usage of each object. The greedy algorithm is used to find an optimal solution to the fractional knapsack problem. There is a greedy approach for the 0-1 knapsack problem that can obtain an approximation solution to the optimal solution within polynomial-time. The greedy approach will choose the items with better ratio of value to weight and insert into the knapsack until it cannot fit any more whole items. Next, the remaining items will be divided into smaller pieces. As the pieces become infinitesimally small, the result approaches the optimal solution. If the size of pieces is limited, the result is guaranteed to be within a bound of the optimal solution [3].

The subset-sum problem is a special case with $w_i = v_i$. It means choosing a subset of the values so that the sum is as large as possible without exceeding a bound. The subset-sum problem with each item's value less than 1 is called the simple knapsack problem in this paper.

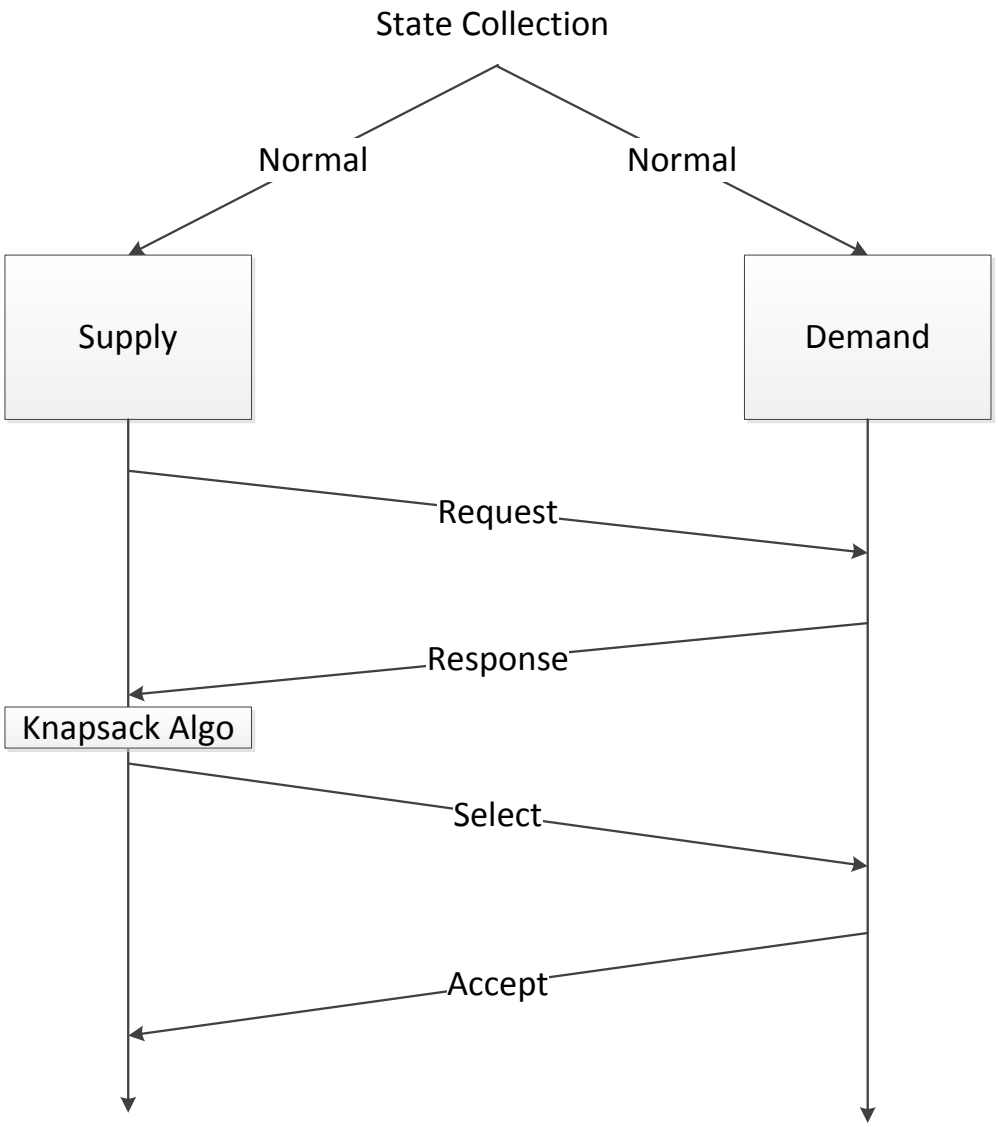


Figure 3.1: Load Balance Algorithm

The secret in our problem is the constraint and solution of the knapsack problem, which is applied in a load balancing algorithm to manage power in smart grid [3]. The implicit information flow between the cyber part and the physical part, particularly the aggregated physical observation under the process of a cyber algorithm, is the focus of this dissertation. As shown in Figure 3.2, the observer could be an outsider A_n or

be inside of the system as A_1 or A_2. What will be an outsider A_n or an insider A_1 will deduce about the secret in the system? Moreover, what will be the communication between an insider A_1 and A_2 help them deduce about the secret in the system?

This section will analyze the information leakage based on the observer's position to the system and will especially focus on the situation that observers would communicate with each other. How does this confidentiality violation influence the whole system? In particular, as more and more observed physical information is collected, specific attention will try to quantify the information leakage of the secret.

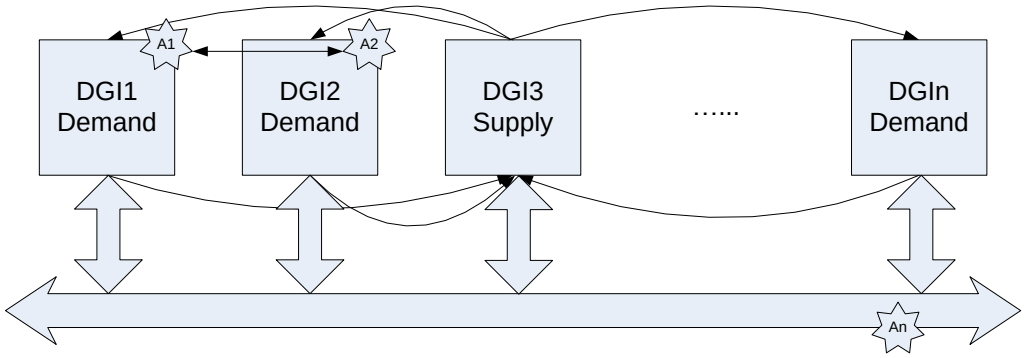


Figure 3.2: FREEDM with Attackers A_1, A_2 and A_n

3.2. INFORMATION FLOW ANALYSIS THROUGH OBSERVABILITY

3.2.1. External Observer Obtaining One Reading. If no observer can obtain the system's readings, the system has zero knowledge leakage. But electric power lines are physically visible and public; an observer could obtain a reading on one line attached to a node. The information will be the power value transmitted to the node. The system, thus, suffers from information leakage.

Suppose there are n demand values $x_1,\ x_2,\ ...,\ x_n$; the number of possible supply values will be 2^n: $X_1,\ X_2,\ ...,\ X_{2^n}$. Assume a supply value is chosen under a uniform

distribution, the probability for each value to be the supply value would be $\frac{1}{2^n}$. The Shannon Entropy of the system is:

$$SE(X) = \sum_{i=1}^{2^n} P(X) log \frac{1}{P(X)} = \sum_{i=1}^{2^n} \frac{1}{2^n} log 2^n = n \tag{16}$$

If only one value x_i is observed to the observer through the power line, the number of possible values for the supply value is 2^{n-1}. Because X is under a normal distribution, $SE(X|x_i)$ can be obtained according to Shannon Entropy:

$$SE(X|x_i) = \sum_{i=1}^{2^{n-1}} \frac{1}{2^{n-1}} log 2^{n-1} = n - 1 \tag{17}$$

The information leakage is calculated as

$$I = SE(X) - SE(X|x_i) = n - (n-1) = 1 \tag{18}$$

3.2.2. Multiple External Observers without DGI. More observers can obtain more readings on the shared power bus. If two values x_i and x_j are revealed to the observer, the number of possible values for the supply value is 2^{n-2}. Assuming X is under a normal distribution, $SE(X|x_i,\ x_j)$ can be obtained according to Shannon's Entropy theory:

$$SE(X|x_i,\ x_j) = \sum_{i=1}^{2^{n-2}} \frac{1}{2^{n-2}} log 2^{n-2} = n - 2 \tag{19}$$

The information leakage is calculated as

$$I = SE(X) - SE(X|x_i,\ x_j) = n - (n-2) = 2 \tag{20}$$

As the values are given to the observer one-by-one, the number of possible values for the supply value is changed from 2^n, 2^{n-1}, 2^{n-2}, until all of the values are revealed. The information leakage increases geometrically as each value is revealed.

Given the assumption that the node has an integer value and each value is identical, two observations of information leakage and explanations are provided. Suppose the total supply value is S, the number of demand nodes in the system is n, one

demand node is observed with value A, and that x is the node with maximum value among the remaining $n-1$ nodes. Since the total number of demand nodes in the system is less than $n-1$, the following inequality must be satisfied.

$$x + (x-1) + (x-2) + ... + (x-(n-2)) > S - A$$

$$(n-1)x - (1+2+...+(n-2)) > S - A$$
$$(n-1)x > S - A + \frac{(n-1)(n-2)}{2}$$
$$x > \frac{S-A}{n-1} + \frac{n-2}{2}$$

Thus,

$$\frac{S-A}{n-1} + \frac{n-2}{2} < x < S - A \tag{21}$$

Similarly, assuming y is the node with a minimum value among the remaining $n-1$ nodes, the following inequality should be satisfied.

$$y + (y+1) + (y+2) + ... + (y+(n-2)) < S - A$$

$$(n-1)y + (1+2+...+(n-2)) < S - A$$
$$(n-1)y < S - A - \frac{(n-1)(n-2)}{2}$$
$$y < \frac{S-A}{n-1} - \frac{n-2}{2}$$

Thus,

$$0 < y < \frac{S-A}{n-1} - \frac{n-2}{2} \tag{22}$$

Observation 3.1. *An observed larger value as a part of solution to the 0-1 knapsack problem increases the certainty of the maximum value and minimum value in the remaining solution.*

Suppose a node with value A is larger than a node with value B. x_A and x_B are maximum value in the remaining solution if A and B are observed separately.

$$\frac{S-A}{n-1}+\frac{n-2}{2}<x_A<S-A$$

$$\frac{S-B}{n-1}+\frac{n-2}{2}<x_B<S-B$$

The $range(x)$ is defined as the upper bound of x minus the lower bound of x.

$$range(x_A)=(S-A)-(\frac{S-A}{n-1}+\frac{n-2}{2})$$

$$range(x_B)=(S-B)-(\frac{S-B}{n-1}+\frac{n-2}{2})$$

$$range(x_A)-range(x_B)=(B-A)\frac{n-2}{n-1}<0$$

Because $range(x_A)$ is less than $range(x_B)$, the observer will have more certainty on the maximum value in the remaining solution if A is observed.

Similarly, y_A and y_B are minimum nodes in the remaining solution if A and B are observed separately.

$$0<y_A<\frac{S-A}{n-1}-\frac{n-2}{2}$$

$$range(y_A)=\frac{S-A}{n-1}-\frac{n-2}{2}$$

$$0<y_B<\frac{S-B}{n-1}-\frac{n-2}{2}$$

$$range(y_B)=\frac{S-B}{n-1}-\frac{n-2}{2}$$

$$range(y_A)-range(y_B)=\frac{B-A}{n-1}<0$$

Because $range(x_A)$ is less than $range(x_B)$, the observer will have more certainty on the minimum value in the remaining solution if A is observed.

Observation 3.2. *As more values are revealed as a part of solution to the 0-1 knapsack problem, the observer will have more certainty on the range of the maximum value in the remaining solution.*

Suppose the first node is observed as value A and x is the one with maximum value among the remaining $n-1$ nodes.

$$\frac{S-A}{n-1}+\frac{n-2}{2}<x<S-A$$

If another node is observed as value $A^{'}$ and $x^{'}$ is the one with maximum value among the remaining $n-2$ nodes.

$$\frac{X-A-A^{'}}{n-2}+\frac{n-3}{2}<x^{'}<X-A-A^{'}$$

Both $range(x)$ and $range(x^{'})$ can now be obtained as follows:

$$range(x)=(S-A)-(\frac{S-A}{n-1}+\frac{n-2}{2})$$

$$range(x^{'})=(S-A-A^{'})-(\frac{S-A-A^{'}}{n-2}+\frac{n-3}{2})$$

The difference between $range(x)$ and $range(x^{'})$ is calculated as

$$(\frac{(S-A)(n-2)}{n-1}-\frac{n-1}{2})-(\frac{(S-A-A^{'})(n-3)}{n-2}-\frac{n-2}{2})$$

$$=(S-A)\frac{(n-2)^2-(n-1)(n-3)}{(n-1)(n-2)}+A^{'}\frac{n-3}{n-2}-\frac{1}{2}$$

$$=\frac{(S-A)}{(n-1)(n-2)}+A^{'}\frac{n-3}{n-2}-\frac{1}{2}>0$$

A similar result on the minimum value in the remaining solution cannot, however, be obtained. Assume that y and $y^{'}$ are minimum values among the remaining $n-1$ and $n-2$ nodes, respectively.

$$0<y<\frac{S-A}{n-1}-\frac{n-2}{2}$$

$$0 < y^{'} = \frac{S - A - A^{'}}{n-2} - \frac{n-3}{2}$$

The difference between $range(y)$ and $range(y^{'})$ is calculated as

$$(\frac{S-A}{n-1} - \frac{n-2}{2}) - (\frac{S-A-A^{'}}{n-2} - \frac{n-3}{2})$$

$$= \frac{A^{'}(n-1) - (X-A)}{(n-1)(n-2)} + \frac{1}{2}$$

The difference would be either positive or negative; the range of the minimum value among the remaining nodes may not decrease.

Theorem 3.1. *Assuming w is the constraint for the 0-1 knapsack problem with n items, when value A_i for $i \in \{1, 2, \cdots, t\}$ is observed in the solution that contains t items, the lower bound of the maximum value in the remaining solution becomes*

$$\frac{w - \sum A_i}{n-t} + \frac{n-t-1}{2}$$

the upper bound of maximum value in the remaining solution is $w - \sum A_i$; the upper bound of minimum value in the remaining solution is

$$\frac{w - \sum A_i}{n-t} - \frac{n-t-1}{2}$$

Proof. Assume x is the node with maximum value among the remaining $n-t$ solutions, y is the node with minimum value among the remaining $n-t$ nodes. Because the total number of demand nodes in the system is less than $n-t$, the following inequality must be satisfied.

$$x + (x-1) + (x-2) + ... + (x - (n-t-1)) > w - \sum A_i$$

$$(n-t)x - (1 + 2 + ... + (n-t-1)) > w - \sum A_i$$

$$(n-t)x > w - \sum A_i + \frac{(n-t)(n-t-1)}{2}$$

$$x > \frac{w - \sum A_i}{n-t} + \frac{n-t-1}{2}$$

$$y + (y+1) + (y+2) + ... + (y + (n-t-1)) < w - \sum A_i$$

$$(n-t)y + (1 + 2 + ... + (n-t-1)) < w - \sum A_i$$

$$(n-t)y < w - \sum A_i - \frac{(n-t)(n-t-1)}{2}$$

$$y < \frac{w - \sum A_i}{n-t} - \frac{n-t-1}{2}$$

The range of maximum value is: $w - \sum A_i > x > \frac{w-\sum A_i}{n-t} + \frac{n-t-1}{2}$. The range of minimum value is: $y < \frac{w-\sum A_i}{n-t} - \frac{n-t-1}{2}$. □

The above observations and theorem emperically indicate that the observer will deduce more information in the remaining solution (e.g., the lower bound of the maximum value and the upper bound of the minimum value) as the observer obtains more readings from the nodes. These larger readings will provide more certainty on the lower bound of the maximum value and the upper bound of the minimum value in the remaining solution than the smaller readings will provide.

3.3. ADVICE COMPLEXITY FOR ONLINE KNAPSACK

3.3.1. Online Problem. The online problem has input that arrives one by one and deals with extracting the solution without knowing some parts of input. The respective algorithms are called online algorithms. Unlike offline algorithms, in which the input is known to the algorithm at the beginning of the computation, the input will be received piece by piece in online algorithms. After receiving one piece of the input, the algorithm has to make a decision on the solution and cannot reverse the decision in the later phase. The algorithm for the online problem tries to generate a solution with zero knowledge about some missing instances in the input. Many problems such as routing, scheduling, or the paging problem are online problems [13].

3.3.2. Competitive Analysis. Competitive analysis is the standard method to measure the quality of an online algorithm. The competitive ratio introduced by Sleator and Tarjan in [80] is defined as the quality of the solution by the online algorithm over the quality of an offline algorithm. For an online problem P, let's define that an optimal solution for a certain input I of P is $OPT(I)$. $A(I)$ denotes

the solution computed by an online algorithm on input I in a sequential order. For an online minimization problem, A is called $c-$*competitive* if there exists some constant $\alpha > 0$ such that, for any input sequence I, $A(I) \leq c \cdot OPT(I) + \alpha$. For an online maximization problem, A is called $c-$*competitive* if $A(I) \geq \frac{1}{c} \cdot OPT(I) - \alpha$ for some constant $\alpha > 0$. Usually, we consider *strictly* $c-$*competitive* in which $\alpha = 0$.

Offline algorithms have more power than online algorithms. But online algorithms have received more and more interest recently because they try to quantify how much information about the forthcoming part of the unknown input is needed. Researchers try to investigate what amount of information online algorithm really lacks to obtain an optimal solution or achieve a specific competitive ratio [48].

3.3.3. Advice Complexity. It seems unfair to compare the offline algorithms with the online algorithms since the offline algorithms have already known the whole input. In order to investigate to what extent the solution quality of online algorithm can be improved, the notion of advice complexity is introduced [22]. Advice complexity of online problem is another quality measurement for the online problem by extracting a given amount of information about the input. It is proposed as one of methods to achieve a specific competitive ratio for the online problems [23].

In the model of advice complexity presented in Figure 3.3, an Oracle, which sees the whole input and has unlimited computational power, would communicate with the online algorithm passively. The oracle will write bitwise information about the forthcoming input onto an advice tape, which will be read by the online algorithm in a sequential order before reading any input $x_1, x_2, \cdots, x_n$. The competitive ratio will be a result from the online algorithm computation over the optimal result from the Oracle. The advice complexity is defined as the number of bits the online algorithm reads from the advice tape.

Definition 3.2 (Online Algorithm with Advice [8])**.** *Consider an input I of an online maximization problem. An online algorithm A with advice computes the output sequence $A^{\phi} = (y_1, y_2, ..., y_n)$ such that y_i is computed from $\phi, x_1, x_2, ..., x_i$ and ϕ is the content of the advice tape (such as an infinite binary sequence). The cost of the computed output is denoted as $cost(A^{\phi}(I))$. The online algorithm A is c-competitive with advice complexity $s(n)$ if there exists some α such that, for every n and for each I of length at most n, there exists some ϕ such that $cost(A^{\phi}(I)) \geq \frac{1}{c} cost(OPT(I)) - \alpha$ and*

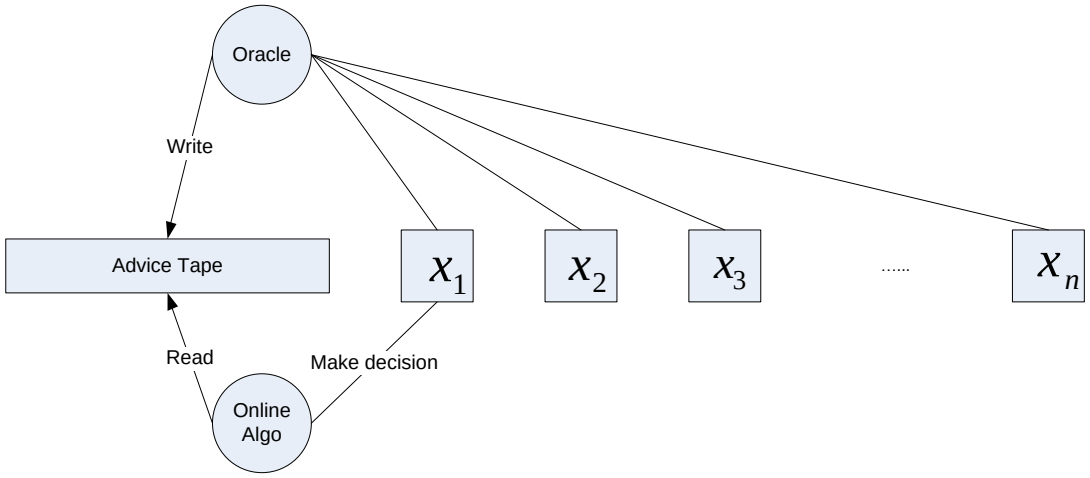


Figure 3.3: Advice Complexity Model

at most the first $s(n)$ bits of ϕ have been accessed during the computation of $A^{\phi}(I)$. If A is c-competitive for $\alpha = 0$, it is called strictly c-competitive.

The model of advice complexity for online problems describes how many bits of advice about the unknown parts of the input are necessary and sufficient to achieve a specific competitive ratio. It is obvious that advice tape with a linear number of advice bits is necessary to achieve an optimal solution by indicating a solution bit to each input. However, it is usually the case that few advice bits are sufficient to improve the competitive ratio. In [9] [13], the knapsack problem, the paging problem, a job shop scheduling problem and the problem of routing communication are examined. The consistent features for those problems is that advice tape with $O(logn)$ bits is sufficiently improve the competitive ratio over the best deterministic algorithm.

3.3.4. Online Knapsack with Advice Complexity. In the online version of the knapsack problem, the items arrive one after another and the algorithm has to decide for each item whether it is in the solution or not. Those items cannot be removed from the solution at a later stage.

Definition 3.3 (Online Knapsack Problem [8])**.** *The online knapsack problem is the following maximization problem. The input consists of a sequence of n items that are tuples of weights and profits, $S = \{s_1, s_2, \cdots, s_n\}$, $s_i = (w_i, v_i)$, where $0 < w_i \leq 1$ and $v_i > 0$ for $i \in \{1, 2, \cdots, n\}$. A feasible solution is any subset S' of S such that*

$\sum w_i \leq 1$; *the goal is to maximize* $\sum v_i$ *for* $i \in S'$. *the items are given an online fashion. For each item, an online algorithm A must specify whether this item is part of solution or not as soon as it is offered. In a simple online knapsack, each item's value* v_i *is smaller than 1 and equals its weight* w_i.

Advice complexity of online knapsack problem has been analyzed thoroughly in [8] The following theorems are all from [8]. First, it shows that it is impossible to achieve a competitive solution without advice.

Theorem 3.2. *No deterministic online algorithm for the simple knapsack problem(and, thus, the general knapsack problem) without advice is competitive.*

Then, the advice tape with a linear number of bits is necessary to compute an optimal solution.

Theorem 3.3. *There exists an optimal online algorithm A for the simple knapsack problem using n bits of advice.*

Theorem 3.4. *Any online algorithm with advice for the simple knapsack problem needs at least* $n - 1$ *bits to be optimal.*

Furthermore, a competitive ratio of 2 is achievable with a single advice bit. However, any further increase of the number of advice bits does not help until a logarithmic number is reached.

Theorem 3.5. *Let A be an online algorithm reading one bit of advice, which indicates whether there exists an item s within the input that has a size larger than* $\frac{1}{2}$. *The online algorithm A for simple knapsack problem is 2-competitive.*

Theorem 3.6. *Let* $b < \lfloor log(n-1) \rfloor$ *and let* $\epsilon > 0$. *No online algorithm for simple knapsack problem using b bits of advice is better than* $(2-\epsilon)$*-competitive.*

Theorem 3.7. *Let* $\epsilon > 0$. *There exists an online algorithm A with advice for the simple knapsack problem that achieves a competitive ratio of* $(1+\epsilon)$ *reading*

$$\left\lceil \frac{2\epsilon+2}{2} \right\rceil \cdot \lceil logn \rceil + 2 \cdot \left\lceil log \frac{2\epsilon+2}{\epsilon} \right\rceil + 2 \cdot \lceil log \lceil logn \rceil \rceil + 1$$

bits of advice.

Proof. For a fixed constant $\epsilon > 0$, let $\delta = \frac{\epsilon}{2+2\cdot\epsilon}$, which is verified in the end of the proof. The first bit in the advice tape will indicate whether there is an item with size larger than δ or not.

Suppose there does not exist any item with the size larger than δ, then the online algorithm could pick the item greedily as each item is coming. The competitive ratio will be calculated as $\frac{1}{1-\delta}$ since the online algorithm will obtain at least $1-\delta$.

$$\frac{1}{1-\delta} = 1 + \frac{\delta}{1-\delta} = 1 + \frac{\epsilon}{2+\epsilon} \leq 1 + \epsilon$$

On the contrary, suppose there exists some items of size larger than δ. Since the Oracle knows the whole input and has unlimited computational power, it can obtain the optimal solution and divide it into two disjoint sets S_1 and S_2, where S_1 denotes the set of items with size $> \delta$ named heavy items and S_2 contains items of size $\leq \delta$ named light items. Let s_1 and s_2 indicate total summation of items in S_1 and S_2 respectively and j represents the number of heavy items.

Since $j \cdot \delta \leq 1$, j need $\lceil log\frac{1}{\delta} \rceil$ bits to be encoded in the advice tape. The indexes of all the heavy items will be written into the advice tape using $j \cdot \lceil logn \rceil$ bits. For the purpose of being able to decode the advice string, the length $\lceil logn \rceil$ has to be included in the advice tape in some form using $2\lceil \lceil logn \rceil \rceil$ bits.

For the light items, the oracle could encode a number k on the advice tape, where k is:

$$k\delta \leq s_2 < (k+1)\delta$$

Since each item in S_2 with size less than δ, their total summation is not smaller than $k\delta - \delta > s_2 - 2\delta$. Therefore, the competitive ratio will be calculated as:

$$\frac{s_1 + s_2}{s_1 + s_2 - 2\delta} = \frac{1}{1-2\delta} = 1 + \frac{2\delta}{1-2\delta} = 1 + \epsilon$$

□

In summary, the advice tape which contains following three parts as shown in Table 3.3 will satisfied the competitive ratio $(1+\epsilon)$:

- The first bit indicates whether there exists any item within the input of size $> \delta$.

Table 3.3: Advice Tape for Simple Knapsack

First bit	j	Index 1	...	Index j	sum_{light}

- The heavy item is recorded with an index and total number of heavy items.
- The light items are recorded with a lower bound summation.

It has also been shown that $O(logn)$ bits (with a larger constant) are sufficient to get arbitrarily close to an optimal solution for the general online knapsack problem.

Theorem 3.8. *Let $\epsilon > 0$. There exists an online algorithm A for knapsack that achieves a competitive ratio of $1 + \epsilon$ using at most $O(logn)$ bits of advice.*

The proof is omitted here and could be found in [8].

4. PRELIMINARY EXPERIMENT RESULTS

In this section, an internal observer within DGI will be analyzed in FREEDM when he observes more and more information. The information leakage model has been generated using the advice tape concept. Quantitive information leakage is plotted out through experiments.

4.1. INFORMATION LEAKAGE MODEL

The attack model is shown in Figure 4.1: the system is the Oracle which sees all the SST values and applies a knapsack algorithm on them; the observer is the online algorithm that tries to obtain the secret supply value and remaining solution. Observing advice tape containing messages and the Load Table provided by DGI, the observer tries to deduce the total supply value as his observations increase.

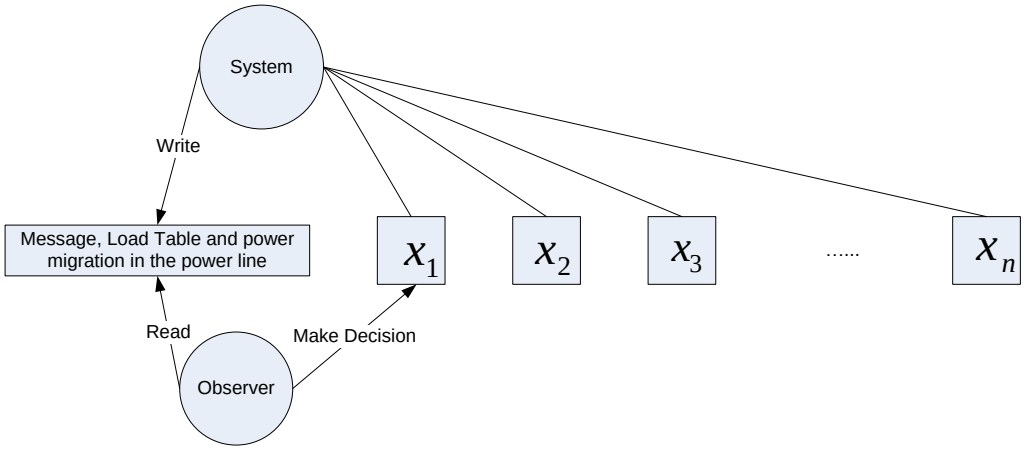


Figure 4.1: Advice Tape Modelfor DGI

The knapsack problem would either be the 0-1 knapsack or the fractional knapsack. The greedy algorithm will find the optimal solution to the fractional knapsack by choosing nodes with the highest demand values and allow fractional power to be

satisfied; while the dynamic programming will find the optimal solution to the 0-1 knapsack by choosing nodes with sum of demand values equal to the supply value. There also exists a greedy approach for the 0-1 knapsack that could obtain result within a bound of optimal solution [3].

Lemma 4.1. *Assuming that power migration is done by satisfying demands in one round and the greedy approach cannot be used for the 0-1 knapsack, it is possible for the observer to deduce what the problem is used within DGI.*

Proof. Because the observer could see the results of the power transfer in the physical power line, he could measure and compare it with the Load Table. The Load Table in DGI displays power of all demand nodes. If the observer doesn't see the select messages for the local node, or if the Load Table doesn't change at all in one power migration cycle, it means that the solution of the cyber algorithm doesn't contain the local node. If this happens once that the observed power in the power line is less than the demand value of local node, the problem will be the 0-1 knapsack rather than the fractional knapsack. □

For example, assume the observer (Node3) own a Load Table in Table 4.1 with his local demand value equals to 20.

Table 4.1: Load Table of Node3

Node1	Demand(30)
Node2	Demand(10)
Node3	Demand(20)
Node4	Supply
Node5	Demand(5)

Assume the supply value is Node4 with value 45 not visible in the Load Table, the 0-1 knapsack algorithm will choose Node1, Node2 and Node5, since $L_1+L_2+L5 = 30+10+5 = 45$. However, the greedy knapsack algorithm will choose Node1 and Node3 because it can satisfy Node3 partially. If the observer doesn't see a select message or any changes of Load Table in one state collection cycle, it means that Node3 isn't in

the solution. However, if the observer sees that the power line experienced 5 or 10 units of power transmission, it means that Node2 and Node5 are in the solution. Then the observer will know the problem is the 0-1 knapsack because the greedy algorithm for the fractional knapsack will choose the Node3 instead of Node2 and Node5.

Lemma 4.2. *Within DGI, Shannon Entropy is not an appropriate measurement for the information leakage as the it does not consider the ratio of revealed part over the secret. The observer will deduce the lower bound ratio of revealed physical observations over the secret.*

Proof. In the previous two sections, Shannon Entropy is used to measure the information leakage as the observer is outside the DGI. If x_i is revealed as a solution among $x_1, x_2, ..., x_n$ through physical observation, Shannon entropy is calculated as

$$SE(X|x_i) = \sum \frac{1}{2^{n-1}} log 2^{n-1} = n - 1 \tag{23}$$

This means that a large observed value will lead to the same information uncertainty as a small observed value. However, it is not appropriate to measure the information uncertainty for the observers within DGI. Because the Load Table will display all values from demand nodes, the supply value will be bounded by the sum of all demand values. Thus, each individual physical observation in the power line will reveal a ratio to the bounded supply value. A large observed value will have different ratio to the supply value with a small observed value. □

A function should be defined to measure the uncertainty of the information by considering both the number of revealed items and the ratio of the revealed part over the secret. The function is defined as F metric. The uncertainty of the information is between 0 and 1. 0 means the observer knows nothing about the information while 1 means the observer knows all about the information. If there is no item revealed, the uncertainty of the information is defined as 0. If there is at least one item revealed, the uncertainty of the information would be calculated by the F metric in the following.

$$F = \frac{2 \times \frac{h}{t} \times \frac{V}{X}}{\frac{h}{t} + \frac{V}{X}} \tag{24}$$

h is the number of revealed items; t is the number of items in the optimal solution; V is the value of revealed items; X is the value for the optimal solution. It is straightforward to obtain that $F = 1$ when $h = t$ and $V = X$.

Theorem 4.1. *The solution to the 0-1 knapsack problem is related with a threshold (δ) that differentiates the items in heavy or light in the solution. Table 4.3 shows that the threshold (δ) can be used for information leakage analysis when observed item in the solution increases.*

Proof. Recall that advice tape has a fixed δ which will differentiate items in solution into two part: heavy item is larger than δ and light item is smaller than δ. Let f represent the flag for a heavy item and m represent the number of heavy items in the advice tape. When the first bit f is revealed, Table 4.2 shows the boundary of X.

Table 4.2: First Bit

First Bit	x_1	x_2	...	x_n
0				$\frac{x_n}{X} \leq \delta \Rightarrow X \geq \frac{x_n}{\delta}$
1				$\frac{x_n}{X} > \delta \Rightarrow X < \frac{x_n}{\delta}$

If the first bit and the number of heavy items are revealed as 1 and m, respectively, the lower bound of X will be the sum of m smallest values in the Load Table, and the upper bound will be the maximum value over δ :

$$x_1 + x_2 + ... + x_m < X < \frac{x_n}{\delta} \tag{25}$$

As each heavy item is revealed, both the lower bound and the upper bound of X are updated correspondingly. If only one item x_i is revealed in advice tape as a heavy item, then

$$x_1 + x_2 + ... + x_{m-1} + x_i < X < \frac{x_i}{\delta} \tag{26}$$

If two items x_i and x_j are revealed as heavy items, then

$$x_1 + x_2 + ... + x_{m-2} + x_i + x_j < X < min(\frac{x_i}{\delta}, \frac{x_j}{\delta}) \tag{27}$$

As more and more items are revealed, both the lower bound and the upper bound of X are updated, the more the certainty of the constraint in the solution has. Table 4.3 shows the boundary of X as the advice tape (with a fixed δ) is revealed bit by bit. □

Table 4.3: Information Leakage

Bit meaning	Number of Bits in Advice Tape	Boundary of X	F Uncertainty
f	1	$\frac{x_n}{\delta} > X$	$F > \frac{2\times\frac{1}{t}\times\delta}{\frac{1}{t}+\delta}$
f, m	$< 1 + \lceil log\frac{1}{\delta} \rceil$	$\frac{x_n}{\delta} > X > x_1 + x_2 + ... + x_m$	$F > \frac{2\times\frac{m}{t}\times m\times\delta}{\frac{1}{t}+m\times\delta}$
f, m, h_1	$< 1 + \lceil log\frac{1}{\delta} \rceil + \lceil logn \rceil$	$\frac{h_i}{\delta} > X > x_1 + x_2 + ... + x_{m-1} + h_1$	$F > \frac{2\times\frac{m}{t}\times\frac{h_1}{X}+(m-1)\times\delta}{\frac{m}{t}+\frac{h_1}{X}+(m-1)\times\delta}$
f, m, h_1, h_2	$< 1 + \lceil log\frac{1}{\delta} \rceil + 2\lceil logn \rceil$	$min(\frac{h_1}{\delta}, \frac{h_2}{\delta}) > X > x_1 + x_2 + ... + x_{m-2} + h_1 + h_2$	$F > \frac{2\times\frac{m}{t}\times\frac{h_1+h_2}{X}\times(m-2)\times\delta}{\frac{m}{t}+\frac{h_1+h_2}{X}+(m-2)\times\delta}$
	...	...	...
$f, m, h_1, ..., h_m$	$< 1 + \lceil log\frac{1}{\delta} \rceil + \lceil log\frac{1}{\delta} \rceil \lceil logn \rceil$	$min(\frac{h_1}{\delta}, \frac{h_2}{\delta}, ..., \frac{h_m}{\delta}) > X > h_1 + h_2 + ... + h_m$	$F > \frac{2\times\frac{m}{t}\times\frac{h_1+h_2+...+h_m}{X}}{\frac{m}{t}+\frac{h_1+h_2+...+h_m}{X}}$

Within DGI, the observer can use the Load Table to calculate the average value among the demand nodes. A big observed power migration in the power line (larger than the average value) indicates there is a heavy item in the solution; while a small observed power migration in the power line (less than the average value) indicates there is a light item in the solution. A selection of a DGI in the Load Balance algorithm is an indication that the observed node is in the solution. The attacker can obtain multiple readings from its neighbors. As the attacker observes more and more items, the boundary of constraint will be calculated out by differentiating it as a heavy item or a light item based on a δ.

4.2. EXPERIMENT RESULTS

The following example presents some results of quantifying the uncertainty of information of the supply value (the constrained resource) from FREEDM.

4.2.1. Case 1: Six Items as Solution to a Knapsack Problem. The Load Table in DGI shows demand values 30, 60, 75, 200, 45, 20, 90, 120, 39, 47, 54, 108, 112 in the system with total summation 1000. Suppose the optimal solution for supply value 422 is $45, 20, 90, 120, 39, 108$.

For $\delta = 0.2$, the heavy items are $90, 120, 108$ because the threshold is $422 \times 0.2 = 84.4$. The total numbers of different sequences of items in the solution are $6! = 720$. The Figure 4.2 shows the uncertainty family curves for six items showing up in different orders with a $\delta = 0.2$. The uncertainty portion is the shaded area.

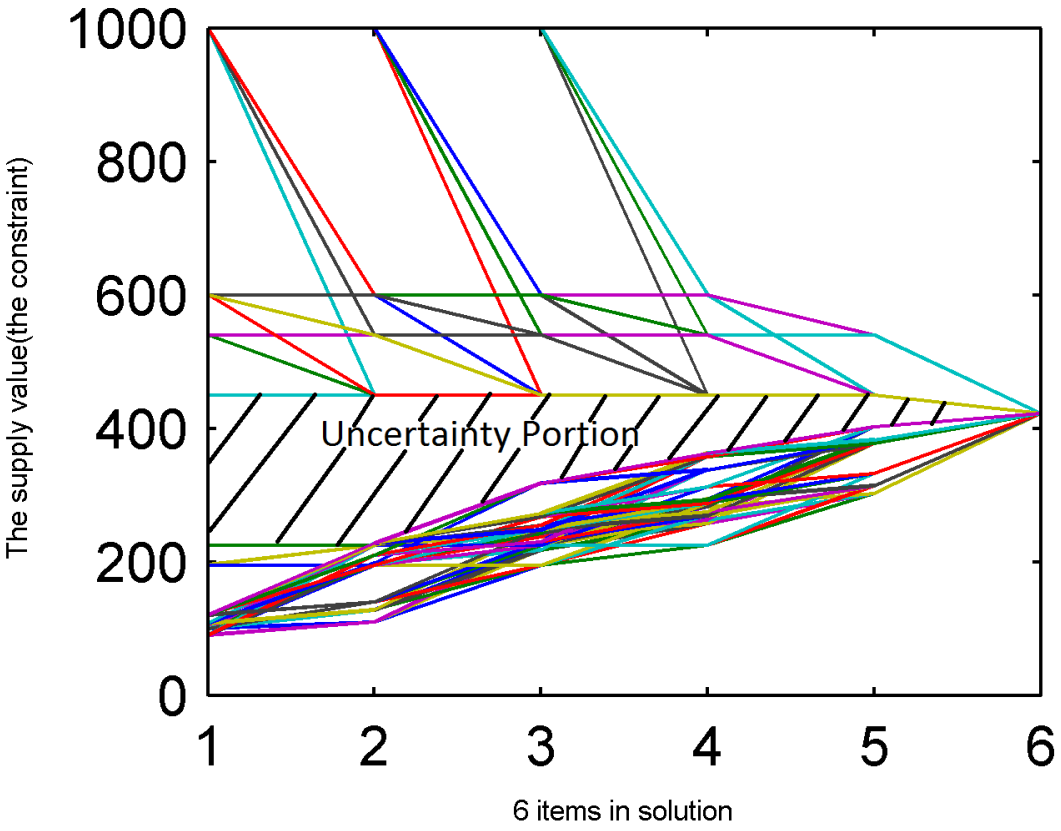


Figure 4.2: Uncertainty Portion with 6 items and $\delta = 0.2$

For a $\delta = 0.08$, the heavy items are $45, 39, 90, 120, 108$ because the threshold is $422 \times 0.08 = 33.76$. The Figure 4.3 shows the whole uncertainty family curves for six

items showing up in different orders with a smaller $\delta = 0.08$. The uncertainty portion is the shaded area.

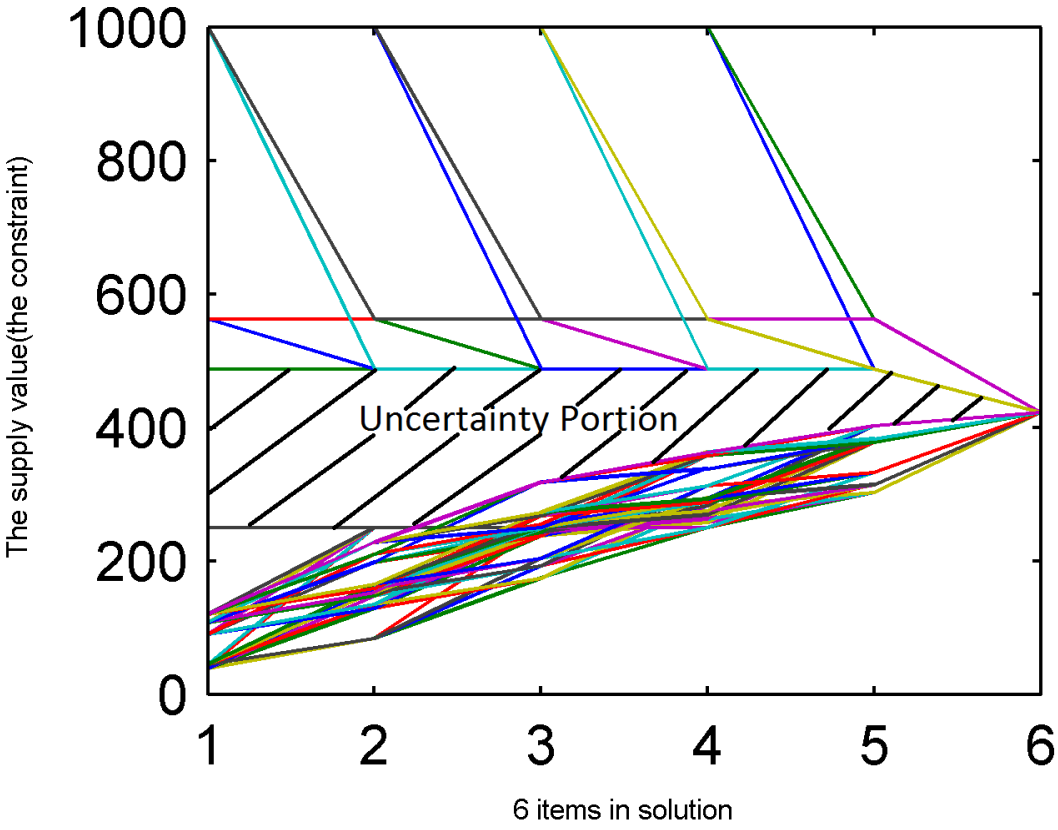


Figure 4.3: Uncertainty Portion with 6 items and $\delta = 0.08$

The similarity of Figure 4.2 and Figure 4.3 shows that for a specific knapsack problem different δ won't make much change to the shape of the uncertainty portion as long as the heavy part contains the item that is close to the threshold.

4.2.2. Case 2: Split the Light Items in Case 1 . The knapsack problem has been kept the heavy items unchanged and split the light items to make bigger differences between the heavy part and the light part.

Suppose the problem is 30, 60, 75, 200, 21, 14, 10, 9, 90, 120, 10, 17, 22, 47, 54, 108, 112 and the optimal solution for constraint 422 is $21, 14, 9, 11, 90, 120, 10, 17, 22, 108$. For those 10 items in the solution, the heavy items are $90, 108, 120$ and the possible

sequences of those 10 items are 10!. Partial sequences have been picked to show the uncertainty curves for $\delta = 0.2$ in Figure 4.4 and $\delta = 0.08$ in Figure 4.5.

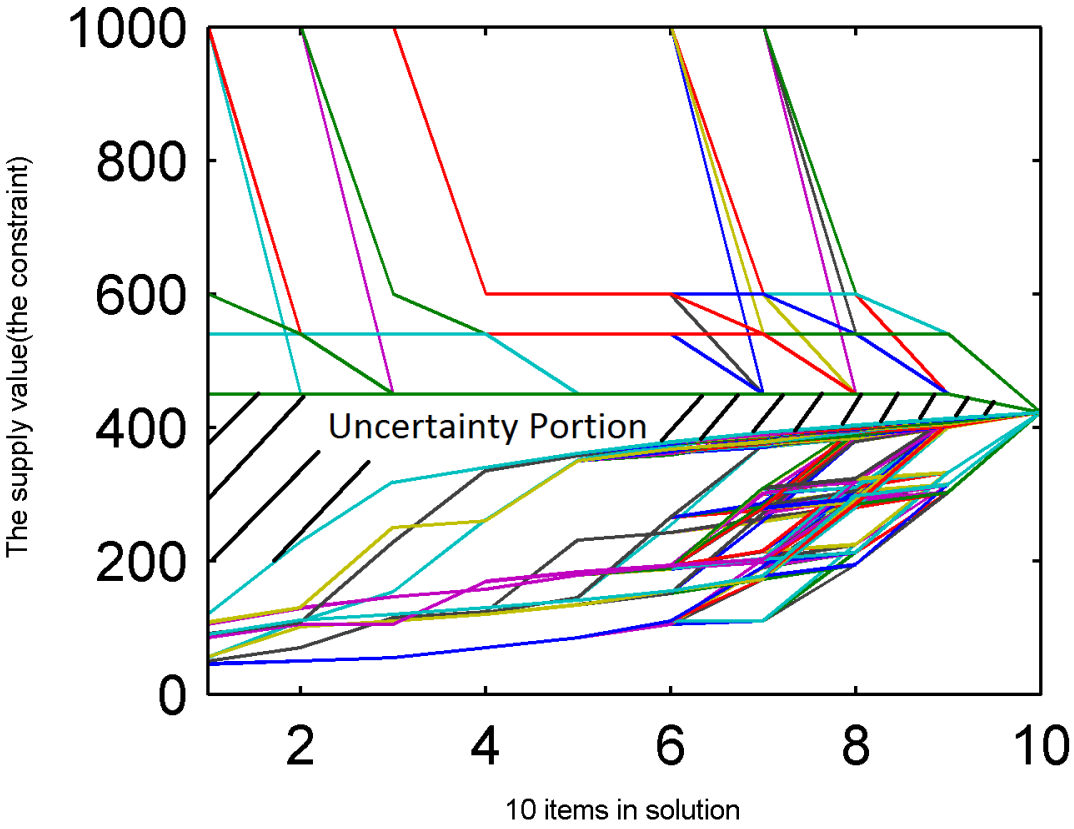


Figure 4.4: Uncertainty Portion with 10 Items and $\delta = 0.2$ by Splitting the Light Items

In Figure 4.5, the uncertainty portion that is the shaded area between the upper bounds and the lower bounds is much bigger than those of the previous. The upper bound of the constraint has not changed until the last item shows up. The difference between Figure 4.4 and Figure 4.5 shows that given the total sum of all items in the system is much larger than the individual item, a heavy part that is far from the the threshold will lead to more uncertainty of the constraint.

4.2.3. Case 3: Split the Heavy Items in Case 1. If the heavy items have been split into smaller and similar values: 30, 60, 75, 66, 44, 62, 26, 45, 12, 8, 42, 48, 73, 47, 55, 53, 39, 49, 54, 43, 69, the optimal solution for constraint 422

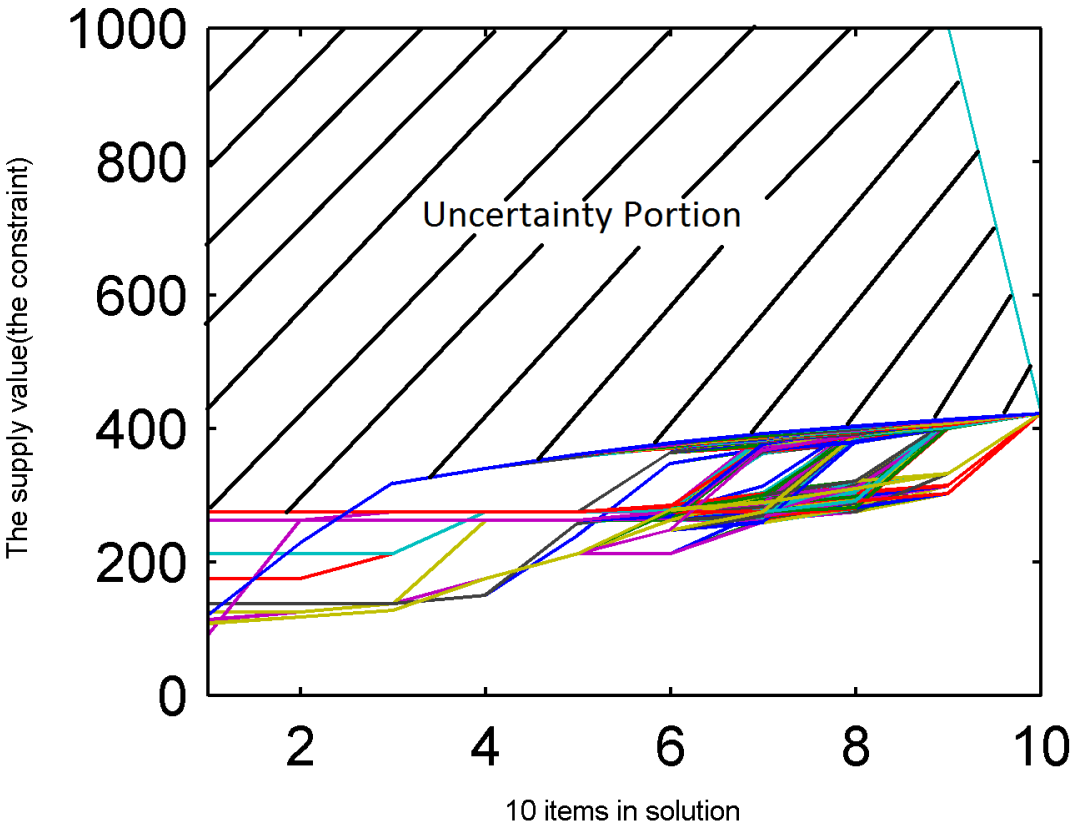


Figure 4.5: Uncertainty Portion with 10 Items and $\delta = 0.08$ by Splitting the Light Items

is $45, 12, 8, 42, 48, 73, 47, 55, 53, 39$. With a $\delta = 0.08$ and threshold 33.67, the heavy items are $45, 42, 48, 73, 47, 55, 53, 39$. The uncertainty curves have been chosen from 10! sequences and shown in Figure 4.6.

The lines above 400 are the upper bounds of the constraint and the lines below 400 are the lower bounds. The uncertainty portion is the shade area between the upper bounds and lower bounds. Because the heavy part contains items that are close to the threshold, the uncertainty portion shows a similar shape as in Figure 4.2, Figure 4.3 and Figure 4.4.

By considering the ratio of revealed part over the constraint, the advice tape will help to deduce some parts of the solution for 0-1 knapsack with special property. A trapdoor knapsack [81] is a knapsack problem with input consisting of a sequence of n items: $\{x_1, x_2, \cdots, x_n\}$ and has a property that: $x_i > \sum_{i-1}^{j=1} x_j$.

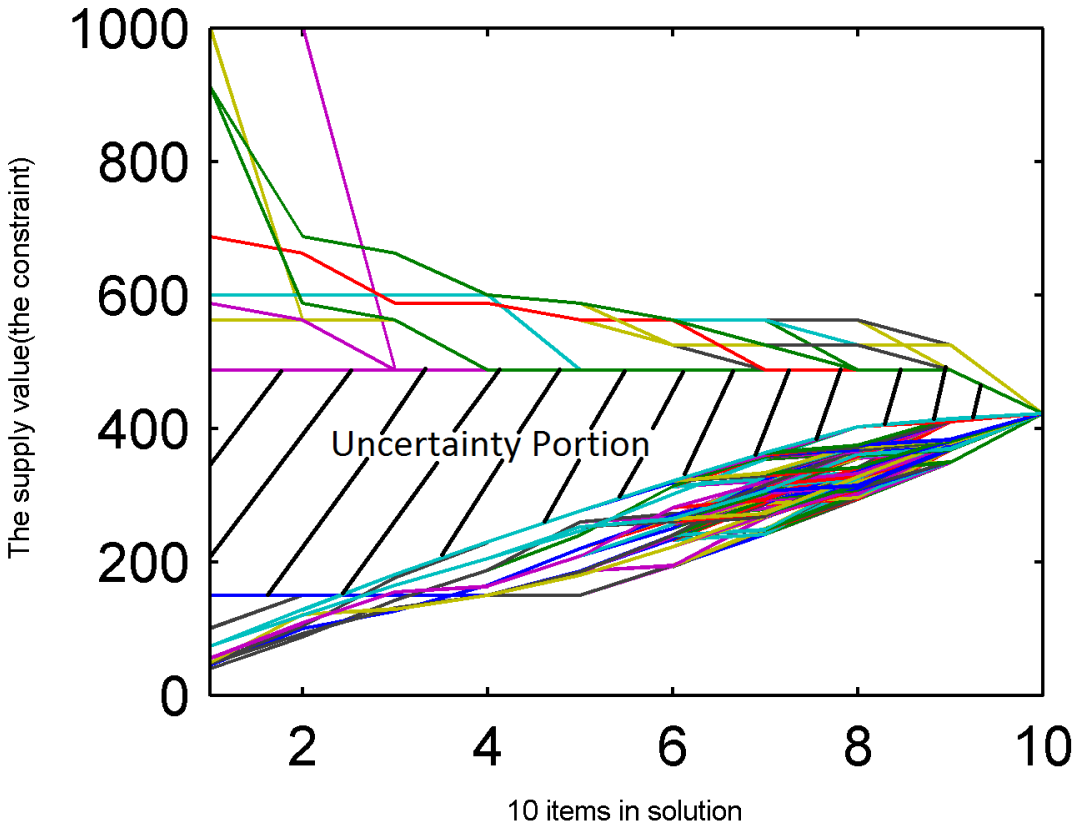


Figure 4.6: Uncertainty Portion with 10 Items and $\delta = 0.08$ by Splitting the Heavy Items

Corollary 4.1. *For a trapdoor knapsack, an observed heavy item will reveal extra information about the solution.*

Proof. If a heavy item A is revealed, the following inequality has to be satisfied:

$$\frac{A}{X} > \delta \Rightarrow X < \frac{A}{\delta}$$

The remaining solution will be revealed under a specific δ. To be more specific, if $\delta \geq 0.5$, then $X < 2A$. Since each item in the Load Table has a property that $x_i > \sum_{i-1}^{j=1} x_j$, the items after item A won't be in the solution as shown in Table 4.4. □

For example, if we have Table 4.5 as Load Table that satisfies the trapdoor knapsack property, Node1 won't belong to the solution as Node5 is known as a heavy

Table 4.4: Revealed Item and Deduced Solution

	x_1	x_2	...	A	...	x_n
no advice				1		
$\delta \geq 0.5$ and A belongs to heavy				1	0	0

item with $\delta \geq 0.5$. If Node5 is known as the heavy item in the solution with an assumption that $\delta \geq 0.5$,

$$\frac{8}{X} > \delta \geq 0.5 \Rightarrow X < 2 * 8 = 16$$

therefore, Node1 won't be in the solution because $X < 16$.

Table 4.5: Another Example of Load Table

Node1	Demand(16)
Node2	Demand(4)
Node3	Demand(2)
Node4	Supply
Node5	Demand(8)

5. THE INFORMATION FLOW QUANTIFICATION FRAMEWORK

The above presented work is a preliminary result that observations in FREEDM can lead to different levels of information leakage. Under this analysis, there are some mitigation strategies that could reduce the information leakage as much as possible. For example, the fractional knapsack is much more vulnerable than the 0-1 knapsack assuming that greedy approximation approach is not used for the 0-1 knapsack. If the greedy algorithm is deduced by the attacker through the observations, the attacker can attack the system by sending out a large fake demand value that is guaranteed to be the solution in the next power migration cycle. The experimental evidence in the previous section also shows that there is a system configuration that makes the observer won't obtain much until in the very late phase that all items are observed Figure 4.5. Empirically, Load Balance in FREEDM should pick heavier items and lighter items with large differences as the solution to enlarge the uncertainty of the information to the observer.

In this section, a framework for quantify information flow in constrained resource CPSs is proposed. Algorithms have been provided to calculate the quantified information leakage as well as the maximal information leakge to a certain number of observations. FREEDM has been used as an example for demonstrate usage of this framework in a real CPS. Security strategy is confirmed through the experiment results on different distributions of the heavy and light items [32].

5.1. CONFIDENCE INTERVAL

In statistics, both point estimation and interval estimation are the uses of the sample data to estimate an unknown parameter. Point estimation uses the data to calculate a single estimate of the variable. Interval estimation provides a range of values that may contain the unknown variable. A confidence interval is an interval estimation that contains the unknown variable with certain degree of confidence. It is used as an estimator to measure the range for the unknown variable.

In general, a confidence interval of an unknown variable is: point estimate $\pm$ reliability coefficient $\times$ standard error. Let $Z_{\alpha/2}$ be the value that cuts off an area of $\alpha/2$ in the upper tail of the standard normal distribution of the unknown variable

x. Let $\bar{x}$ be the mean value for the sampling observations, σ the deviation and n the number of sampling observation. A $1-\alpha$ confidence interval for the variable is:

$$(\bar{x} - Z_{\alpha/2}\frac{\sigma}{\sqrt{n}}, \bar{x} + Z_{\alpha/2}\frac{\sigma}{\sqrt{n}}) \tag{28}$$

The length of the confidence interval is defined as: 2× reliability coefficient × standard error, that is $2 \times Z_{\alpha/2} \times \frac{\sigma}{\sqrt{n}}$. The margin error is defined as half of the length of the confidence interval: reliability coefficient × standard error, that is $Z_{\alpha/2} \times \frac{\sigma}{\sqrt{n}}$.

5.2. FRAMEWORK

The observation from the physical system will give the observer a feeling about the constrained resource in the solution. The output of cyber algorithm and physical observations will provide a certain number of samples to the constrained resource calculated by applying advice tape analysis. Given the assumption that the constrained resource has a public distribution, quantification information flow is accomplished by calculating the confidence interval of the constrained resource.

The proposed framework is presented in Figure 5.1.

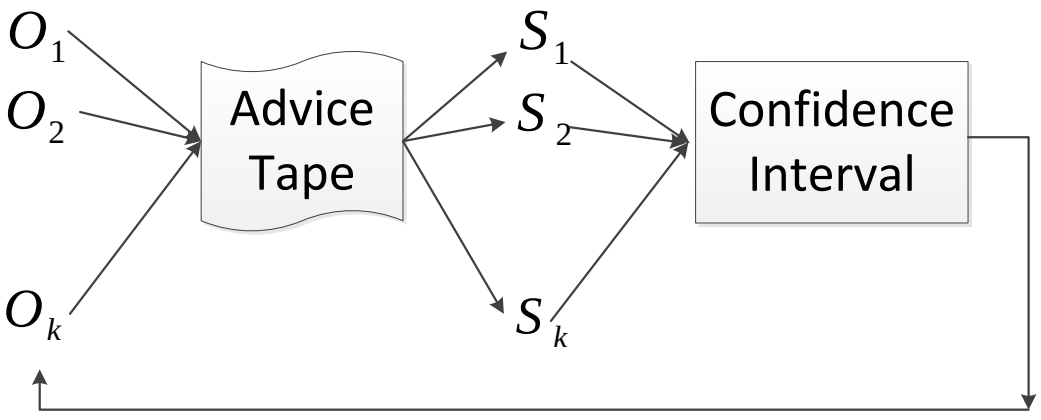


Figure 5.1: Observation Based Information Leakage Analysis Framework

The $O_1, O_2, \ldots, O_k$ represent multiple observations. Under the analysis of the advice tape, each observation will represent an approximate sample $S_1, S_2, \ldots, S_k$ to the constrained resource. Assuming the constrained resource has a known distribution, different subsets of observations will generate different confidence intervals for the constrained resource. Those ranges could be used as the mathematical way to measure the information leakage of the observations in return.

Normal distributions are often used for real-valued random variables whose distribution is unknown. In this paper, it is assumed that the constrained resource (the supply value) has a normal distribution with an unknown mean μ and a known or unknown standard deviation σ in FREEDM. It tries to answer questions such as how to quantify information leakage based on observations and what observations will give the attacker most valuable information. To be more specific, in the load balance algorithm in FREEDM, the output will be a Load Table that consists of all values from demand nodes. Given that the constrained resource(the supply value) has a known distribution, what sequences of the observed demand nodes combined with the Load Table will lead to higher information leakage and how to quantify it?

The following will analyze the confidence interval for the unknown mean μ based on the Load Table and the advice tape.

Based on an advice tape with threshold δ, if an item claims it's a heavy item, it will contribute an upper bound of the constrained resource.

$$\frac{H}{C} > \delta \Longrightarrow C < \frac{H}{\delta} \tag{29}$$

If an item claims it's a light item, it will contribute a lower bound of the constrained resource.

$$\frac{L}{C} \leq \delta \Longrightarrow C \geq \frac{L}{\delta} \tag{30}$$

Suppose the total number of items in the optimal solution is m. If the solution items are showing one by one individually, the attacker will obtain m guessing values $C_1, C_2, ..., C_m$ based on the equation (29) and (30). The mean value among from those m guessing values will be calculated by (31).

$$\bar{C} = \frac{C_1 + C_2 + ... + C_m}{m} \tag{31}$$

A $1 - \alpha$ confidence interval for the supply value is:

$$(\bar{C} - Z_{\alpha/2}\frac{\sigma}{\sqrt{m}}, \bar{C} + Z_{\alpha/2}\frac{\sigma}{\sqrt{m}}) \tag{32}$$

The margin error is $Z_{\alpha/2}\frac{\sigma}{\sqrt{m}}$.

To be specific, a 95% confidence interval for μ-the supply value is:

$$(\bar{C} - 1.96\frac{\sigma}{\sqrt{m}}, \bar{C} + 1.96\frac{\sigma}{\sqrt{m}}) \tag{33}$$

The margin error is $2 \times 1.96\frac{\sigma}{\sqrt{m}}$.

If the number of heavy items are known as h and all heavy items are shown individually as $H_1, H_2, ..., H_h$, the attacker could obtain h by guessing numbers $C_1, C_2, ..., C_h$ and calculate the mean of observations.

$$\bar{C_h} = \frac{C_1 + C_2 + ... + C_h}{h} \tag{34}$$

Then, for a 95% confidence interval for μ, the supply value is:

$$(\bar{C_h} - 1.96\frac{\sigma}{\sqrt{h}}, \bar{C_h} + 1.96\frac{\sigma}{\sqrt{h}}) \tag{35}$$

Similarly, if the number of light items are known as l and all light items are shown individually as $L_1, L_2, ..., L_l$, the attacker could obtain l by guessing numbers $C_1, C_2, ..., C_l$ and calculate mean of the observations.

$$\bar{C_l} = \frac{C_1 + C_2 + ... + C_l}{l} \tag{36}$$

Then, for a 95% confidence interval for μ, the supply value is:

$$(\bar{C_l} - 1.96\frac{\sigma}{\sqrt{l}}, \bar{C_l} + 1.96\frac{\sigma}{\sqrt{l}}) \tag{37}$$

The above analysis is based on the assumption that the deviation σ is known to the attacker. However, in reality, σ is usually unknown. Based on the value from the Load Table and the advice tape, let s be the variance that could be calculated by

$$s^2 = \frac{1}{m-1}\sum_{i=1}^{m}(C_i - \bar{C})^2 \tag{38}$$

Then the confidence intervals can be constructed based on the t-statistic, a continuous probability distribution to estimate the mean of a normally distributed population where the sample size is small and standard deviation is unknown.

$$T = \frac{\bar{C} - \mu}{s/\sqrt{m}} \tag{39}$$

Let $t_{\alpha/2}(m-1)$ be the value that cuts off the upper area of $\alpha/2$ in a t-distribution, the confidence interval based on the t-distribution is:

$$(\bar{C} - t_{\alpha/2}(m-1)\frac{s}{\sqrt{m}}, \bar{C} + t_{\alpha/2}(m-1)\frac{s}{\sqrt{m}}) \tag{40}$$

Observation 5.1. *As the margin of error depends on the number of observations and variance, the more values that are confirmed to be the solution, the narrower the observed range of the constrained resource (the supply value) will be.*

Proof. Given only one observation O_1, it does not contribute to the confidence interval analysis. Because it should be either the heavy item or the light item, it can lead to either the upper bound of the constrained resource or the lower bound of the constrained resource.

Given two observations O_1 and O_2, they will lead to the range $R(O_1, O_2)$ of the constrained resource if one belongs to heavy and one belongs to light. Based on the range $R(O_1, O_2)$, two samples $C_1, C_2 \in R(O_1, O_2)$ could be guessed and used for calculating mean value C and variance s.

Given three observations, O_1, O_2 and O_3, the range of the constrained resource will be $R(O_i, O_j)$ - the minimum range among $R(O_1, O_2), R(O_2, O_3)$ and $R(O_1, O_3)$. Based on the minimum range, three samples $S_1, S_2, S_3 \in R(O_i, O_j)$ could be guessed and used for calculating mean value C' and variance s'. Since the range is smaller or

equal to the range from the above two observations, the variance s' will be smaller or equal to s.

As observations have been revealed, s the variance will decrease or at least stay the same according to above analysis. The margin of error is defined as $Z_{\alpha/2}\frac{s}{\sqrt{m}}$. As m will be larger, the margin error will decrease. In other words, the range of the constrained resource will be narrowed down as shown in Figure 5.2.

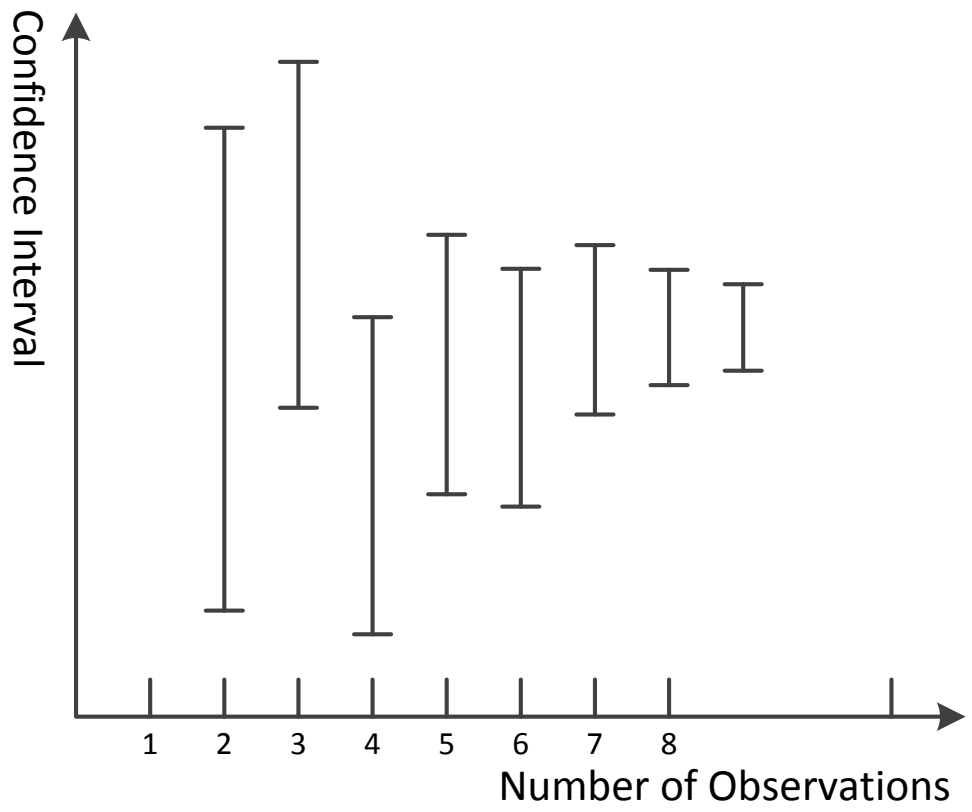


Figure 5.2: Confidence Interval with Observations

□

Lemma 5.1. *Under certain constrained resources and thresholds in the advice tape, given the same number of revealed light items and heavy items, the confidence interval*

given by the light items will be larger than the confidence interval given by the heavy items.

Proof. Suppose the revealed number of heavy items is k with values $H_1, H_2, ..., H_k$, under threshold δ, the attacker will obtain the range for the constrained resource C by satisfying the equation (41).

$$H_1 + H_2 + ... + H_k < C < min(\frac{H_1}{\delta}, \frac{H_2}{\delta}, ..., \frac{H_k}{\delta}) \tag{41}$$

Similarly, k revealed light items will have to satisfy the range for the constrained resource C based on equation (42).

$$max(\frac{L_1}{\delta}, \frac{L_2}{\delta}, ..., \frac{L_k}{\delta}) \leq C \tag{42}$$

Suppose $C_1, C_2, ..., C_k$ are samples to C chosen from equation (41) and $C'_1, C'_2, ..., C'_k$ are samples to C chosen from equation (42). The s_h and s_l represent the variance for heavy items and light items respectively.

The mean value is calculated from equation (43), s_h from equation (44), s_l from equation (45)

$$\bar{C} = \frac{\sum_{i=1}^{k} C_i + \sum_{i=1}^{k} C'_i}{2 * k} \tag{43}$$

$$s_h^2 = \frac{1}{k-1} \sum_{i=1}^{k} (H_i - \bar{C})^2 \tag{44}$$

$$s_l^2 = \frac{1}{k-1} \sum_{i=1}^{k} (L_i - \bar{C})^2 \tag{45}$$

Since each $L_i < H_i < \bar{C}$, the differences between L_i with $\bar{C}$ are much bigger. Thus, s_h will be less than s_l.

According to equation (40), the confidence interval by k heavy revealed items is less than k light items. In other words, the same number of heavy item will leak more information than light items regarding the constrained resource. □

In a CPS with constrained resources, the resource allocation is assumed to be done by applying 0-1 knapsack algorithm. Given the advice tape to 0-1 knapsack

problem, the threshold δ differentiates the normalized input items as either heavy ($> \delta$) or light ($\leq \delta$). The observations $O = O_1, O_2, ..., O_k$ from the physical devices in CPSs will be divided into heavy parts $H_1, H_2, ..., H_i$ and light parts $L_1, L_2, ..., L_j$ with $i + j = k$. The constrained resource C has been assumed a normal distribution $N(\mu, \delta^2)$.

Theorem 5.1. *Given any subsets of the entire observation $H_1, H_2, ..., H_i, L_1, L_2, ..., L_j$, observed from the output of a cyber algorithm, the confidence interval $(\bar{C} - t_{\alpha/2}(m-1)\frac{s}{\sqrt{m}}, \bar{C} + t_{\alpha/2}(m-1)\frac{s}{\sqrt{m}})$ is calculated based on the t-statistic and represents the uncertainty of the actual value of the constrained resource in the CPS. m is the number of observed items.*

Proof. According to equation (41) and (42), any subsets of the entire observation gives range for the observer to obtain a range of the value of the constrained resource C.

For any subsets observation $H_1, ..., H_u, L_1, ..., L_v$, the equation(46) has to be satisfied.

$$max((\sum_{x=1}^{u} H_x + \sum_{y=1}^{v} L_y), max(\frac{L_y}{\delta})) < C < min(\frac{H_x}{\delta}) \tag{46}$$

The observer generates $C_1, C_2, ..., C_{u+v}$ within the above range corresponding to $H_1, ..., H_u, L_1, ..., L_v$. According to the mean value calculation, equation (38), and equation (39), the confidence interval $(\bar{C} - t_{\alpha/2}(m-1)\frac{s}{\sqrt{m}}, \bar{C} + t_{\alpha/2}(m-1)\frac{s}{\sqrt{m}})$ is obtained with $m = u + v$. □

By applying specific values from the system and comparing the range of the confidence interval, the amount information leakage can be quantified in terms of the observations.

5.3. CASE STUDY AND EVALUATION RESULTS IN FREEDM

The examples from [31] will be used for the case study here. Suppose the problem is 30, 60, 75, 200, 21, 14, 10, 9, 90, 120, 10, 17, 22, 47, 54, 108, 112 and the optimal solution for constraint 422 is $21, 14, 9, 11, 90, 120, 10, 17, 22, 108$. For those 10 items in the solution, the heavy items are $90, 108, 120$ with $\delta = 0.2$.

5.3.1. Case 1: The Heavy Items are Seen Before the Light Items. For the specific sequence of the observations, the Confidence Interval with $Z_{\alpha/2} = 1.96$ has shown in Table 5.1. Figure 5.3 shows the confidence interval with the corresponding observations.

Table 5.1: Observations with Confidence Interval for Case 1

No.	Observed Items	Confidence Interval
1	90	N/A
2	90, 120	(129, 600)
3	90, 120, 21	(150, 553)
4	90, 120, 21, 108	(258, 540)
5	90, 120, 21, 108, 11	(269, 499)
6	90, 120, 21, 108, 11, 10	(279, 459)
7	90, 120, 21, 108, 11, 10, 14	(300, 458)
8	90, 120, 21, 108, 11, 10, 14, 17	(315, 441)
9	90, 120, 21, 108, 11, 10, 14, 17, 9	(403, 429)
10	90, 120, 21, 108, 11, 10, 14, 17, 9, 22	422

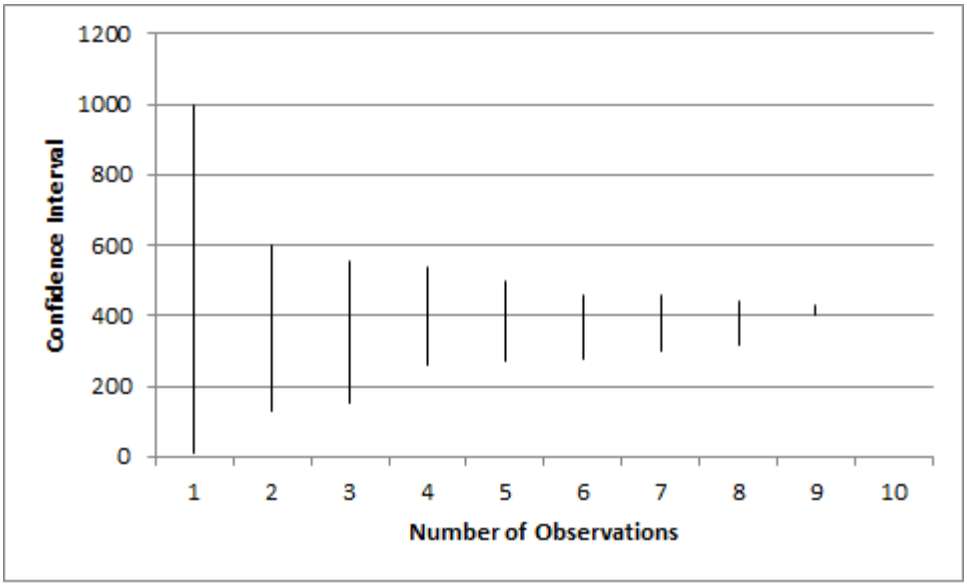


Figure 5.3: Observation to Confidence Interval for Case 1

5.3.2. Case 2: The Light Items are Seen Before the Heavy Items. For another sequence of the observations (the light items are seen first before the heavy items), the Confidence Interval with $Z_{\alpha/2} = 1.96$ has shown in Table 5.2. Figure 5.4 shows the confidence interval with the corresponding observations.

Table 5.2: Observations with Confidence Interval for Case 2

No.	Observed Items	Confidence Interval
1	11	N/A
2	11, 10	(55, 1000)
3	11, 10, 21	(109, 1000)
4	11, 10, 21, 14	(116, 1000)
5	11, 10, 21, 14, 17	(123, 1000)
6	11, 10, 21, 14, 17, 90	(184, 330)
7	11, 10, 21, 14, 17, 90, 22	(197, 317)
8	11, 10, 21, 14, 17, 90, 22, 9	(195, 305)
9	11, 10, 21, 14, 17, 90, 22, 9, 108	(300, 374)
10	11, 10, 21, 14, 17, 90, 22, 9, 108, 120	422

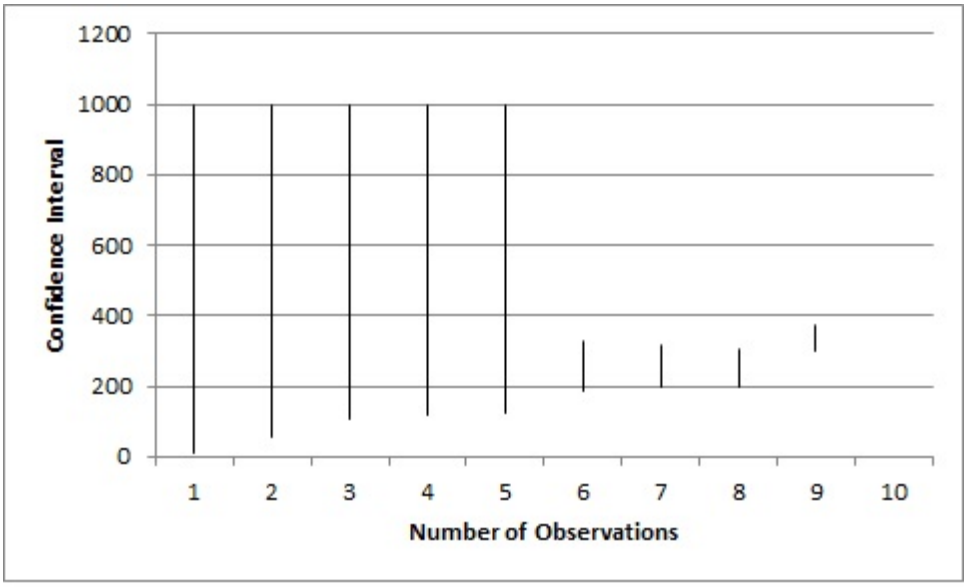


Figure 5.4: Observation to Confidence Interval for Case 2

From Figure 5.3 and Figure 5.4, it is clear that observed heavy items will reduce the confidence interval much more than the light items. That is, the observed heavy items will help the attacker known much more sensitive information than the light ones.

5.3.3. FREEDM Example and Evaluation Result. The implemented algorithm in the DGI for the Supply Node to pick the Demand nodes is a greedy knapsack. The greedy algorithm is used to approximate an optimal solution to the knapsack problem.

In this experiment, as the Supply node has chosen the Demand nodes 21, 14, 9, 11, 90, 120, 10, 17, 22, 108 for power migration, it will do power migrations with the Demand node 120 since it is the highest Demand node. Unit power migration is transmitted in each power migration cycle. The Supply node will migrate 12 units of power only for the Demand node 120 until $120 - 12$ and 108 are both highest Demand nodes. Then power migration will switch between two Demand nodes 120 and 80 until $120 - 30$, $108 - 18$ and 90 are the highest Demand nodes. So on and so forth, all the Demand nodes will be satisfied as shown in Figure 5.5.

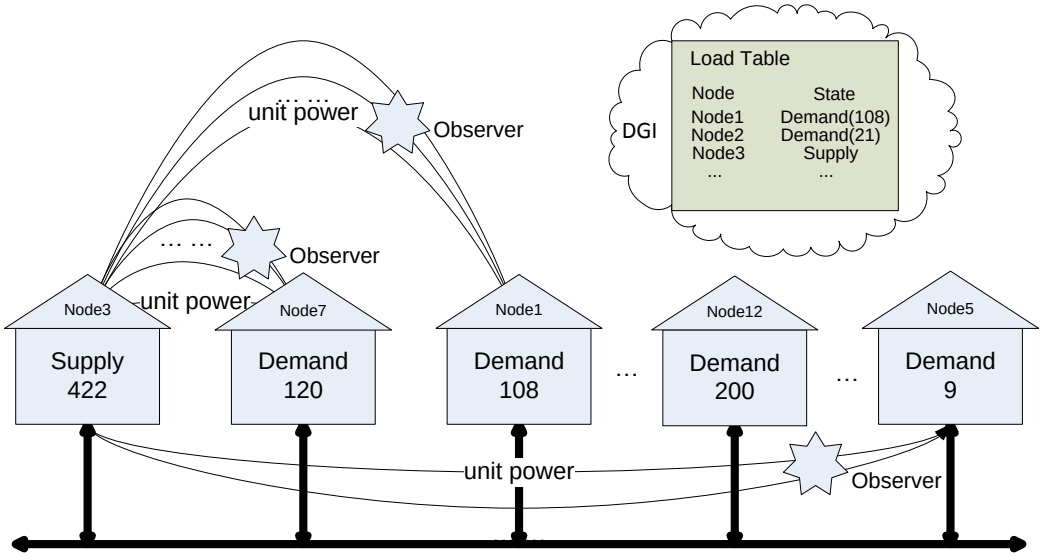


Figure 5.5: Power Migration in FREEDM for the Specific Example

Based on advice tape analysis, the observer will be able to generate Confidence Intervals for the constrained resource by counting the unit power migration in the power line and see the Load Table. Table 5.3 and Figure 5.6 show the results based on the experiment and analysis.

Table 5.3: Observations with Confidence Interval from FREEDM

Power	Observed Items	Confidence Interval
1-12	120	N/A
13-48	120, 108	(352, 540)
49-252	120, 108, 90	(409, 444)
253-256	120, 108, 90, 22	(366, 447)
257-276	120, 108, 90, 22, 21	(373, 433)
277-294	120, 108, 90, 22, 21, 17	(380, 422)
295-315	120, 108, 90, 22, 21, 17, 14	(397, 423)
316-323	120, 108, 90, 22, 21, 17, 14, 11	(408, 424)
324-332	120, 108, 90, 22, 21, 17, 14, 11, 10	(419, 431)
333-422	120, 108, 90, 22, 21, 17, 14, 11, 10, 9	422

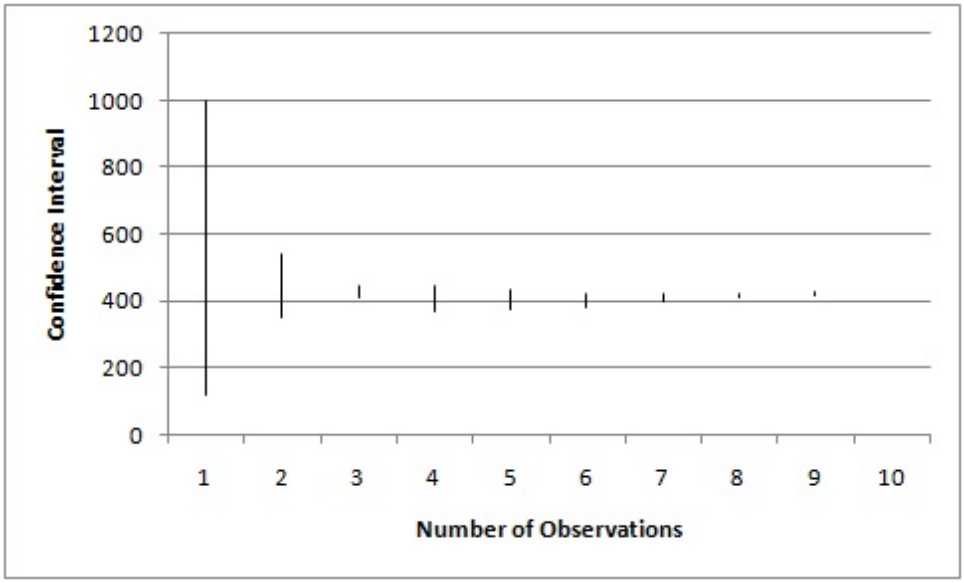


Figure 5.6: Observation to Confidence Interval in FREEDM

5.4. ALGORITHM

In constrained flow based CPS, if the values of all items are available just like the Load Table, the information leakage can be quantified in Algorithm 1 represented by the confidence interval using advice tape analysis as the observations are shown up. The Algorithm 2 provides a way to calculate maximal confidence interval to a certain number of observations based on Algorithm 1.

Theorem 5.2. *The Algorithm 2 correctly calculates the maximal Confidence Interval for a number of observations using a greedy method.*

Proof. The correctness of this algorithm is proved by mathematical induction. The greedy property is proved by contradiction.

For the base case, the number of observations are 2, there will be three scenarios:

- Both are light items: $CI1 = ConfInteCalc(N, \delta, V, L_u, L_{u+1})$
- One is light item; one is heavy item: $CI2 = ConfInteCalc(N, \delta, V, L_u, H_x)$
- Both are heavy items: $CI3 = ConfInteCalc(N, \delta, V, H_x, H_{x+1})$

The maximal confidence interval for two observations is $f(2) = max(CI1, CI2, CI3)$.

Assuming the maximal confidence interval for $m-1$ observations is $f(m-1)$ with the light items $[L_u, L_w]$ and heavy items $[H_x, H_y]$. For the next observation (the mth), it could be either a light item or a heavy item. To be more specific, it is before L_u or after L_w if it is a light item; or it is before H_x or after H_y if it is a heavy item.

For case 1, there are two light items L_{u-2}, L_{u-1} with $L_{u-2} < L_{u-1} < L_u$. Based on the advice tape analysis, the lower boundary will be $\frac{L_{u-2}}{\delta} < \frac{L_{u-1}}{\delta}$. The sample will be $C_{u-2} < C_{u-1}$. Since the mean value $\bar{C}$ is larger than the samples from the light items, the invariance $s^2 = \frac{1}{m-1}\sum_{i=1}^{m}(C_i - \bar{C})^2$ will lead to $s_{u-2} > s_{u-1}$. According to the confidence interval equation $(\bar{C} - t_{\alpha/2}(m-1)\frac{s}{\sqrt{m}}, \bar{C} + t_{\alpha/2}(m-1)\frac{s}{\sqrt{m}})$, the L_{u-2} will have larger confidence interval than the L_{u-1}. By choosing the L_i that is most less than the L_u as the next observation, $CI1 = ConfInteCalc(N, \delta, V, [L_i, L_w], [H_x, H_y])$ will be the maximal confidence interval as the next observed item is light before L_u.

For case 2, there are two light items L_{w+1}, L_{w+2} with $L_w < L_{w+1} < L_{w+2}$. Based on the advice tape analysis, the lower boundary will be $\frac{L_{w+1}}{\delta} < \frac{L_{w+2}}{\delta}$. The sample will be $C_{w+1} < C_{w+2}$. Since the mean value $\bar{C}$ is larger than the samples from the light items, the invariance $s^2 = \frac{1}{m-1}\sum_{i=1}^{m}(C_i - \bar{C})^2$ will lead to $s_{w+1} > s_{w+2}$. According to the confidence interval equation $(\bar{C} - t_{\alpha/2}(m-1)\frac{s}{\sqrt{m}}, \bar{C} + t_{\alpha/2}(m-1)\frac{s}{\sqrt{m}})$, the L_{w+1} will have larger confidence interval than the L_{w+2}. By choosing the L_i that is next larger than the L_w as the next observation, $CI2 = ConfInteCalc(N, \delta, V, [L_u, L_i], [H_x, H_y])$ will be the maximal confidence interval as the next observed item is light after L_w.

For case 3, there are two heavy items H_{x-2}, H_{x-1} with $H_{x-2} < H_{x-1} < H_x$. Based on the advice tape analysis, the upper boundary will be $\frac{H_{x-2}}{\delta} < \frac{H_{x-1}}{\delta}$. The sample will be $C_{x-2} < C_{x-1}$. Since the mean value $\bar{C}$ is less than the samples from the heavy items, the invariance $s^2 = \frac{1}{m-1}\sum_{i=1}^{m}(C_i - \bar{C})^2$ will lead to $s_{x-2} < s_{x-1}$. According to the confidence interval equation $(\bar{C} - t_{\alpha/2}(m-1)\frac{s}{\sqrt{m}}, \bar{C} + t_{\alpha/2}(m-1)\frac{s}{\sqrt{m}})$, the H_{x-1} will have larger confidence interval than the H_{x-2}. By choosing the H_i that is next less than the H_x as the next observation, $CI3 = ConfInteCalc(N, \delta, V, [L_u, L_w], [H_i, H_y])$ will be the maximal confidence interval as the next observed item is heavy before H_x.

For case 4, there are two heavy items H_{y+1}, H_{y+2} with $H_y < H_{y+1} < H_{y+2}$. Based on the advice tape analysis, the upper boundary will be $\frac{H_{y+1}}{\delta} < \frac{H_{y+2}}{\delta}$. The sample will be $C_{y+1} < C_{y+2}$. Since the mean value $\bar{C}$ is less than the samples from the heavy items, the invariance $s^2 = \frac{1}{m-1}\sum_{i=1}^{m}(C_i - \bar{C})^2$ will lead to $s_{y+1} < s_{y+2}$. According to the confidence interval equation $(\bar{C} - t_{\alpha/2}(m-1)\frac{s}{\sqrt{m}}, \bar{C} + t_{\alpha/2}(m-1)\frac{s}{\sqrt{m}})$, the H_{y+2} will have larger confidence interval than the H_{y+1}. By choosing the H_i that is most larger than the H_y as the next observation, $CI4 = ConfInteCalc(N, \delta, V, [L_u, L_w], [H_x, H_i])$ will be the maximal confidence interval as the next observed item is heavy after H_y.

In Lemma 5.1, it has already proved that under certain constrained resources and thresholds in the advice tape, given the same number of revealed light items and heavy items, the confidence interval given by the light items will be larger than the confidence interval given by the heavy items. Based on this Lemma, this algorithm can be proven to be a greedy solution by applying contradiction.

If only light items are observed to the attacker, suppose the algorithm has a solution $[L_u, L_i]$ to generate the maximal confidence interval, while there is an optimal solution assuming to have the maximal confidence interval with light items $[L_u, L_j]$. It is assumed that $L_i < L_j$. Since the $[L_u, L_j]$ has the larger confidence interval than

$[L_u, L_i]$ based on the assumption, the variance $s_{u,j}$ will be larger than $s_{u,i}$. The sample from the light items are less than the mean value $\bar{C}$ and $s^2 = \frac{1}{m-1}\sum_{i=1}^{m}(C_i - \bar{C})^2$. Then $C_{u,j} < C_{u,i}$, thus $\frac{L_j}{\delta} < \frac{L_i}{\delta}$ and $L_j < L_i$, which is contradicted with the initial assumption $L_i < L_j$.

If light items and heavy items are observed to the attacker, suppose the algorithm has a solution with light items $[L_u, L_w]$ and heavy items $[H_x, H_i]$, while there is an optimal solution assuming to have the maximal confidence interval with light items $[L_u, L_w]$ and heavy items $[H_x, H_{i-1}]$ and then H_j. It is assumed that $H_i > H_j$. Because the optimal solution has the maximal confidence interval than the solution from the algorithm, the variance $s_{u,j}$ will be larger than $s_{u,i}$. The sample from the heavy items are larger than the mean value and $s^2 = \frac{1}{m-1}\sum_{i=1}^{m}(C_i - \bar{C})^2$. Then $C_{u,j} > C_{u,i}$, thus $\frac{H_j}{\delta} > \frac{H_i}{\delta}$ and $H_j > H_i$, which is contradicted with the initial assumption $H_i > H_j$.

As the above four scenarios illustrate all the possible maximal confidence interval for m observed items and greedy proporty has been proved through contradiction, the $f(m) = max(CI1, CI2, CI3, CI4)$ is the max confidence interval for m observations.

□

5.5. EXAMPLES BY APPLYING THE ALGORITHMS

This section will applying above two algorithms to obtain the maximal Confidence Interval with corresponding number of observations. There are two different distributions of the candidates to a 0-1 knapsack problem that will be considered. For case 1, the heavy items are much larger than the light items with the satisfaction of $Heavy > \frac{Light}{\delta}$. For case 2, the heavy items are similar to the light items with the satisfaction of $Heavy < \frac{Light}{\delta}$.

5.5.1. Case 1:Heavy Items and Light Items Satisfy Heavy $> \frac{\texttt{Light}}{\delta}$.

Suppose the problem is 30, 60, 75, 200, 21, 14, 10, 9, 90, 120, 10, 17, 22, 47, 54, 108, 112 and the optimal solution for constraint 422 is 21, 14, 9, 11, 90, 120, 10, 17, 22, 108. For $\delta = 0.2$, the heavy items will be 90, 120, 108 and the light items will be 21, 14, 9, 11, 17, 10, 22. The heavy items and light items are satisfying $Heavy > \frac{Light}{\delta}$.

Applying the Algorithm 1 and Algorithm 2, the maximal Confidence Interval to each number of observations are shown in the Figure 5.7.

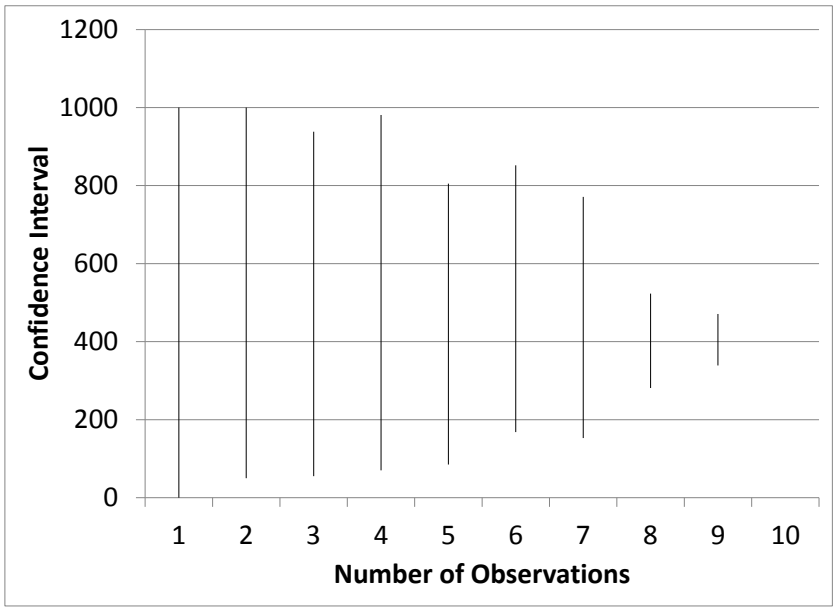


Figure 5.7: Observation to Maximal Confidence Interval for Case 1

The observations corresponding to the maximal Confidence Interval are shown in the Table 5.4.

5.5.2. Case 2:Heavy Items and Light Items Satisfy Heavy $< \frac{\texttt{Light}}{\delta}$. Suppose the problem is 30, 60, 75, 200, 47, 54, 45, 20, 50, 40, 34, 53, 33, 39, 66, 42, 112 and the optimal solution for constraint 422 is 45, 20, 50, 40, 34, 53, 33, 39, 66, 42. For $\delta = 0.1$, the heavy items will be 45, 50, 53, 66 and the light items will be 20, 40, 34, 33, 39, 42. The heavy items and light items are satisfying $Heavy < \frac{Light}{\delta}$.

Applying the Algorithm 1 and Algorithm 2, the maximal Confidence Interval to each number of observations are shown in the Figure 5.8.

The observations corresponding to the maximal Confidence Interval are shown in the Table 5.5.

The above examples and results shows that nodes that share constraint resources will show different information leakage patterns as partial have been observed. The information leakage patterns that are defined by calculating the Confidence Interval are depending on the value distribution of the nodes and the sequence of the observed

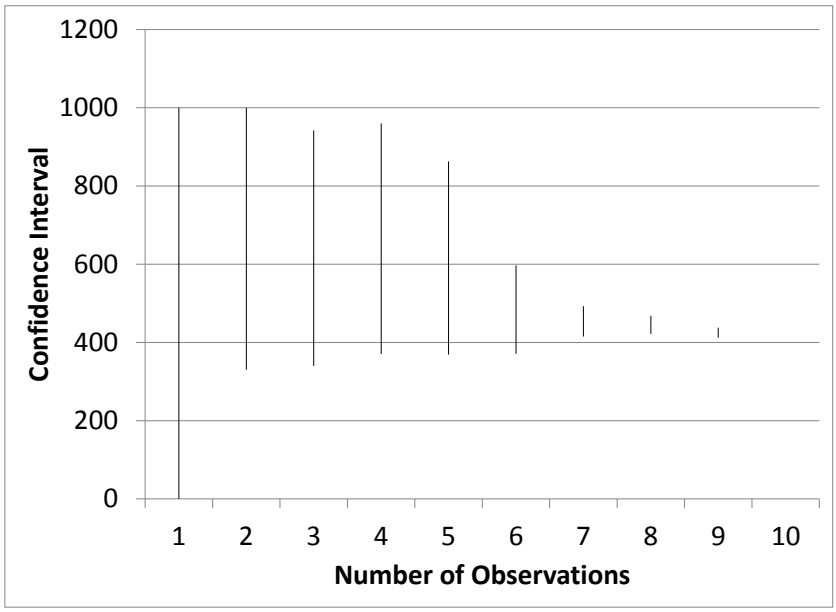


Figure 5.8: Observation to Maximal Confidence Interval for Case 2

nodes. To be more specific, nodes that $Heavy < \frac{Light}{\delta}$ will reveal less information compared with nodes that $Heavy > \frac{Light}{\delta}$ as the light items are observing first.

Considering the above conclusion, security strategies for power migration as the 0-1 simple knapsack is the algorithm could be proposed as choosing candidates nodes with distribution of $Heavy > \frac{Light}{\delta}$ instead of $Heavy < \frac{Light}{\delta}$, allowing power migration first happening to satisfying the light items with chunk unit, and allowing unit power migration to the heavy items in the later phase.

Algorithm 1: Information Leakage Quatification with Advice Tape: ConfInteCalc(N, δ, V, V_i or $V_1, V_2, ..., V_k$)

input : System N, δ, all nodes value $V[N]$, observed one node V_i or node set $V_1, V_2, ..., V_k$
output: Confidence Interval to the input
S_{heavy} is the min Heap;
S_{light} is the max Heap;
$SUM = V_1 + V_2 + ... + V_N$;
if $n == 1$ **then**
  if $V_i == S * \delta$ **then**
    Max information leakage to the secret $S = \frac{V_i}{\delta}$;
  else if $V_i > S * \delta$ **then**
    $S_{heavy}.push(V_i)$;
  else
    $S_{light}.push(V_i)$;
  end
else
  for $j \leftarrow 1$ **to** k **do**
    $partialSUM+ = V_j$;
    if $V_i == S * \delta$ **then**
      Max information leakage to the secret $S = \frac{V_i}{\delta}$;
    else if $V_j > S * \delta$ **then**
      $S_{heavy}.push(V_j)$;
    else
      $S_{light}.push(V_j)$;
    end
  end
  $Secret_{upper} \leftarrow MIN(\frac{S_{heavy}.top()}{\delta}, SUM)$;
  $Secret_{lower} \leftarrow MAX(\frac{S_{light}.top()}{\delta}, partialSUM)$;
  if $heavy_{min} > \frac{light_{max}}{\delta}$ **then**
    $Secret_{lower} \leftarrow \text{MAX}(heavy_{min}, Secret_{lower})$;
  end
  The boundary of the secret will be $(Secret_{lower}, Secret_{upper})$;
  choose m samples $C_1, C_2, ..., C_m$ *from the boundary of the secret*;
  calculate $\bar{C} = \frac{C_1 + C_2 + ... + C_m}{m}$;
  claculate $s^2 = \frac{1}{m-1} \sum_{i=1}^{m} (C_i - \bar{C})^2$;
  obtain confidence interval $(\bar{C} - t_{\alpha/2}(m-1)\frac{s}{\sqrt{m}}, \bar{C} + t_{\alpha/2}(m-1)\frac{s}{\sqrt{m}})$;
end

Algorithm 2: Maximal Information Leakage with m corresponding observations: maxConfInte()

input : System N, δ, all nodes value $V[N]$
output: Maximal Confidence Interval $f(m)$ with Corresponding m observations, light from L_u to L_w, heavy from H_x to H_y

for *2 observations* **do**
- /* both are light items */
- $O_1 = (L_u, L_w)$;
- $Boundary_{O_1} = (MAX(L_u + L_v, \frac{L_u}{\delta}, \frac{L_v}{\delta}), SUM)$;
- $CI1 = ConfInteCalc(N, \delta, V, u, w) = (\bar{C} - Z_{\frac{\alpha}{2}} \times \frac{s}{\sqrt{2}}, \bar{C} + Z_{\frac{\alpha}{2}} \times \frac{s}{\sqrt{2}})$;
- /* one is light item and one is heavy item */
- $O_2 = (L_u, H_x)$;
- $Boundary_{O_2} = (MAX(L_u + H_x, \frac{L_u}{\delta}), MIN(\frac{H_x}{\delta}, SUM))$;
- $CI2 = ConfInteCalc(N, \delta, V, u, x) = (\bar{C}' - Z_{\frac{\alpha}{2}} \times \frac{s'}{\sqrt{2}}, \bar{C}' + Z_{\frac{\alpha}{2}} \times \frac{s'}{\sqrt{2}})$;
- /* both are heavy items */
- $O_3 = (H_x, H_y)$;
- $Boundary_{O_3} = ((H_x + H_y), MIN(\frac{H_x}{\delta}, \frac{H_y}{\delta}))$
- $CI3 = ConfInteCalc(N, \delta, V, x, y) = (\bar{C}'' - Z_{\frac{\alpha}{2}} \times \frac{s''}{\sqrt{2}}, \bar{C}'' + Z_{\frac{\alpha}{2}} \times \frac{s''}{\sqrt{2}})$;

end

$f(2) = max(CI1, CI2, CI3)$;

for $i \leftarrow 3$ **to** m **do**
- /* {Loop Invariant: $f(i-1)$ is the maximal confidence interval for $i-1$ observations} */
- Case 1: Pick one observation that is most less than L_u;
- /* {$L_i <$ lower bound of light item L_u of $f(i-1)$} */
- $CI1 = ConfInteCalc(N, \delta, V, [L_i, L_w], [H_x, H_y])$;
- Case 2: Pick one observation that is next larger than L_w;
- /* {$L_i >$ upper bound of light item L_w of $f(i-1)$} */
- $CI2 = ConfInteCalc(N, \delta, V, [L_u, L_i], [H_x, H_y])$;
- Case 3: Pick one observation that is next less than H_x;
- /* {$H_i <$ lower bound of heavy item H_x of $f(i-1)$} */
- $CI3 = ConfInteCalc(N, \delta, V, [L_u, L_w], [H_i, H_y])$;
- Case 4: Pick one observation that is most larger than H_y;
- /* {$H_i >$ upper bound of heavy item H_y of $f(i-1)$} */
- $CI4 = ConfInteCalc(N, \delta, V, [L_u, L_w], [H_x, H_i])$;
- $f(i) = max(CI1, CI2, CI3, CI4))$;
- /* {$f(i)$ is the maximal confidence interval for i observations} */

end

Table 5.4: Observations to Maximal Confidence Interval for Case 1

Num	Observed Items	Maximal Confidence Interval
1	N/A	N/A
2	9, 10	(50, 1000)
3	9, 10, 11	(55, 938)
4	9, 10, 11, 14	(70, 981)
5	9, 10, 11, 14, 17	(85, 805)
6	9, 10, 11, 14, 17, 21	(168, 852)
7	9, 10, 11, 14, 17, 21, 22	(153, 771)
8	9, 10, 11, 14, 17, 21, 22, 120	(281, 523)
9	9, 10, 11, 14, 17, 21, 22, 120, 108	(379, 471)
10	11, 10, 21, 14, 17, 90, 22, 9, 108, 120	422

Table 5.5: Observations to Maximal Confidence Interval for Case 2

Num	Observed Items	Maximal Confidence Interval
1	N/A	N/A
2	20, 33	(330, 1000)
3	20, 33, 34	(340, 942)
4	20, 33, 34, 39	(370, 960)
5	20, 33, 34, 39, 40	(369, 863)
6	20, 33, 34, 39, 40, 66	(391, 597)
7	20, 33, 34, 39, 40, 66, 53	(415, 493)
8	20, 33, 34, 39, 40, 66, 53, 50	(422, 468)
9	20, 33, 34, 39, 40, 66, 53, 50, 45	(412, 438)
10	45, 20, 50, 40, 34, 53, 33, 39, 66, 42	422

6. INTEGRATING WITH PHYSICAL SYSTEM

Correctness, stability and safety are most important for the CPS systems and the CPS must be able to maintain their physical properties. In computer science, invariants [6] are logical statements that must remain true throughout a system's execution. In recent research, the invariant approach has been shown its effectiveness in a smart grid CPS to ensure the system stability [71]. Through developing invariants for each individual subsystem such as physical and network subsystems in a CPS and regarding a unified invariant as a guard in the cyber algorithm, the CPS will be safe and stable at system level. This section will analyze the influence of the physical invariant to the Confidence Interval based quantification information flow approach in FREEDM. Through updating the cyber algorithm, FREEDM could be both stable and more secure.

6.1. INVARIANT FOR THREE NODES SWITCHED SYSTEM

Lyapunov function $V(x)$ [18] is well known for stability analysis of an autonomous continuous system. $V(x)$ is defined as positive definite, radially unbounded and non-increasing. The system is stable if its derivative is non-positive; the system is asymptotically stable if its derivative is strictly negative. A Lyapunov-like function [15] works similarly but is more flexible since it doesn't require its derivative to be negative. Multiple Lyapunov-like functions can be used for different operating modes. As the cyber system evolves, the Lyapunov-like function is checked and its derivative is monitored. As a Lyapunov or Lyapunov-like function captures the dynamics between the switching instants, it can be developed as an invariant for the switched power system. By checking the result of this invariant, the cyber portion can determine whether or not to act at a switching time.

A simplified version of FREEDM was implemented with the physical systems model simulated using PSCAD as shown in Figure 6.1. The system is comprised of three converters (all in three phase) interconnected through lines of impedance and eventually connected to a generator by a transformer. Among the three converters, two converters inject power (source) into the grid and act as inverters. One converter delivers power out of the grid (load), thereby acting as a rectifier [71].

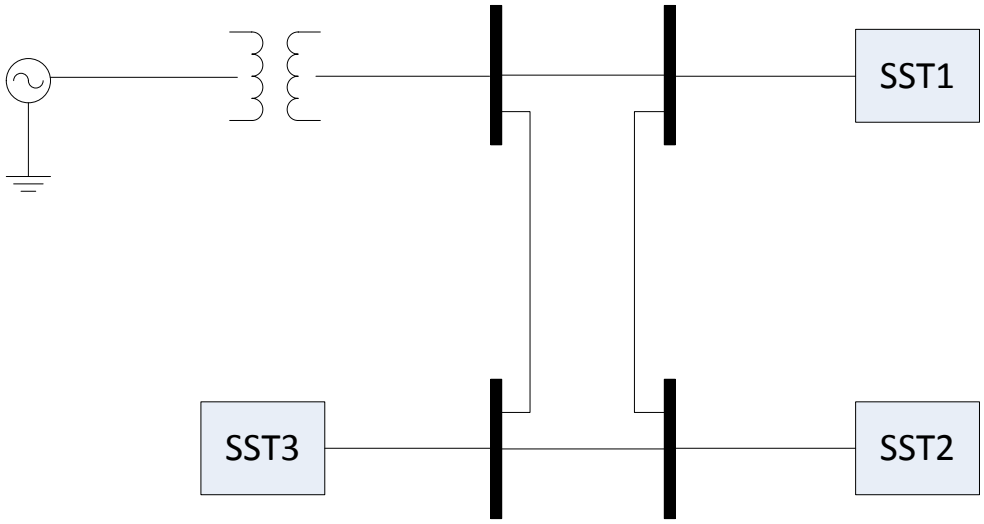


Figure 6.1: Three Nodes Switched System Topology

In this switched system, a Lyapunov-like function is defined to composed of the error in stored energy. The objective is to formulate an invariant that may be used in the cyber analysis. If asymptotic stability is considered, Equation 47 is proposed as the physical invariant for FREEDM [71]. ω is the grid frequency, ω_0 is the nominal frequency, D, k, m are const parameters, δK is the net imbalance in power (the difference between the total generated power and the total load power) that will tend to instability of the frequence.

$$\left\{I_{P1} : (\omega - \omega_0)^2(D\omega + m) + (\omega - \omega_0)(kP^2) > \delta K(\omega - \omega_0)\right\} \tag{47}$$

The cyber portion that runs Power Balance Algorithm must guard its changes from causing instability in the physical system by checking I_{P1}. In Figure 6.2a, knapsack based Power Balance Algorithm performs migrations at times 9 and 14 (upper plot) eventually driving the system to instability (system frequency in the lower plot). In Figure 6.2b, knapsack based Power Balance Algorithm prevent the migration at

time 14 from occurring that would have driven the system unstable by checking the physical invariant in the cyber algorithm.

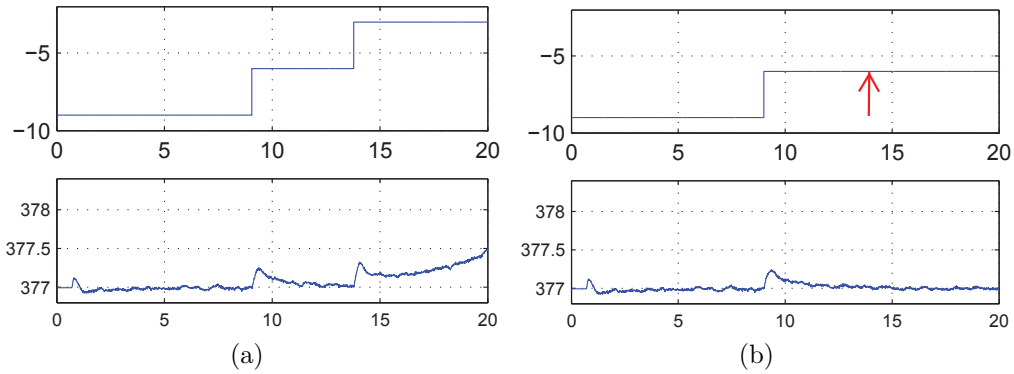


Figure 6.2: Simulated microgrid performance in response to two Power Balance Algorithms. Case (a) indicates instability due to excessive power migrations, whereas in case (b) the red arrow indicates that Power Balance Algorithm finally guards the migration, preventing instability.

The above experimental result shows that the Lyapunov-like invariant could restrict the behavior of the power system, hence, by doing so it changes the algorithm's behavior and make the system stable. The following section shows that the physical invariant will influence the information leakage analysis in an available power system.

6.2. INVARIANT FOR TWO-GENERATORS SEVEN NODES SYSTEM

Power system correctness requires that no power lines are overloaded all the time. On the other hand, if a transaction between entities result in one or more lines being overloaded, system correctness won't be preserved. The physical system is changed with two generators in Figure 6.3.

The concept of Available Transfer Capability (ATC) has been used to ensure appropriate power transactions in the past. ATC has the property that can be explored to determine if the power change between potential entities would compromise the system stability or not. The ATCs are proposed to act as line flow invariant to monitor the system correctness in [33] and [70]. The invariant is formulated as the proposed transaction should be less than the ATC for any two transacting parties.

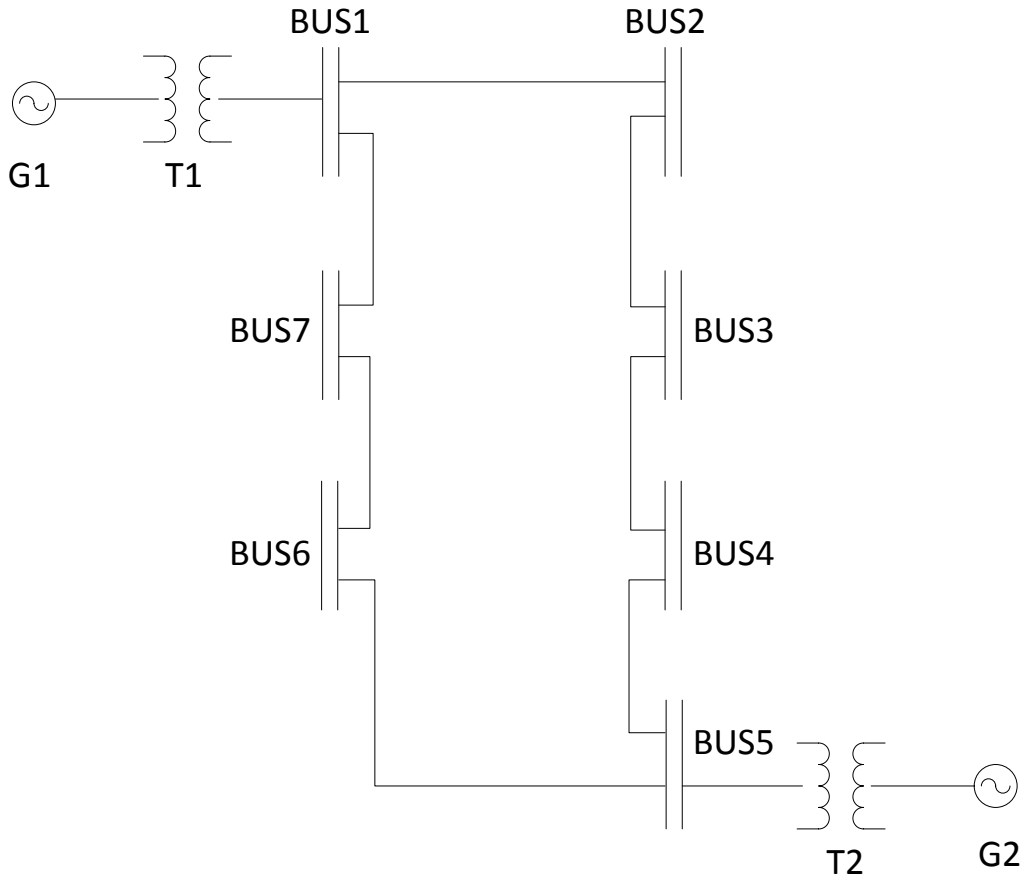


Figure 6.3: Two Generators Seven Nodes System Topology

The ATC is evaluated by realizing the new flow on the line connecting entities i and j due to a transaction between buses m and n. The new flow on the line connecting entities i and j is the sum of the original base flow of the line and the change due to the new transaction between entities m and n. The maximum power that can be transferred between any two transacting entities without overloading a certain transmission line ij is defined as P_{mn}^{New}. Thus, the maximum power that can be transferred between any two transacting entities m and n for any combination of ij in the system can be evaluated. The minimum of these values gives the maximum allowable transaction between any two sub-systems without overloading a line. This can be mathematically represented as [48]:

$$\{ATC_{mn} = Min(P_{ij,mn}^{Max}) \ \forall \ ij\} \tag{48}$$

The ATCs for every pair of entities were computed in advances and then used as a guard in the power migration algorithm. The cyber algorithm will be adapted by adding an invariant check in step4 as shown in Table 6.1.

Table 6.1: Load Balance Algorithm for DGI with Invariant Check

1. A node in the Supply state will advertise a draft request message to the Demand nodes in its load table. 2. On receiving the draft request from Supply node, the Demand node responds with a message called draft age indicating requested power. 3. The Supply node will select one Demand node based on the greedy solution to the knapsack problem. 4. **The Supply node will check the invariant and send out a draft select message to the Demand node if the invariant is true.** 5. Upon receiving the draft select message, the Demand node will initiate power migration by increasing its local SST and send back an accept message to the Supply node. 6. Then the Supply node will finish the power migration by decreasing its local SST.

The results in [33] and [70] shows that the line flow invariant was not only successful in determining whether any line in the system could be overloaded due to the next transaction, but also stop such a transaction for the system stability.

6.3. CONFIDENCE INTERVAL ANALYSIS WITH INTEGRATION OF PHYSICAL SYSTEM INVARIANT

As the invariant indicates the characteristic of the physical system, it represents the correctness requirements for the system. The above discussion shows that the physical systems in the CPSs could be extracted an invariant that can be evaluated to accommodate dynamic behavior and prevent unanticipated events. To be more specific, the line flow invariant based on the ATC has been a logical guard integrated in the cyber algorithm in FREEDM for the system correctness. This section will show that the confidence interval will become less due to the physical constraint.

Based on the results from the previous section, the items in the solution for the knapsack problem that satisfy $Heavy > \frac{Light}{\delta}$ will have more uncertainty than the items that satisfy $Heavy < \frac{Light}{\delta}$ as the observation is increasing. That is, for any supply node, when choosing the demand nodes to migrate power, it will choose the candidates that satisfy $Heavy > \frac{Light}{\delta}$ and migrating the light demand first. The ATC based line flow invariant indicates that the proposed transaction should be less than the ATC for any two transacting parties in order to make sure the system is stable.

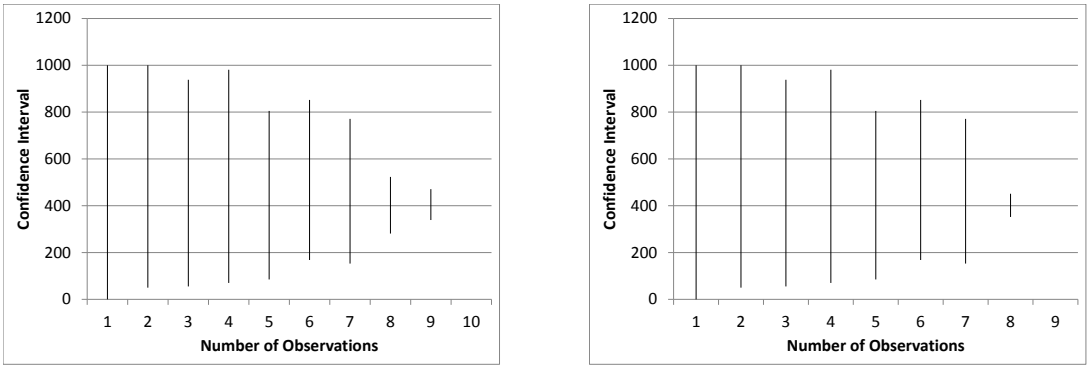
Observation 6.1. *The ATC based line flow invariant that indicates a threshold transaction for the flow based CPSs does not increase the confidence interval for the observer.*

Proof. Assuming the invariant indicating the threshold transaction is less than the heavy items but larger than the light items. If the observer see the transaction seems stop in the middle, the observer will conclude that there will be at least one heavy item that is larger than the invariant is in the solution. As the smallest heavy item h_s that is larger than the invariant will be picked for next observation, the variance s^2 will be smaller since the upper bound of the sample C is decided by $\frac{h_s}{\delta}$. Thus, the confidence interval will be smaller than any other heavy items larger than the h_s for the next observation. □

Several knapsack problems have been conducted Confidence Interval analysis without and with ATC. The results in Table 6.2 show that Confidence Interval is decreased as the ATC is applied. The distribution of $Heavy > \frac{Light}{\delta}$ produce much more Confidence Interval decrease than the distribution of $Heavy < \frac{Light}{\delta}$. First knapsack problem satisfying $Heavy > \frac{Light}{\delta}$ is 30, 60, 75, 200, 21, 14, 10, 9, 90, 120, 10, 17, 22, 47, 54, 108, 112 with optimal solution for constraint 422 is 21, 14, 9, 11, 90, 120, 10, 17, 22, 108. Second knapsack problem satisfiying $Heavy < \frac{Light}{\delta}$ is 47, 54, 45, 20, 50, 30, 60, 75, 111, 89, 40, 34, 53, 33, 39, 66, 42, 100, 12 and the optimal solution for constraint 422 is 45, 20, 50, 40, 34, 53, 33, 39, 66, 42. Third knapsack problem satisfying $Heavy > \frac{Light}{\delta}$ is 7, 2, 12, 33, 59, 73, 14, 9, 6, 95, 70, 10, 4, 13, 3, 19, 82, 76, 135, 65, 46, 50 and optimal solution for constraint 422 is 7, 2, 12, 14, 9, 6, 95, 70, 76, 10, 4, 13, 3, 19, 82. Figure 6.4 shows the comparison of no ATC and ATC guarded confidence interval analysis for No.1 knapsack problem in Table 6.2.

Table 6.2: Results of Reduced Confidence Interval due to ATC

No.	δ	$Heavy > \frac{Light}{\delta}$	ATC	Migration Stop Index	Reduced Confidence Interval	Reduced Percentages
1	0.2	Yes	60	8	[281, 523]→[352, 451]	60%
2	0.08	No	42	7	[415, 493]→[420, 438]	77%
3	0.15	Yes	60	12	[281, 734]→[348, 440]	77%



(a) (b)

Figure 6.4: Observation to Confidence Interval for (a) Line Overloading Case without ATC (b) ATC Invariant Guarded Case

The cyber algorithm can make changes that will generate more uncertainty to the observer as well as maintain the system correctness. The Power Balance algorithm in FREEDM has been updated by comparing the candidate items with the computed ATC in the physical system as follows:

- If the light items are less than the ATC, the light items will be kept in the solution.
- If the heavy items are larger than the ATC, they cannot be migrated due to the line overload and reduced confidence interval. As the heavy items are splitted into smaller parts that are less than the ATC, system correctness is maintaind as well as the confidence interval is increased.

The following experimental results show that after considering the computed ATC from the physical system, the confidence interval will decrease compared with no ATC consideration in FREEDM. As the heavy items that are larger than the ATC are splitted and the observer might regard a split item from a heavy item as a light item, the uncertainty has increased. Suppose the optimal solution for constraint 422 is 7, 2, 12, 6, 8, 9, 11, 90, 120, 10, 4, 13, 3, 19, 108. For $\delta = 0.2$, the heavy items will be $90, 120, 108$ and the light items will be 7, 2, 12, 6, 8, 9, 11, 10, 4, 13, 3, 19. The heavy items and light items are satisfying $Heavy > \frac{Light}{\delta}$. The maximum confidence interval will be Figure 6.5 with the line overloading happens in the physical system.

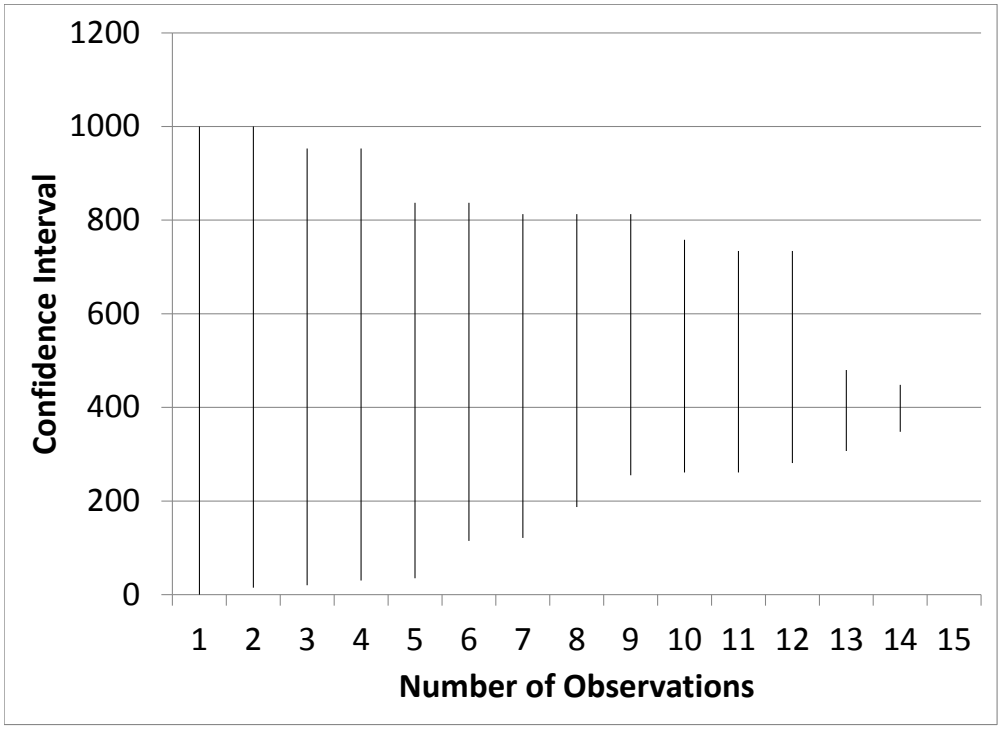


Figure 6.5: Line Overloading Case Without ATC

In order to prevent the line overload, the algorithm will stop the power migration if the ATC invariant is checked to be false. Assuming the ATC = 60, the observations

will stop after the light items have all shown up. Once the power migration stops, the observer could conlcude that one from $90, 108, 120, 200$ would be in the solution since they are all larger than the ATC. The maximum confidence interval is shown in Figure 6.6 with the power migration stop at 12 to prevent line overload.

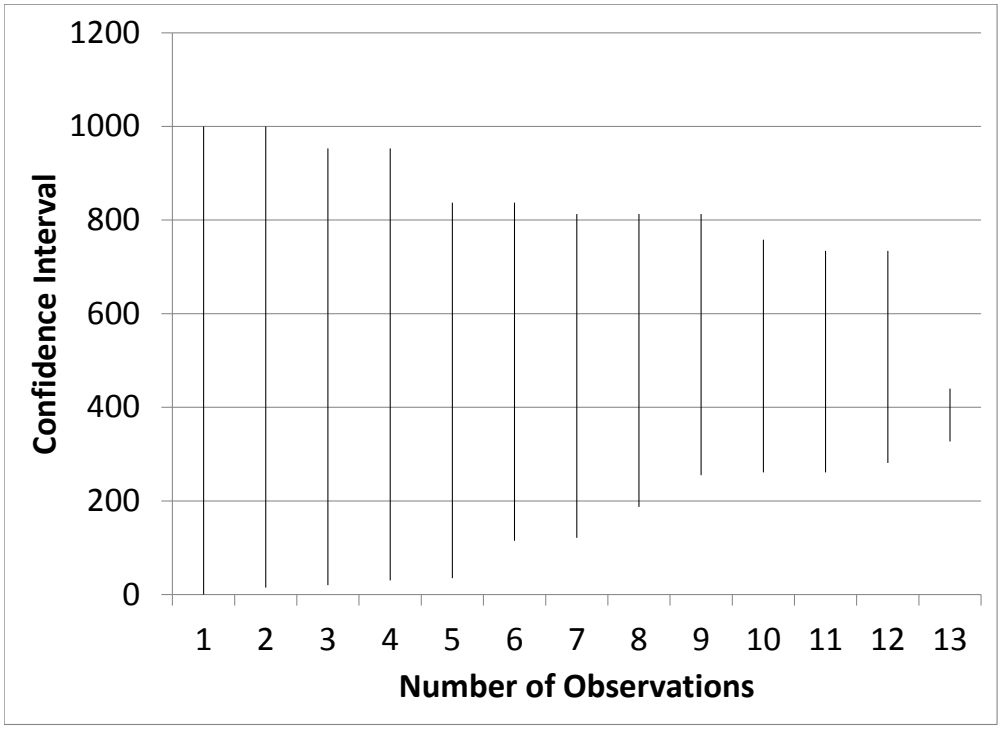


Figure 6.6: ATC Invariant Guarded Case

The heavy item $90, 120, 108$ will need to be divided into small parts in order to obtain the stability of the system. Supposing the 90 changes to $40, 50$, 120 changes to $40, 40, 40$ and 108 changes to $50, 58$. If the small parts of the heavy items in the power line have been observed by the attacker and considered as the light items, the maximal confidence interval will be Figure 6.7.

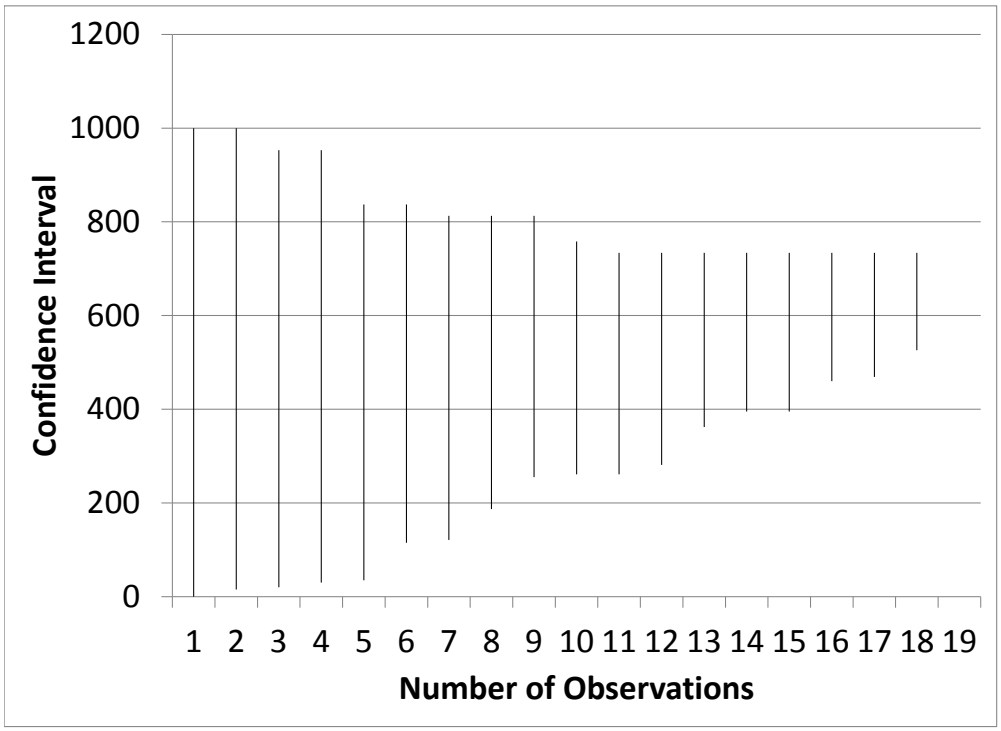


Figure 6.7: ATC Invariant Guarded Case with Splitting

It is obvious from the above experiment that by considering the characteristics of the physical system that is represented by the ATC invariant, the cyber algorithm could be updated to bring more uncertainty to the observer.

Theorem 6.1. *Without the ATC invariant, the power system will be overloaded. As the ATC invariant is applied as a guard before each power migration, the observer will get more certainty of the supply. If the heavy items are split to items that are less than the ATC invariant, the algorithm can produce a larger Confidence Interval.*

Proof. Suppose revealed heavy items are $H_1, H_2, ..., H_i$ and revealed light numbers are $L_1, L_2, ..., L_j$ under threshold δ. Assuming there are k heavy items that are larger than the ATC. In order to maintain the correctness of the physical system, those k heavy items should be divided into pieces that are less than the ATC. Suppose those k heavy items have been divided into k' light items that are less than the ATC. Thus,

$k' > k$. Based on Lemma 5.1, given the same number of revealed light items and heavy items, the confidence interval given by the light items will be larger than the confidence interval given by the heavy items. After splitting k heavy items into k' light items, any k new light items from k' items will generate larger Confidence Interval than the k original heavy items. Thus more uncertainty will be given to the observer through the splitting opration on the heavy items. □

7. CONCLUSIONS AND FUTURE WORK

7.1. CONCLUSIONS AND CONTRIBUTIONS

Traditional security mechanisms are not enough for the CPSs due to complicated interactions that happen frequently between the cyber portion and physical portion. Information flow based quantification that aims at control information from end to end is necessary for security analysis in CPSs. The observability problem in CPSs is defined as how to quantify the uncertainty to the observer as parts of the secret in the system are showing up and aggregating.

Given the background of security and related work on quantifying information flow, this dissertation first introduces the problem statement as well as preliminary results analyzing information leakage. A framework to quantify information flow in constrained resource CPSs is proposed. The assumption for the constrained resource is that its distribution is known to the observer. Physical observations provide a range of the constrained resource values, quantified by applying advice tape analysis. Quantifying information is accomplished by calculating the confidence interval produced by the observations. The algorithms are presented in details for the calculation of leaked information with the increased number of observations; results from cases study show the preference of the specific distribution.

Physical conservation is the natural law that exists in any CPSs. For example, in any flow based CPS, the physical conservation states the balance between the in and the out, the generation and the demand. Without physical invariant, the system would have poor behavior that will lead to system unstable. Through developing an invariant of the physical system to express the correctness of the system, it will make sure that the system is stable. It will also lead to a way of updating the cyber algorithm to obtain more uncertainty to the observer. This dissertation shows that the proposed framework and algorithm are integrated with invariant approach that is effective in ensuring the stability of the CPSs. By considering the characteristics of the physical portion as an invariant, the proposed information quantification method would generate more appropriate Confidence Interval to the observer in a CPS. FREEDM has been used as an example for demonstrate usage of this framework in a real CPS.

The key use of this result is that the algorithm can allocate power such that the uncertainty of the supply value is maintained, this limiting an observer's knowledge. For example, in constrained flow based CPS such as FREEDM, the number of items sharing the resource is very important. If the values of all items are available just like the Load Table, the information leakage can be quantified using advice tape analysis. If only one demand node exists, it will know the value of the constrained resource. If more than two demand nodes exist, the constrained resource will be shared among demand nodes. Both the number of demand nodes and the values of the demand nodes impact the uncertainty. The given algorithms can calculate the maximized uncertainty to the observer based on the aggregated observations.

7.2. FUTURE WORK

Future work can happen in many exciting directions. Some of the thoughts are enumerated as follows.

One is to apply this work to trust management with a defined trust function under the constrained resource system. How can observations through advice tape analysis be used for trust evaluation in the constrained resource CPS? How does one maximize uncertainty to an observer while maximizing trust among the demand and supply nodes?

Another interesting investigation might be exploring advice tape generation method for general online problems that only partial inputs are known to the public. By knowing the compositions of the advice tape to the online problem, statistical model such as confidence interval will work well in analyzing the information leakage of the constrained resources as the secret input due to the aggregation of the observed outputs.

The work in this dissertation would be a first step towards information flow quantification considering the integrations of the cyber algorithm with the physical constraint. By understanding the dynamics of CPSs and extracting the physical invariant that could express the physical law, information flow analysis would affect the cyber algorithm and lead to a more secure way for the CPSs.

8. ZERO KNOWLEDGE PROOFS

Zero Knowledge Proofs (ZKPs) are interactive protocols in which one party, named the prover, can convince the other party, named the verifier, that some assertion is true without revealing anything other than the fact that the assertion being proven is true. This chapter is a survey on ZKPs including their background, important concepts, applications for NP problems, and composition operations of ZKPs. The remarkable property of being both convincing and yielding nothing except the assertion is indeed valid makes ZKPs very powerful tools for the design of secure cryptographic protocols. In this chapter, ZKPs are constructed for the exact cover and 0-1 simple knapsack problem. It was explored for security protocol design for the CPSs, however, due to the iterative property in nature of the existing ZKPs and multiple communication rounds required, they are not appropriate security protocols for the CPSs in which information exchanges are not iterative between the cyber parts and the physical parts. This section is included as the ZKP for the knapsack is original work and published in [30].

8.1. BACKGROUND AND MOTIVATION

In order to deal with the problem of distrustful parties convincing each other that the message they are sending is indeed computed according to their predetermined procedure without yielding any secret knowledge, a number of privacy-preserving communication protocols have been proposed and designed, including for example, the fuzzy vault scheme, k-anonymity, nondeducibility, ZKPs, and so on.

Shamir provided a scheme to share a secret in [78]. Rather than straightforward encryption, the main idea is to divide data D into n pieces in such a way that D is easily reconstructible from any k pieces, but even combining k-1 pieces won't reveal any information about D. This scheme is called a (k, n) threshold scheme. Threshold schemes are ideally suited to applications in which a group of mutually suspicious individuals with conflicting interests must cooperate.

The fuzzy vault is a novel cryptographic notion developed by A. Juels and M. Sudan in [54], which can be considered as a form of error-tolerant encryption operation. In a fuzzy vault scheme, one party places a secret value k in a fuzzy vault and

"locks" it using a set A of elements from some public universe U; the other party can "unlock" the vault to get the secret value k using a set B of similar length only if A and B overlap substantially. A fuzzy fault scheme can be constructed based on any type of linear error-correcting code.

Another privacy-preserving framework for communication is called k-anonymity. A data record is k-anonymous if and only if it is indistinguishable in its identifying information from at least k specific records. k-anonymity provides a formal way of preventing re-identification of data to fewer than a group of k data items. k-anonymity provides a formal way of generalizing aggregated data without violating the privacy [53].

Nondeducibility is a security model based on information theory, especially for information sharing. The model states that information flows in a system from high-level objects to low-level objects if and only if some possible assignment of values to the low-level objects in the state is incompatible with a possible assignment of values to the state's high-level objects [63]. To be more specific, a system is considered nondeducible secure if it is impossible for a low-level user, through observing visible events, to deduce anything about the sequence of inputs made by a high-level user.

The notion of ZKPs, which is first introduced by Goldwasser, Micali and Rackoff [45], is to obtain a system in which it is possible for a prover to convince a verifier of his knowledge of an assertion without disclosing any information except the validity of his assertion. ZKP study has become a very active research area because of its fascinating nature of a seemingly contradictory definition; ZKPs have the remarkable property of being both convincing and yielding nothing except that the assertion is indeed valid, which makes them very powerful tools for the design of a secure cryptographic protocol [42].

8.2. INTRODUCTION

8.2.1. Interactive Proof and Its Properties. Before presenting the definition of ZKPs, we first introduce the notion of an overarching class, the interactive proof.

Definition 8.1. *(Interactive Proof [45]) An interactive proof system for a language L is a pair of interactive Turing Machines, (P, V) in which V, the verifier, executing a*

probabilistic polynomial-time strategy and P, the prover, executing a computationally unbounded strategy, satisfy two properties: completeness and soundness.

- Completeness can be expressed as:

$$Prob[(P,V)(x) \;=\; 1 \mid x \in L] \;\geq 1 - |x|^{-c} \;\; (c > 0)$$

- Soundness can be expressed for every interactive Turing machine P^*as:

$$Prob[(P^*,V)(x) \;=\; 0 \mid x \notin L] \;\geq 1 - |x|^{-c} \;\; (c > 0)$$

Completeness means that the verifier accepts the proof if the statement or assertion is true. In another words, an honest verifier will always be convinced of a true statement by an honest prover. Soundness means if the fact is false, the verifier rejects the proof, which indicates that a cheating prover can convince an honest verifier that some false statement is actually true with only a small probability. Both conditions allow a negligible error probability, which plays an important role in ZKPs with a robust notion of rareness; a rare event should occur rarely even if we repeat the experiment for a feasible number of times [38].

Proofs here do not represent the traditional mathematical concept of a proof. Mathematical proofs are strict, using either self evident statements obtained or from proofs established beforehand. Here, proofs are not a fixed and static object; they are similar to the dynamic process used to establish the truth of a statement throughout the exchange of information. Thus, they are considered to be probabilistic rather than absolute. The class of problems having interactive proof systems is denoted as IP [38].

8.2.2. A Simple Example. "In cryptography, the zero-knowledge protocol is an interactive method for one party to prove to another that a statement is true, without revealing anything other than the veracity of the statement [42]". Each party in the protocol does a sequence of actions. While receiving a message from the other party, it performs a private computation, then it sends a message to the other party. Those actions will be repeated many rounds. Then the verifier either accepts or rejects the prover's proof.

A very simple example of a ZKP could be illustrated as a story of Ali Baba's Cave [2]. In this story Figure 8.1, Peggy (the prover) has uncovered the secret word

used to open a magic door in a cave. The cave is shaped like a circle, with the entrance on one side and the magic door blocking the opposite side. Victor (the verifier) says he'll pay her for the secret, but not until he's sure that she really knows it. Peggy says she'll tell him the secret, but not until she receives the money. They devise a scheme by which Peggy can prove that she knows the word without telling it to Victor. Peggy goes into a random branch of the cave without letting Victor knowing which branch she chose. Standing at the entrance to the cave, Victor calls out a random branch he wants Peggy to come out from. If Peggy indeed knows about the secret password, she can obey every time. If she doesn't know the secret password, she has a 50% of initially fooling Victor. If Victor is happy with a 1 in 1024 chance that Peggy is cheating, there will be 10 iterations since $\frac{1}{2^{10}} = \frac{1}{1024}$.

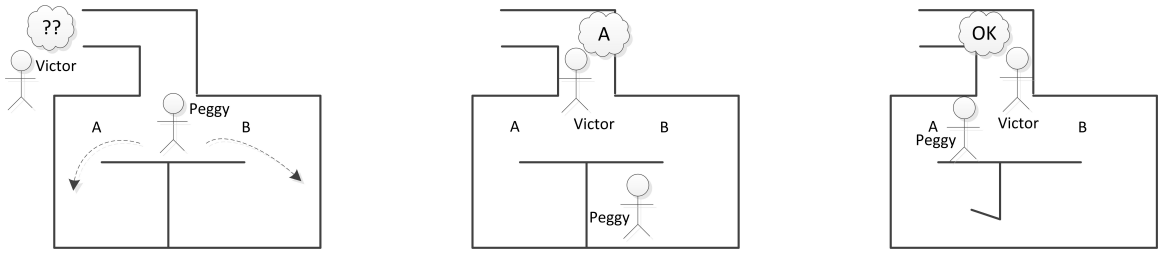


Figure 8.1: Ali Baba's Cave

8.2.3. Computational Indistinguishability. "Effective similarity" is a widely accepted notion in Modern Cryptography. If the differences between the objects can't be observed through a feasible computation, then the two objects are considered equivalent. In the definition of zero knowledge, it is impossible to tell the differences among computationally indistinguishable objects. The notion of computational indistinguishability underlies the definition of general zero-knowledge [38].

For $S \subseteq \{0,\ 1\}^*$, the probability ensembles $X = \{X_\alpha\}_{\alpha \in S}$ and $Y = \{Y_\alpha\}_{\alpha \in S}$ contain X_α and Y_α, which is a distribution that ranges over strings of length polynomial in $|\alpha|$. A polynomial-size circuit $\{D_n\}$ means there exists a deterministic Turing Machine which has a running time polynomial in n on the circuit. $\Pr[D_n(X_\alpha) = 1]$ denotes the probability that circuit D_n outputs 1 on input X_α.

Definition 8.2. *(Computational Indistinguishability [44], [86]) $X = \{X_\alpha\}_{\alpha \in S}$ and $Y = \{Y_\alpha\}_{\alpha \in S}$ are computationally indistinguishable if for every family of polynomial-size circuits $\{D_n\}$, every polynomial p, and sufficiently large n and every $\alpha \in \{0,\ 1\}^n \cap S$,*

$$|\Pr\left[D_n\left(X_\alpha\right) = 1\right]\ - \Pr\left[D_n\left(Y_\alpha\right) = 1\right]\ | < \frac{1}{p(n)}$$

where the probabilities are taken over the relevant distribution.

8.2.4. One-way Function. In general, Modern Cryptography is always concerned with a question of whether one-way functions exist. One-way functions provide us the equivalent of digital lockable boxes. They are functions that are easy to evaluate but hard (on the average) to invert, which has an intractability characteristic.

Definition 8.3. *(One-way Function [38]) A function $f\ : \{0,\ 1\}^* \to \{0,\ 1\}^*$ is called one-way if the following two conditions hold:*

1. *Easy to evaluate: There exists a polynomial-time algorithm A such that $A\left(x\right) = f(x)$ for every $x \in \{0,\ 1\}^*$.*
2. *Hard to invert: For every family of polynomial-size circuits $\{C_n\}$, every polynomial p, and all sufficiently large n,*

$$Pr[C_n\left(f\left(x\right)\right) \in f^{-1}\left(f\left(x\right)\right)] < \frac{1}{p(n)}$$

where the probability is taken uniformly over all the possible choices of $x \in \{0,\ 1\}^n$.

Because the problem of factoring large integers is computationally intractable, the function that takes as input two (equal length) primes and outputs their product is widely believed to be a one-way function [53].

8.2.5. Simulation Paradigm. The simulation paradigm is another important notion, which was first developed in the context of zero-knowledge, presented by Goldwasser, Micali and Rackoff [45]. The crucial point is that for every algorithm that represents the strategy of the verifier, there exists a simulator that can simulate the entire interaction of the verifier and the honest prover without access to the prover's auxiliary information. It is an approach which makes the adversary gains nothing if

whatever it can obtain by unrestricted adversarial behavior can be obtained within essentially the same computational effort as the prover. In this way, the adversary gains nothing.

A simulator is defined as a method or procedure that generates fake (generated without the prover) views that are indistinguishable from a genuine (generated with the prover) view of a proof [79]. To be more specific, whatever a verifier might have learned from the interaction with the prover, he could have actually learned by himself by running the simulator. Figure 8.2 illustrates the idea. The concept of a simulator helps defining the zero knowledge property in another way, that is, a proof of knowledge has the zero knowledge property if there exists a simulator for the proof.

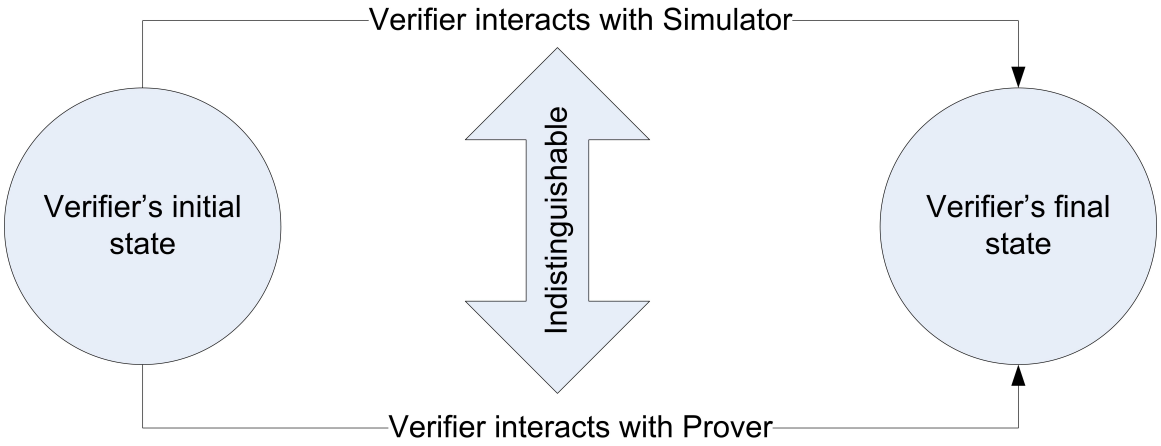


Figure 8.2: Simulator

8.2.6. Definition of ZKP. Based on the above important concepts, ZKP is defined as an interactive proof system with zero-knowledge. The zero-knowledge property means no amount of knowledge should pass between the prover and verifier.

Definition 8.4. *(Zero-knowledge [45]) An interactive strategy A is auxiliary-input zero knowledge on inputs from the set S if, for every probabilistic polynomial-time (interactive) strategy B^* and every polynomial p, there exists a probabilistic polynomial-time (non-interactive) algorithm C^* such that the following two probability ensembles are computationally indistinguishable [38]:*

1. $\{(A,\ B^*(z))\,(x)\}_{x\in S\ ,\ \ z\in\{0,\ 1\}^{p(|x|)}} \stackrel{\text{def}}{=}$ *the output of* B^* *when having auxiliary-input* z *and interacting with* A *on common input* $x \in S$*; and*

2. $\{C^*(x,\ z)\}_{x\in S\ ,\ \ z\in\{0,\ 1\}^{p(|x|)}} \stackrel{\text{def}}{=}$ *the output of* C^* *on input* $x \in S,\ \ z \in \{0,\ 1\}^{p(|x|)}$.

The first one represents an actual execution of an interactive protocol. The second one represents the computation of a stand-alone procedure: the simulator does not interact with anybody. The more basic definition of zero knowledge is obtained by eliminating the auxiliary-input z from this definition. But almost all known ZKPs are in fact auxiliary-input zero-knowledge.

Since whatever can be extracted from interaction with A on input $x \in S$ can also be extracted from x itself, nothing was gained by the interaction itself. Thus, in the concept of ZKPs, the verifier does not obtain further information about the assertion other than its validity. Furthermore, there is no degradation with repeated usage of ZKPs, which means the number of iterations the protocol runs won't change the chances of success of obtaining the secret.

Zero knowledge is a security property which could protect the prover from leaking unnecessary information to any verifier.

8.2.7. Witness Indistinguishability. The notion of witness indistinguishability (WI) was introduced by Feige and Shamir in [27], which means the view of any verifier is "computationally independent" of the witness that is used by the honest prover as auxiliary private input.

NP, that refers to "nondeterministic polynomial time", is the set of all decision problems for which the answer "yes" can be verified in polynomial time. Loosely speaking, for any NP-relation, an argument system for the corresponding language is called WI if no feasible verifier can distinguish the case in which the prover uses one NP-witness to some input from the case in which the prover is using a different NP-witness to the same input [38].

A witness is defined as a string w such that $(x, w) \in R$ [5]. For a binary relation R, $L(R)$ is defined as

$$\{x|\exists\ w\ s.t\ (x, w) \in R\}$$

Specifically, when proving that a string x is in $L(R)$, the prover gets a witness w such that $(x, w) \in R$. If the proof system is WI, then for any w, w' such that $(x, w) \in R$,$(x, w') \in R$, and for any polynomial-size verifier, the view of the verifier when interacting with the prover that uses w as auxiliary input is computationally indistinguishable from its view when interacting with the prover that uses w' as auxiliary input. The WI property is preserved under arbitrary composition of protocols [27].

8.2.8. Honest Verifier VS General Cheating Verifier. The definition of zero-knowledge refers to all feasible verifiers as either honest or cheating. Typically the verifier is viewed as an adversary that is trying to cheat and attempt to gain something by interacting with the prover. With respect to the honest verifier, it will interact with the prover as expected, excepting that it may also maintain a record of the entire interaction and output this record in the end [38].

8.2.9. Black-box Simulator VS Non-black-box Simulator. The definition of zero-knowledge requires that the interaction of the prover with any verifier is computational indistinguishable with an ordinary probabilistic polynomial-time machine that interacts with no one.

Since whatever can be computed after interacting with a prover can be computed on any random input by itself, the ZKP is computationally equivalent to a trusted oracle. This is the notion of a black-box simulator. A black-box simulator is one that can simulate the interaction of the prover with any such a verifier when given oracle access to the strategy of that verifier. To be more specific, a black-box simulator indicates that using any verifier as a black box, produces fake views that are indistinguishable from the verifier's actual interaction with the prover [5]. Most of previous zero-knowledge proofs or arguments are established using a black-box simulator.

But how can the simulator generate an interaction that is indistinguishable from the actual interaction with the prover? First, the simulator has access to the verifier's random question transcript, which means it can actually determine the next question that the verifier is going to ask. Second, the simulator has many ways to answer the verifier's questions. The technique is called rewinding which means if the simulator fails to answer a question provided by the verifier, it simply rewinds the verifier back and asks again [5]. However, when using the verifier's strategy as a black-box, it is hard for the simulator to take advantage of the knowledge of the verifier's random

question transcript. Therefore, there are restrictions and negative results while applying black-box simulator. In [40], Goldreich and Krawczyk has shown non-existence of black-box 3-round ZKPs where a round defines number of steps in the protocol.

In order to overcome the restrictions of black-box simulators, non-black-box simulators were developed. The power of non-black-box simulators has been discovered by Barak in [5]. Barak has shown how to construct non-black-box simulators and obtained several results which are considered to be unachievable through a black-box simulator. He presented a zero-knowledge argument system for NP with following properties:

- It is the first zero-knowledge argument with a constant number of rounds and negligible soundness error.
- It is an Arthur-Merlin (public coins) protocol. The Arthur-Merlin protocol, introduced by Babai [4], is an interactive proof system in which the verifier's coin tosses are known to the prover too.
- It has a strict polynomial time simulator rather than expected polynomial time.

By taking advantage of the strategies of the verifier, since the knowledge of the verifier is not an oracle in non-black-box simulator, Barak's result calls for the re-evaluation of many common beliefs, such as non-existence of (black-box) 3-round ZKPs and impossibility of (black-box) constant-round concurrent zero-knowledge.

Barak's construction uses the technique which was presented by Feige, Lapidot and Shamir in [29]. Two phases in the FLS technique will be combined together to obtain a ZKP. The first phase is called the generation phase, in which the prover and verifier will generate a string. In the second phase, the prover proves to the verifier that either the theorem is true or that the generated string satisfies some property. The FLS technique is based on the notion of WI. Barak's framework is described in [5]:

- Common Input: x and a language L, such that $x \in L$, n is security parameter that makes sure this protocol remains zero-knowledge when executed up to n times concurrently.
- Auxiliary input to prover: w - a witness to the fact that $x \in L$.

1. $Phase1 : GenerationPhase$. The prover and verifier engage in some protocol whose output is a string τ.

2. $Phase2 : WIProofPhase$. The prover proves to the verifier using a WI argument that it knows either a string w such that $(x, w) \in R$ or a string σ such that $(\tau, \sigma) \in S$. R and S are two statements based on language L that the prover wants to convince to the verifier. R is a binary relation that (x, w) is tested. $L(R) \stackrel{def}{=} \{x | \exists w \ s.t. \ (x, \ w) \in R\}$. It means that if $x \in L(R)$ and w is a string such that $(x, w) \in R$. then w is a witness for the fact that $x \in L(R)$. S is a binary relation that (τ, σ) is tested. $L(S) \stackrel{def}{=} \{\tau | \exists \sigma \ s.t. \ (\tau, \ \sigma) \in S\}$. Similarly, it means that σ is a witness for the fact that $\tau in L(S)$. More formally, the prover proves to the verifier using a WI proof knowledge that it knows a witness to the fact that $(x, \tau) \in L(S')$ where S' is defined so that $(x, \tau) \in L(S')$ if and only if there exists w' such that either $(x, w') \in R$ or $(\tau, w') \in S$.

8.2.10. Quality of ZKPs. The soundness condition is considered as a "security" property because it protects the verifier from adversarial behavior by the prover [84]. Usually, it has two commonly used versions:

- Statistical Soundness: If $x \notin L$, then for all, even computationally unbounded, strategies P^*, V accepts in $(P^*, V)(x)$ with probability at most 1/3. This gives rise to interactive proof systems.

- Computational Soundness: If $x \notin L$, then for all polynomial-time strategies P^*, V accepts in $(P^*, V)(x)$ with probability at most 1/3. This gives rise to interactive argument systems.

Zero knowledge is another "security" property, which protects the prover from leaking unnecessary information to the verifier. It comes in three versions of ZKPs based on three interpretations of "similarity".

1. The perfect zero-knowledge notion was suggested by Brassard and Crepeau [72]. The prover is restricted to polynomial time, while the verifier may have unlimited computing power. The computation simulating the actual interaction is identical to the original one.

2. Statistical zero-knowledge requires that the computation simulating the actual interaction is statistically (e.g. in variation distance) close to the original one (with negligible error - smaller than any polynomial fraction in the length of common input) [38]. What's more, statistical zero-knowledge requires that the zero-knowledge condition hold even for computationally unbounded verifier strategy V^*.

3. Computational (or rather general) zero-knowledge requires that the transcript of the interaction is indistinguishable only under the computational complexity assumption, which means that no polynomial-time algorithm can distinguish the two distributions except with negligible probability. It only requires the zero-knowledge condition hold for polynomial-time verifier strategy V^*.

The following definitions are from [52]. Let $(A,\ B)$ be an interactive protocol. Let T, the transcript, be a random variable denoting the verifier view during the protocol on input x. That is, T is the sequence of messages between the prover and verifier under the sequences of coin tosses for A and B. The string h denotes any private input that the verifier may have with the only restriction that its length is bounded by a polynomial in the length of the common input. $M(x,\ h)$ is distributed over the coin tosses of M on inputs x and h.

1. $(A,\ B)$ is perfect zero-knowledge for L if $\exists$ a probabilistic polynomial time Turing machine M s.t. $\forall x \in L$, for all $a > 0$, for all strings h such that $|h| < |x|^a$, the random variable $M(x,\ h)$ and T are identically distributed.

2. $(A,\ B)$ is statistically zero-knowledge for L if $\exists$ a probabilistic polynomial time Turing machine M s.t. $\forall x \in L$, for all $a > 0$, for all strings h such that $|h| < |x|^a$,for all constant $c > 0$ and sufficiently large $|x|$,

$$\sum_{\alpha} |prob\left(M\left(x,\ h\right) = \alpha\right) - prob(T = \alpha)| < \frac{1}{|x|^c}$$

3. $(A,\ B)$ is computationally zero-knowledge for L if $\exists$ probabilistic polynomial time Turing machine M s.t. $\forall$ polynomial size circuit families $C = C_{|x|}$, $\forall$

constant $a,\ d > 0$, for all sufficiently large $|x|$ s.t. $x \in L$, and for all strings h such that $|h| < |x|^a$,

$$prob\left(C_{|x|}(\alpha) = 1 \mid \alpha\ random\ in\ M(x,\ h)\right) - prob\left(C_{|x|}(\alpha) = 1 \mid \alpha\ random\ in\ T\right) < \frac{1}{|x|^d}$$

Based on the two security conditions - soundness and zero knowledge that ZKPs has, we can obtain four kinds of ZKPs. They can be denoted as SZKP, CZKP, SZKA and CZKA, in which the prefix of S indicates statistical and C indicates computational, while the suffix of P indicates interactive proofs (statistical soundness) and A indicates interactive arguments (computational soundness).

SZKPs are defined in the definition of zero-knowledge that the distribution ensembles are required to be statistically indistinguishable rather than computationally indistinguishable. As far as security is concern, SZKPs are of course the most attractive because the following several reasons [38]:

- SZKPs offer information-theoretic security to both parties. CZKPs only offer computational security to the prover, and SZKA only offer computional security to the verifier. While SZKPs hold their security properties regardless of the computational power of the verifier.

- SZKPs provide a clean model for the study of various questions regarding zero-knowledge.

- SZKPs have some properties from a complexity theoretic point of view. It contains a number of specific problems believed to be hard such as Graph Nonisomorphism, Graph Isomorphism, Quadratic Residuosity, and Quadratic Nonresiduosity which are Karp-reducible and is closed under complementation.

Recalling the definition of zero-knowledge, it is a CZKP rather than a SZKP because there do not exist commitment schemes that are simultaneously statistically hiding and statistically binding, which will be discussed in the next section. Under standard complexity assumptions, SZKA can also be constructed for every NP. The intractability assumption used for constructing SZKA for NP seems stronger than the assumption used for constructing CZKP for NP. Assuming both constructions exist, which is more preferable is more or less depending on the application: is it

more important to protect the prover's secrets or to protect the verifier from being convinced of false assertions?

8.3. NP PROBLEM AND ZKPS

Complexity theory has an intriguing property that if any NP-complete problem has a polynomial-time solution, then every problem in NP has a polynomial-time solution.

Definition 8.5. *(The Class NP) NP is the class of decision problems X that admit a proof system $F \subseteq X \times Q$, s.t. there exists a polynomial $p(n)$ and a polynomial-time algorithm A such that [79]:*

1. *$\forall\ x\ \in X\ \exists q \in Q$ s.t. $(x,\ q) \in F$ and moreover, the size of q is at most $p(n)$, where n is the size of x.*

2. *For all pairs $(x,\ q)$, algorithm A can verify whether or not $(x,\ q) \in F$.*

Definition 8.6. *(Polynomial Reduction) Let A and B be two problems. We say that A is polynomially Turing reducible to B if there exists an algorithm for solving A in a time that would be polynomial if we could solve arbitrary instances of problem B at unit cost [79].*

Definition 8.7. *(NP-complete Problem) A decision problem X is $NP-complete$ if $X \in NP$ and for every problem $Y \in NP$, Y is polynomially Turing reducible to X [79].*

It is not yet known if there are efficient solutions (polynomial time) to these problem in NP-complete. Classic examples of these problems include satisfying a boolean formula, the traveling salesman problem, the knapsack problem and so on. But the validity of a proposed solution is easily tested. For example, consider the Hamiltonian Cycle Problem: it is hard to find an efficient algorithm to solve this problem, but it is very easy to verify if a sequence of nodes is a Hamiltonian cycle.

Assuming the existence of secure encryption functions, it has been shown that graph three-colorability, which is known as a NP-complete problem, has a ZKP [42]. Since any problem in NP can be reduced to graph three-colorability according to Karp [56], it ensures that any language in NP has associated with a ZKP system.

In the physical world, the secure encryption functions are represented by opaque, lockable boxes which are usually assumed to be available for the prover, and only the prover has the key. In order to implement the physical boxes algorithmically, a cryptographic primitive named "commitment scheme" [39] will be used . In ZKPs, the way of locking information is to "committing a secret bit" [87], which can be implemented by using any one-way function (assumed to exist).

This commitment scheme describes a two-stage protocol between the prover and the verifier [39]. In the first stage, the prover commits to a value b, which could be digital analogies of sealed envelopes (or, letter, locked boxes). This means sending the value to the verifier through binding some unique value without revealing the original value to the verifier (as when getting a locked box). This property is called hiding. The "hiding" property says that the verifier does not know anything about b during the commit stage. Later on, the prover "decommits" or "reveals" this value to the verifier. This means sending some auxiliary information that allows the verifier to read the uniquely committed value (as when sending the key to the lock) in order to assure that it is the original value. This property is called binding. The binding property means that after the commit stage, there is at most one value that the verifier can successfully open [83]. Hiding and binding can be statistical (holding against computationally unbounded cheating strategies, except with negligible probability) or computational (holding against polynomial-time cheating strategies, except with negligible probability). It's impossible to achieve statistical security for both hiding and binding. But any one-way function could be used in the commitment scheme for statistical hiding or statistical binding.

The widely accepted theorem on how to construct CZKPs system for any NP-set was demonstrated by Goldreich, Micali and Wigderson [42]. And the first construction of SZKA was given by Brassard, Chaum and Crepeau independently [7]. Feige, Lapidot and Shamir designed a ZKP on identification schemes in [29]. Constant-round ZKPs were first showed by Feige and Shamir in [28].

8.4. ZKPS APPLICATIONS

In this section, several problems are introduced with their ZKPs construction. To provide a ZKP for a general NP statement, one can translate it into an instance of the graph three-colorability problem using a Karp reduction [56]. Another way to provide

ZKPs avoiding a Karp reduction is using the simulation method after exploiting the properties of the specific NP-complete language.

8.4.1. ZKP for Graph Three-Colorability.

Definition 8.8. *(Graph Three-Colorability) A graph $G(V,\ E)$ is said to be three-colorable if there exists a mapping $\phi\ : V\ \rightarrow \{1,\ 2,\ 3\}$ (called a proper coloring) such that every two adjacent vertexes are assigned different colors (i.e., each $(u,\ v) \in E$ satisfies $\phi(u) =\ \phi(v)$). Such a three-coloring induces a partition of the vertex set of the graph to three independent sets. The language graph three-colorability, denoted G3C, consists of the set of undirected graphs that are three-colorable.*

Figure 8.3 is a Peterson graph [1] with three colored vertices (Red, Green and Blue).

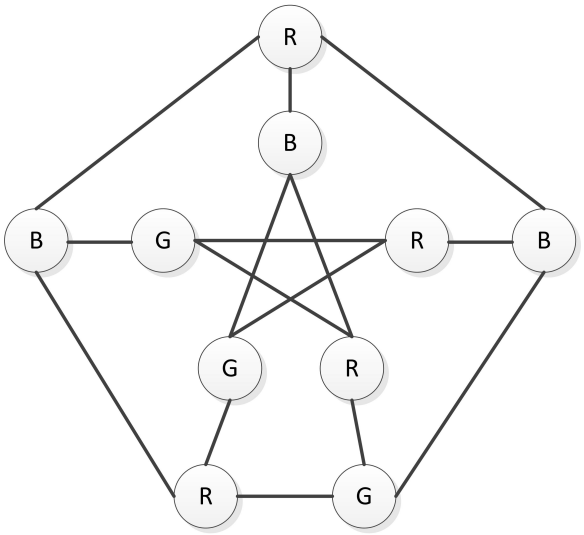


Figure 8.3: 3Colored Graph

Suppose graph $G(V,\ E)$ is colored by $\phi\ (\phi\ : V\ \rightarrow \{1,\ 2,\ 3\})$. Let $n = |V|\ ,\ m = |E|$ and $S_3 = Sym\{1,\ 2,\ 3\}$. Since the graph is (simple and) connected, n and m are polynomially related (i.e., $n - 1 \leq m\ < n^2/2$).

Common input: A graph $G(V,\ E)$

The following four steps are executed m^2 times, each time using independent coin tosses [42].

1. P: The prover chooses at random an assignment of three colors to the three independent sets induced by ϕ, colors the graph using these three colors, and places these colors in n locked boxes each bearing the number of the corresponding vertex. More specifically, the prover chooses a permutation $\pi \in_R S_3$, places $\pi(\phi(i))$ in a box marked i $(\forall\ i\ \in V)$, locks all boxes and sends them (without the keys) to the verifier.

2. V: The verifier chooses at random an edge $e \in_R E$ and sends it to the prover. (Intuitively, the verifier asks to examine the colors of the endpoints of $e\ \in E$)

3. P: If $e = (u,\ v) \in E$, then the prover sends the keys of u and v. Otherwise, the prover does nothing.

4. V: The verifier opens boxes u and v using the keys received and checks whether they contain two different elements of $\{1,\ 2,\ 3\}$. If the keys do not match the boxes, or the contents violates the condition then the verifier rejects and stops. Otherwise, the verifier continues to the next iteration.

If the verifier has completed all m^2 iterations then it accepts.
Figure 8.4 illustrates the ZKP for G3C.

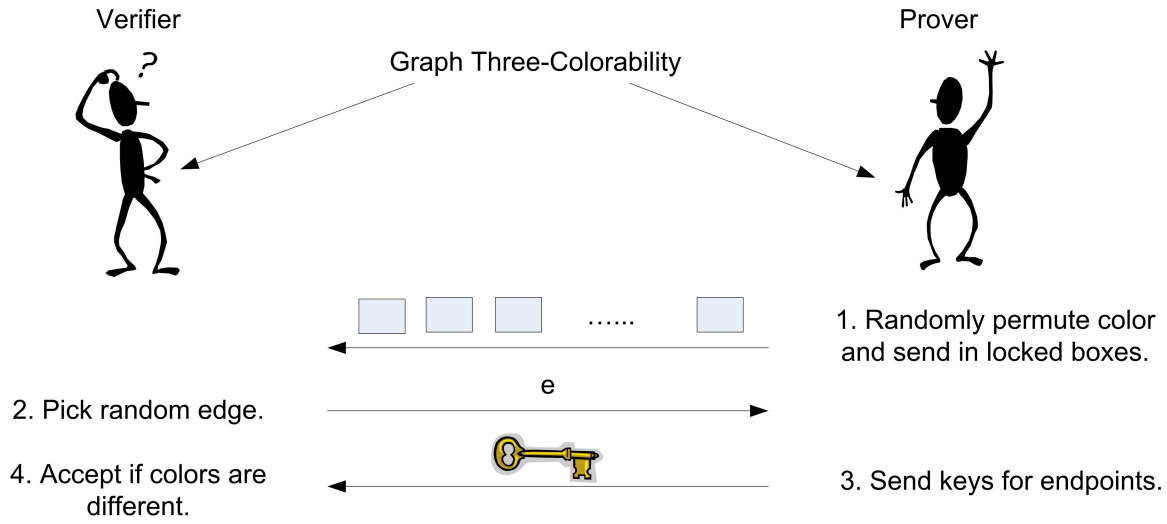


Figure 8.4: ZKP for G3C

This protocol constitutes a ZKP for G3C since it satisfies three properties.

- Completeness: If the graph is three-colorable, then any pair of boxes u and v corresponding to some edge of the graph will certainly be colored differently. Therefore, an honest verifier will complete all m^2 iterations and accept with probability 1.

- Soundness: If the graph is not three-colorable and the verifier follows the protocol, then no matter how the prover plays, at each round the verifier will reject with probability at least $\frac{1}{m}$. The probability that the verifier will accept is bounded by $(1 - \frac{1}{m})^{m^2} \approx \exp(-m)$.

- Zero-knowledge: The only information received by the verifier at each round is a pair of different randomly selected elements of $\{1, 2, 3\}$. It is crucial that the prover uses at each round an independently selected random permutation of the colors. Thus, the names of three classes at one round are uncorrelated to the names at another round.

8.4.2. ZKP for Feige-Fiat-Shamir Identification Scheme. The Feige-Fiat-Shamir identification scheme is one of the classic authentication zero-knowledge schemes [26]. The security of the Feige-Fiat-Shamir system is based on the fact that it is difficult to extract square roots modulo large composite integers of unknown factorization [79].

Initialization [68]:

1. A trusted center T selects and publishes an RSA(Ron Rivest, Adi Shamir and Leonard Adleman)-like modulus $n = pq$, but keep the primes p and q secret.

2. The prover selects a secret s coprime to n, $1 \leq s \leq n-1$, computes $v = s^2 \; mod \; n$, and registers v with T as her public key.

Identification Protocol [68]:

The following steps are executed t times, each time using independent random coin tosses.

1. P: The prover choses a random r, $1 \leq r \leq n-1$ and sends $x = r^2 \; mod \; n$ to the verifier.

2. V: The verifier randomly selects a bit $\alpha \in \{0, 1\}$ and sends α to the prover.

3. P: The prover computes and sends to the verifier y, where $y = r$ $(if\ \alpha = 0)$, or $y = rs\ mod\ n$ $(if\ \alpha = 1)$

4. V: The verifier rejects if $y = 0$ or if $y^2 \neq x \cdot v^{\alpha} mod\ n$.

Figure 8.5 illustrates the ZKP for Feige-Fiat-Shamir identification scheme.

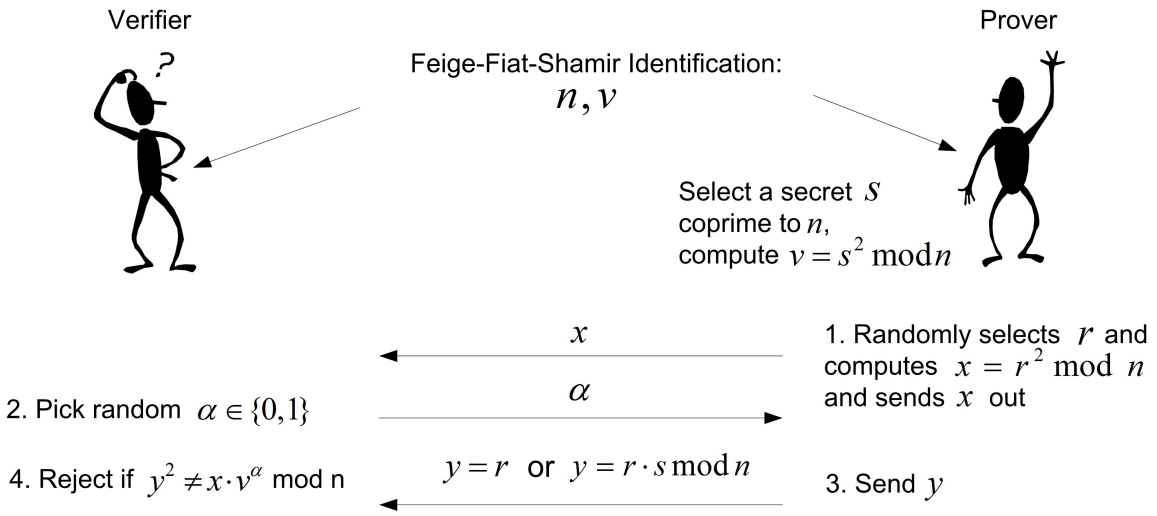


Figure 8.5: ZKP for Feige-Fiat-Shamir Identification Scheme

If the verifier has completed all t iterations, then he accepts.

This protocol is a ZKP because it upholds the properties of completeness, soundness and zero-knowledge.

- Completeness: Suppose the prover possesses the secret s. Then she can always correctly provide the verifier with $y = r$ or $y = rs\ mod\ n$ upon request. Therefore, an honest verifier will complete all t iterations and accept with probability 1.

- Soundness: Suppose the prover does not possess the secret s. Then, during any given round, she can provide only one of $y = r$ or $y = rs\ mod\ n$. Therefore, an honest verifier will reject with probability $\frac{1}{2}$ in each round, so the probability the verifier will be fooled is 2^{-t}.

- Zero-knowledge: The only information revealed in each round is $x = s^2 mod\ n$, and either $y = r$ or $y = rs\ mod\ n$. Such pairs (x, y) could be simulated by choosing y randomly, then define $x = y^2$ or $x = y^2/v$. Such pairs are computationally indistinguishable from the interaction with the prover.

8.4.3. ZKP for Graph Isomorphism. The Graph Isomorphism (GI) problem is basically asking the question: Given two graphs G_0 and G_1, is there a bijection between their sets of nodes that preserves edges?

Definition 8.9. *(Graph Isomorphism) Two graphs $G(V, E)$ and $G'(V, E')$ are isomorphic if and only if there exists a permutation $\pi \in S_{|V|}$ (the symmetric group of $|V|$ elements) such that $(u, v) \in E$ iff $(\pi(u), \pi(v)) \in E'$. We can write $G' = \pi G$.*

There is an isomorphism between two graphs in Figure 8.6. $f(a) = 1, f(b) = 6, f(c) = 3, f(d) = 5, f(e) = 2, f(f) = 4$

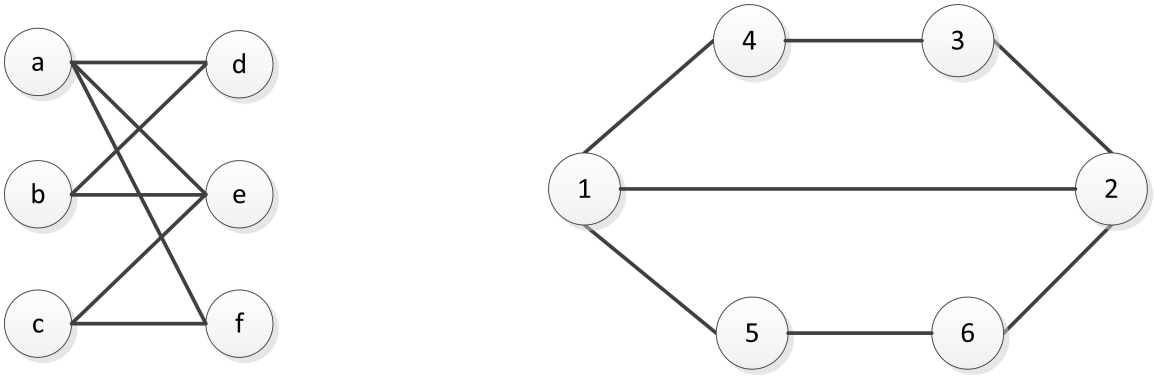


Figure 8.6: Graph Isomorphism

The language graph isomorphism (GI) consists of all the pairs of isomorphic graphs.

Common Input: Two graphs $G_0(V,\ E)$ and $G_1(V,\ E')$. Let ϕ denote the isomorphism between G_0 and G_1, that is $G_1 = \phi G_0$.

The following steps are executed t times, each time using independent random coin tosses [68].

1. P: The prover generates a graph, H, which is a random isomorphic copy of G_1. This is done by selecting a permutation $\pi \in_R S_{|V|}$, and computing $H = \pi G_1$. The prover sends the graph H to the verifier.

2. V: The verifier chooses at random $\alpha \in_R \{0, 1\}$, and sends α to the prover.

3. P: If $\alpha = 1$, then the prover sends $\beta = \pi$ to the verifier, else ($\alpha = 0$) the prover sends $\beta = \pi \cdot \phi$. Here, $\pi \cdot \phi$ indicates the combination of π and ϕ.

4. V: If the permutation β received from the prover is not an isomorphism between G_α and H (i.e., $H \neq \beta G_\alpha$), then the verifier stops and rejects; otherwise, he continues.

Figure 8.7 illustrates the ZKP for graph isomorphism.

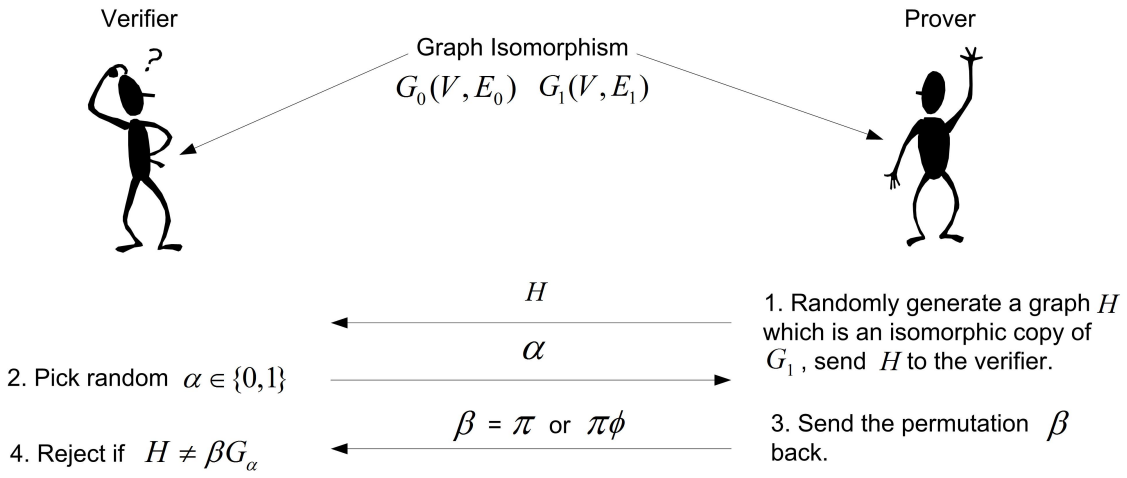


Figure 8.7: ZKP for Graph Isomorphism

If the verifier has completed t iterations of the above steps, then he accepts.

This protocol is a ZKP because it upholds the properties of completeness, soundness and zero-knowledge.

- Completeness: If $(G_0,\ G_1) \in GI$, then the random isomorphic copy H of G_1will always be isomorphic to both G_0 and G_1. Therefore, an honest verifier will complete all iterations and accept with probability 1.

- Soundness: If $(G_0,\ G_1) \notin GI$, then the random isomorphic copy H of G_1 will be isomorphic to only one of G_0 or G_1. Therefore, an honest verifier will reject with probability $\frac{1}{2}$ in each round.

- Zero-knowledge: The only information revealed in each round is either π or $\pi \cdot \phi$ where $\pi \in_R S_{|V|}$. Due to the random selection of π, a simulator which computes a random isomorphic copy of G_0 or G_1 or both is computationally indistinguishable from interaction with the prover.

8.4.4. ZKP for Hamiltonian Cycle [12].

Definition 8.10. *(Hamiltonian Cycle) Let G be a graph. A Hamiltonian Cycle in G is a cycle that passes through all the nodes of G exactly once. The n nodes of G are labeled $N_1,\ N_2,\ \ldots\ldots N_n$.*

Figure 8.8 has a Hamiltonian Cycle labeled by the solid line.

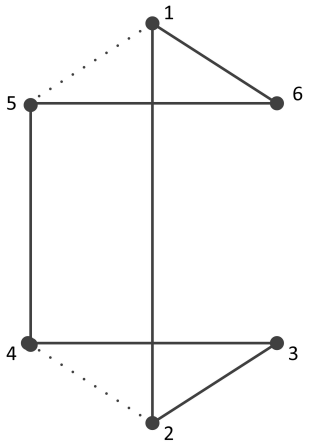


Figure 8.8: Hamiltonian Cycle

The following protocol will be executed for k rounds.

1. P: The prover encrypts G with the boxes in secret by randomly mapping n labeled nodes $N_1, N_2, \ldots\ldots N_n$ 1-1 into n labeled boxes $B_1, B_2, \ldots\ldots B_n$ in such a way that each one of the $n!$ permutations of the nodes into the boxes is equally

probable. For every pair of boxes ($B_i,\ B_j$) prepare a box labeled B_{ij}. This box is to contain a 1 if the node placed in B_i is adjacent to the node in B_j; 0 otherwise. All $(n + C_2^n)$ boxes are then to be locked and presented to the verifier.

2. V: Upon receiving $(n + C_2^n)$ boxes, the verifier has two choices:

   - ($\alpha = 0$) The verifier can request that the prover unlock all the boxes. In this case, the verifier may check that the boxes contain a description of G. (For example, if N_1 is adjacent to both N_2, N_5 but no other nodes in G. If N_1 is in B_i, N_2 in B_j, and N_5 in B_k, then there should be a 1 in both B_{ij}, B_{ik}, and a 0 in B_{ix} for every other value of x)
   - ($\alpha = 1$) The verifier can request that the prover open exactly n boxes $B_{ij}, B_{jk}, B_{kl} \ldots\ldots B_{l'i}$, which can demonstrate that G contains a Hamiltonian Cycle if all these boxes all contain a 1. Since the B_i are not opened, the sequence of node numbers defining the Hamiltonian Cycle in G is not revealed.

3. P: The prover opens the appropriate boxes according to one of the two random requests from the verifier (either the graph or the Hamiltonian Cycle)

4. V: The verifier accepts the proof if the prover complies and rejects otherwise.

Figure 8.9 illustrates the ZKP for Hamiltonian Cycle.

This protocol is a ZKP because it upholds the properties of completeness, soundness and zero-knowledge.

- Completeness: If the prover knows the graph, G, contains a Hamiltonian Cycle, he can successfully show the graph or cycle according to the requests from the verifier. So the verifier will complete all k rounds and accept with probability 1.

- Soundness: If the graph doesn't contain a Hamiltonian Cycle, the prover's probability of convincing the verifier that he does know the graph contains a Hamiltonian Cycle is 1/2 in each round. His probability of convincing the verifier that he does know are $\leq 1/2^k$ when there are k rounds.

- Zero-knowledge: In each round, the verifier either obtains an encrypted graph of G or a random n cycle. When the prover reveals all the boxes that describe

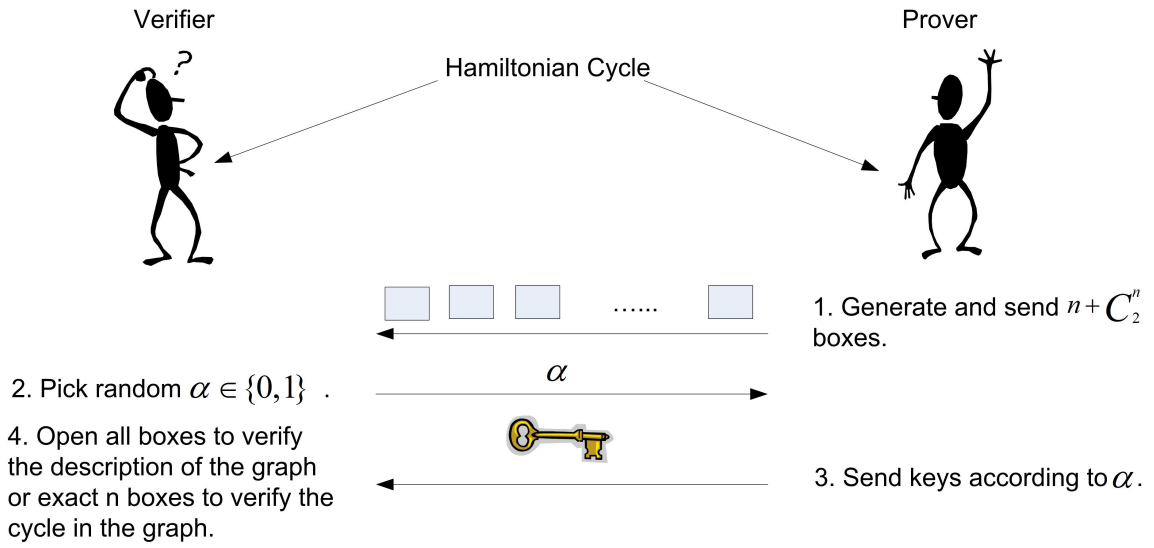


Figure 8.9: ZKP for Hamiltonian Cycle

the G, it is only one of the $n!$ random mappings of the n nodes of G into the n labeled boxes. When the prover reveals the boxes containing a cycle, it is just a random n cycle. Thus the verifier gets no useful information.

8.4.5. Another ZKP for Graph Three-Colorability [12]. The n nodes of G are labeled $N_1,\ N_2,\ \ldots\ldots N_n$. Suppose the graph is colored with Red, White and Blue. If node N_i is colored red, it is called N_i^R.

1. P: The prover prepares $3n$ pairs of boxes $\langle B_1^c,\ \ B_1\rangle$, $\langle B_2^c,\ \ B_2\rangle$, $\ldots\ldots \langle B_{3n}^c,\ \ B_{3n}\rangle$. Without revealing to the verifier, the prover randomly maps $3n$ nodes

   $N_1^R,\ \ldots\ldots N_n^R,\ N_1^W,\ \ldots\ldots N_n^W,\ N_1^B,\ \ldots\ldots N_n^B$ 1-1 into the $3n$ pairs of boxes. Each of the $(3n)!$ permutations mapping the $\{N_i^x\}$ on to the $\langle B_j^c,\ \ B_j\rangle$ is equally probable. The prover puts x (the color) into B_j^c, and puts i (the node number) into B_j. For every pair of number-containing boxes $(B_i,\ \ B_j)$, prepare a box labeled B_{ij}. This box contains 1 if the node of G in B_i has the color B_i^c, the node of G in B_j has the color $B_j^{c'}$, and if the node in B_i is adjacent in G to the node in B_j and contains 0 otherwise. All boxes are then locked and presented to the verifier.

2. V: Upon receiving $(2\cdot 3n + C_2^n)$ boxes, the verifier have two choices:

(a) The verifier can request that the prover unlock all the boxes B_{ij} and all the numbers containing boxes B_i, but none of the colors containing B_i^c. In this way, the prover reveals the graph G without revealing its coloring. The verifier can check the boxes contain a correct description of G.

(b) The verifier can request that the prover open the $3n$ boxes $\{B_i^c\}$ to reveal the colors they contain, and then open just those boxes B_{ij} such that B_i^c contains the same color as B_j^c. The opened boxes B_{ij} will all contain a 0 if and only if any 2 nodes that are colored the same are not adjacent in the graph represented by the boxes. The verifier can check the correct three-coloring.

3. P: The prover opens the appropriate boxes according to one of the two random requests from the verifier (either all boxes without colors or adjacent vertices with different colors)

4. V: The verifier accepts the proof if the prover complies and rejects otherwise.

This protocol is a ZKP because it upholds the properties of completeness, soundness and zero-knowledge.

- Completeness: If the graph is three-colorable, those nodes that are colored the same are not adjacent in graph. So the verifier will complete all rounds and accept with probability 1

- Soundness: If the graph is not three-colorable, the prover's probability of convincing the verifier that he does know the graph is three colorable is $\frac{1}{2}$ in each round. His chance of convincing the verifier that he does know are $\leq 1/2^k$ when there are k rounds.

- Zero-knowledge: In each round, the verifier either obtains an encrypted graph or boxes containing "0". When the prover reveals all the boxes that describe the graph, it is only one of the random mappings of the n nodes of graph into n labeled boxes. When the prover reveals the boxes containing "0", the verifier gets no useful information without the corresponding numbers of nodes.

8.4.6. ZKP Sketch for SAT. The quadratic residuosity problem is the question of distinguishing by calculating the quadratic residues modulo n, where n is

a composite number of two odd primes p and q. Under the assumption of quadratic residuosity that two states are computational indistinguishable through the calculation of the quadratic residues modulo n, a protocol for satisfiability (SAT) is suggested that directly simulates a circuit that evaluates given instances of SAT [49].

The general technique in [72] is through the simulation of an arbitrary boolean circuit without disclosing the inputs or any intermediary results. At the end of the protocol, if the final output of the circuit is 1, then the circuit is satisfiable, but nothing else.

Let $u = b_{1,}\, b_{2,} \ldots\ldots b_k$ be a k bit string of the prover. For each $1 \leq i \leq k$, let z_i and z'_i be the two encryptions of b_i randomly chosen by the prover. It is easy for the prover to convince the verifier that the k bit strings encrypted by $z_{1,}\, z_{2,} \ldots\ldots z_k$ and $z'_{1,}\, z'_{2,} \ldots\ldots z'_k$ are identical without providing the verifier with any additional information by the following string equality protocol.

Definition 8.11. *(String Equality Protocol): For each $i,\ 1 \leq i \leq k$, the prover gives the verifier some $x_i \in \mathbb{Z}^*_{\kappa}$ (denoting the set of integers relatively prime to n between 1 and $n-1$) so that $z_i z'_i \equiv x_i^2 (mod\ n)$.*

Definition 8.12. *(Boolean Computation Protocol): Consider any boolean function $B : \{0,\ 1\}^t \rightarrow \{0, 1\}$ agreed upon between the prover and the verifier, and any bits $b_{1,}\, b_{2,} \ldots\ldots b_t$ only known to the prover. For $1 \leq i \leq t$, let z_i be an encryption of b_i known to the verifier. Let $b = B\left(b_{1,}\, b_{2,} \ldots\ldots b_t\right)$. The prover produces an encryption z for b and convinces the verifier that z encrypts the correct bit without giving the verifier any information on the input bits $b_{1,}\, b_{2,} \ldots\ldots b_t$ nor on the result b.*

A permuted truth table for the boolean function B is introduced here, which is a binary string of length $(t+1)2^t$ formed of 2^t blocks of $t+1$ bits. The last bit of each block is the value of B on the other t bits of the block. Let s be the number of permutations agreed upon between the prover and the verifier.

1. P: The prover randomly chooses s permuted truth tables for B and discloses encryptions for each of them.

2. V: The verifier selects a random subset $X \subseteq \{1,\ 2,\ \ldots\ldots,\ s\}$ and sends it to prover as a challenge

3. P: The prover chooses one of the following options based on the request from the verifier.

   - For each $j \in X$, the prover opens the entire encryption of the j^{th} permuted truth table.
   - For each $j \notin X$, the prover points to the appropriate block in the encryption of the j^{th} permuted truth table and uses the following string equality protocol to convince the verifier that $z_1, z_2, \ldots\ldots z_t z$ encrypts the same bit string as this block.

4. V: The verifier makes the following verifications.

   - The verifier checks if it is a valid truth table for B.
   - The verifier checks if $z_1, z_2, \ldots\ldots z_t z$ encrypts the same bit string.

Figure 8.10 illustrates the ZKP for boolean computation.

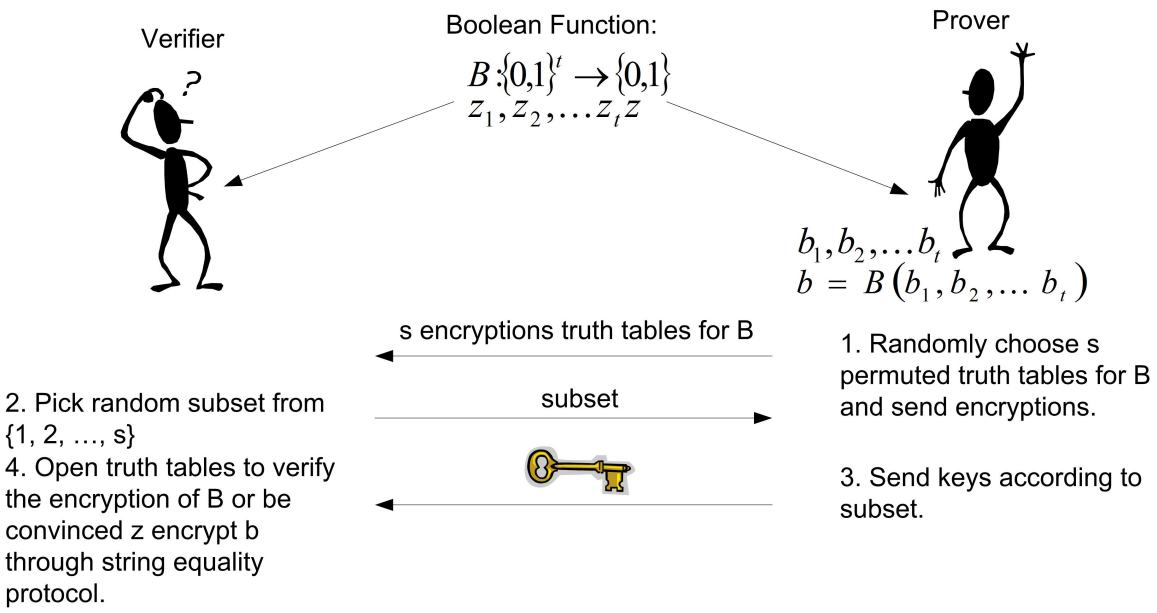


Figure 8.10: ZKP for Boolean Computation

Based on the above discussions, a ZKP sketch has been designed for SAT. $f : \{0,\ 1\}^k \rightarrow \{0,\ 1\}$ is the function computed by some satisfiable boolean formula for

which the prover knows that there is an assignment $b_{1,}\ b_{2,} \ldots\ldots b_k \in \{0,\ 1\}$ so that $f\left(b_{1,}\ b_{2,} \ldots\ldots b_k\right) = 1$. Assume the boolean formula is given using arbitrary unary and binary boolean operators. The prover will produce encryptions $z_{1,}\ z_{2,} \ldots\ldots z_k$ of $b_{1,}\ b_{2,} \ldots\ldots b_k$. Then the prover will guide the verifier through the encrypted evaluation of the formula, using the boolean computation protocol, one boolean operator at a time. The result will be a z which is the encryption for the value of $f\left(b_{1,}\ b_{2,} \ldots\ldots b_k\right)$. Then the prover opens z and shows the verifier that it encrypts a 1.

8.4.7. ZKP for Circuit Computations. In [49], it is possible to directly simulate the computation of any given computational device. Simulation is used to separate the data encryption from the encryption of the device's structural (or control) information. It directly proves the result of the computation, avoiding the Karp reduction from a specific NP-complete problem.

In the protocol, the essential idea is that the prover constructs and sends to the verifier a copy of a simulation of the computing device. This copy should include encoding of the possible input, intermediate results, and output data. In addition, it includes encoding of structural information about the computing device. Upon receiving the encoding information, the verifier chooses a bit $\alpha \in \{0,\ 1\}$ and sends it back.

- With probability $\frac{1}{2}$ the verifier decides to verify, that is to request that the prover open all the encryptions in the copy. In this way, the verifier can check if the construction is a legal one with the correct structural information.

- With probability $\frac{1}{2}$ the verifier chooses to compute, in which case the prover opens only the result of the computation. In order to prove the output presented is in fact the computed result, the prover opens only parts that are involved in the computation, while other information is left encrypted. The unopened information appears random.

The process is repeated r times; each time the verifier either chooses to verify or chooses to compute. If all verifications are successful, and all the computations produce a connection to the same opened output, then the verifier accepts the result of the computation.

The prover uses an encryption procedure E and simulates the computation of a circuit C in a minimum-knowledge fashion with the verifier. Let E(C) denote the encryption.

1. P: The prover probabilistically encrypts C using an encryption procedure E. Then the prover sends E(C) to the verifier.

2. V: The verifier chooses a bit $\alpha \in \{0,\ 1\}$ and sends it back.

3. P: The prover chooses one of the corresponding actions based on the requests from the verifier.

   - $\alpha = 0$ {verify}: the prover opens all the encryptions of all gates in E(C) and sends the cleartext circuit to the verifier
   - $\alpha = 1$ {compute}: the prover opens only the pointers of the specific computation an sends the cleartexts of these pointers to the verifier including the outputs

4. V: The verifier does the corresponding verifications.

   - $\alpha = 0$, the verifier verifies that C is properly encrypted;
   - $\alpha = 1$, the verifier verifies that the opened pointers lead from the (unopened) input entries to the output pointers.

Figure 8.11 illustrates the ZKP for circuit computation.

If all computations give the same value and all verifications are successful then the verifier accepts. Otherwise, the verifier rejects.

8.4.8. ZKP for Exact Cover. The Exact Cover is defined in the following.

Definition 8.13. *(Exact Cover): A set* $\{u_i,\ i = 1,\ 2,\ \ldots\ldots t\}$. *A family* $\{S_j\}$ *is a subset of the set* $\{u_i,\ i = 1,\ 2,\ \ldots\ldots t\}$. *Exact cover means there is a subfamily* $\{T_h\} \subseteq \{S_j\}$ *, such that sets* T_h *are disjoint and* $\bigcup T_h = \bigcup S_j = \{u_i,\ \ i = 1,\ 2,\ \ldots\ldots t\}$.

The following protocol will be executed for t rounds. The verifier uses a coin toss in each round.

Let $|T_h| = m$, $|S_j| = n$

1. P: The prover prepares the following four kinds of boxes:

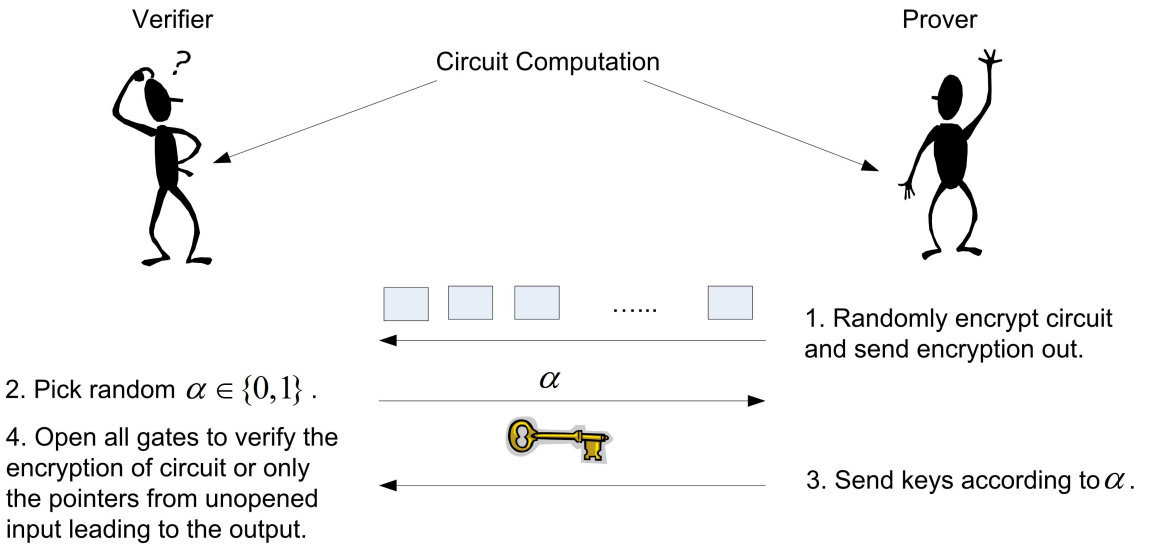


Figure 8.11: ZKP for Circuit Computation

- t labeled boxes $B_1,\ B_2,\ \ldots\ldots B_t$ which will contain set $\{u_i,\ \ i = 1,\ 2,\ \ldots\ldots t\}$ in a random order
- n labeled boxes $A_1,\ A_2,\ \ldots\ldots A_n$ which will contain element in S_j
- $n \cdot t$ black boxes in a matrix as follows, if the set member in B_i is in the S_j of A_j, then insert a 1; otherwise insert a 0.

	B_1	B_2	B_3	...	...	B_t
A_1	1	0	0	...	...	0
A_2	0	0	1	...	...	0
...	0	1	0	...	...	0
...	0	0	0	...	1	0
A_n	1	0	0	...	...	1

- For every two rows in matrix, a pair of boxes is prepared as $< Index * Index, Result >$. The first box contains indexes of two rows. The second box contains the product operation of any two row vectors (one is row vector, another is the transpose of row vector). If the result of product operation is 0, $Result = 0$; otherwise, $Result = 1$.

2. V: Upon receiving $(2 \cdot t + n \cdot t + 2 \cdot C_2^n)$ black boxes, the verifier has two choices:

   - $(\alpha = 0)$ The verifier can ask to open all the boxes to verify if the description of the problem is correct.
   - $(\alpha = 1)$ The verifier asks the prover to open some pairs of boxes with *Result* $= 0$. With the index showing in these boxes, the verifier also asks to open the corresponding row in the matrix to check if $\bigcup T_h = \bigcup S_j = \{u_i,\ i = 1,\ 2,\ \ldots\ldots t\}$.)

3. P: The prover opens the appropriate boxes according to the two random requests from the verifier (either all the boxes describing the exact cover problem or disjoint solution sets).

4. V: The verifier accepts the proof if the prover complies and reject otherwise.

Figure 8.12 illustrates the ZKP for exact cover.

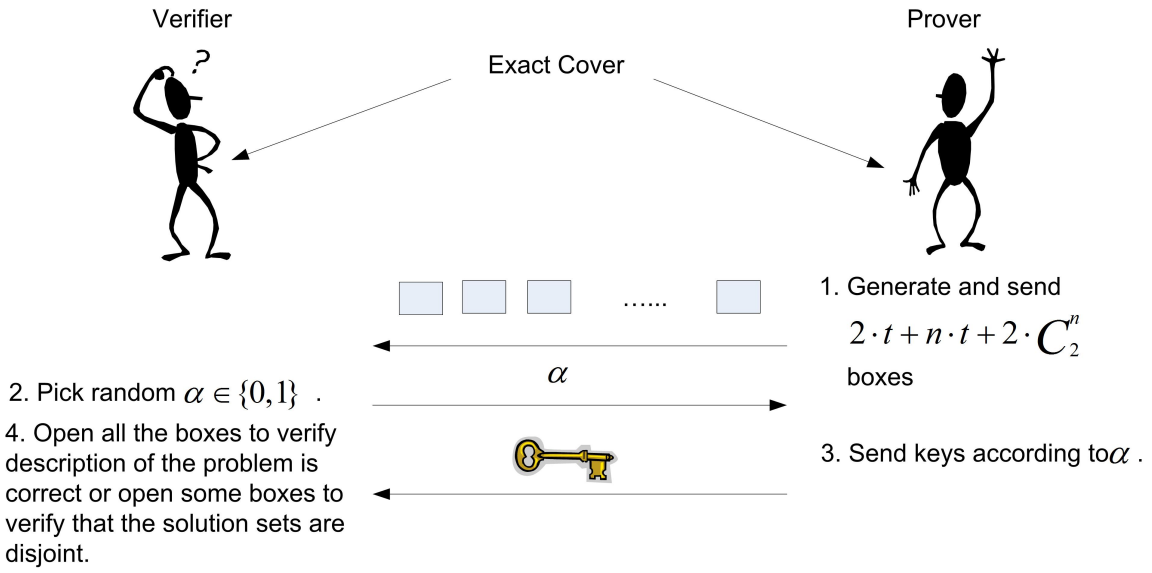


Figure 8.12: ZKP for Exact Cover

This protocol is a ZKP because it upholds the properties of completeness, soundness and zero-knowledge.

- Completeness: If the prover knows the exact cover, he can encrypt the correct information in the matrix. So the verifier will complete all rounds and accept with probability 1.

- Soundness: If the prover doesn't know the exact cover, the prover's chance of convincing the verifier that he does know the exact cover is $\frac{1}{2}$ in each round: either the exact cover problem is correctly encrypted or there are disjoint solution sets. His chance of convincing the verifier that he does know the exact cover are $\leq 1/2^k$ when there are k rounds.

- Zero-knowledge: In each round, the verifier either obtains a random matrix with "0" and "1" or obtains one element in set $\{S_j\}$ which is known to the verifier beforehand. Thus, there is no useful information transferred.

An example will be used to illustrate ZKP for exact cover , suppose: $S = \{A, B, C, D, E, F\}$. $\{u_i\} = \{1, 2, 3, 4, 5, 6, 7\}$. $A = \{1, 4, 7\}, B = \{1, 4\}, C = \{4, 5, 6\}, D = \{3, 5, 6\}, E = \{2, 3, 5, 6\}, F = \{2, 7\}$.

In ZKP for exact cover, the prover will prepare seven black boxes named $B_1, B_2, ...B_7$, which will contain elements in $\{u_i\}$ in a random order. Additional six black boxes named $A_1, A_2, ...A_6$ will be used to contain sets in $\{S_i\}$. With A_i, B_i, the following matrix will be prepared.

		2	1	6	5	3	7	4
		B_1	B_2	B_3	B_4	B_5	B_6	B_7
S_6	A_1	1	0	0	0	0	1	0
S_1	A_2	0	1	0	0	0	1	1
S_4	A_3	0	0	1	1	1	0	0
S_2	A_4	0	1	0	0	0	0	1
S_3	A_5	0	0	1	1	0	0	1
S_5	A_6	1	0	1	1	1	0	0

The prover also prepares following pairs Table 8.1 of boxes which contain indexes and result of * operation of any two rows:

The prover will send all of those black boxes to the verifier. Upon receiving $(2 \cdot 7 + 6 \cdot 7 + 2 \cdot 15) = 86$ black boxes, the verifier has two choices:

Table 8.1: Black-Boxes for Index and Product Operation

1*2, 1	1*3, 0	1*4, 0	1*5, 1	1*6, 1
2*3, 0	2*4, 1	2*5, 1	2*6, 0	3*4, 0
3*5, 0	3*6, 1	4*5, 1	4*6, 0	5*6, 1

- The verifier asks the prover to open all black boxes to verify the boxes contain the public exact cover problem.
- The verifier asks the prover to open some pair of boxes with $Result = 0$. The prover will open (13, 0) and (34, 0). With index 1, 3 and 4, the verifier ask the prover to open row 1, 3 and 4 in the matrix and check if elements in row 1, 3 and 4 are combined together covering all elements in the set $\{u_i\}$

8.4.9. ZKP for 0-1 Knapsack. The 0-1 Knapsack is defined in the following.

Definition 8.14. *(0-1 Knapsack): There are n kinds of items. Each kind of item i has a value v_i and a weight w_i. The maximum weight that a bag can carry is W. The 0-1 knapsack algorithm determines the subsets of items which give maximum value without exceeding the maximum weight of the bag. Mathematically, the 0-1 knapsack problem can be formulated as:*

- *Maximize* $\sum_i^n v_i \cdot x_i$ *with* $x_i \in \{0, 1\}$
- *Subject to* $\sum_i^n w_i \cdot x_i \leq W$ *with* $x_i \in \{0, 1\}$

The simple case of the problem with the property that for each kind of item the weight equals the value $w_i = v_i$ is as follows.

The 0-1 simple knapsack problem is to find a binary n-vector x such that a given W equals to $w \cdot x$. A supposed solution x is easily checked in at most n additions, but finding a solution is believed belong to NPC. As n is larger than 100 or 200, the number of operations grows exponentially and computationally infeasible for exhaustive trial and error search over all 2^n possible x.

However, the degree of difficulty is crucially dependent on the choice of w. If $w = (1, 2, 4, ..., 2^n)$, then solving for x is equivalent to finding the binary representation of W. Generally, if for all i, $w_i > \sum_{j=1}^{i-1} w_j$, then x is also easily found. $x_n = 1$ if and only

if $W \geq w_n$ and for $i = n-1, n-2, ..., 1$, $x_i = 1$ if and only if $W - \sum_{j=i+1}^{n} x_j \cdot w_j \geq w_i$. A trapdoor knapsack is defined as one in which carefully choice of w as above. They form a proper subset of all possible knapsacks and their solutions are not as difficult as the hardest knapsacks in NP theory [66]. In this paper, we are not considering trapdoor knapsacks.

According to the Karp's reduction theorem, the ZKP for knapsack problem can be generated in a similar way as the ZKP for exact cover problem.

The following protocol will be executed for t rounds. The verifier uses a coin toss in each round.

1. P: The prover prepares the following four kinds of boxes:

   - n labeled boxes $B_1, B_2, \ldots\ldots B_n$ which will contain items $w_1, w_2, \ldots\ldots, w_n$ in a random order
   - m boxes $S_1, S_2, \ldots\ldots S_m$ will contain some numbers which are summation of random items from w_i
   - $m \cdot n$ black boxes in a matrix as follows, if the item B_i is included in sum S_i, then insert a 1; otherwise insert a 0.

	B_1	B_2	B_3	...	...	B_n
S_1	1	0	0	...	...	0
S_2	0	0	1	...	...	0
...	0	1	0	...	...	0
...	0	0	0	...	1	0
S_m	1	0	0	...	...	1

   - For every two rows in matrix, a pair of boxes is prepared as $< Index * Index, Result >$. The first box contains indexes of two rows. The second box contains the product operation of any two rows vectors (one is row vector, another is the transpose of row vector). If the result of product operation is 0, $Result = 0$; otherwise, $Result = 1$.

2. V: Upon receiving $(2 \cdot n + m \cdot n + 2 \cdot C_2^m)$ black boxes, the verifier has two choices:

   - $(\alpha = 0)$ The verifier asks the prover to open all the boxes to verify if they are satisfied the original simple knapsack problem.

- ($\alpha = 1$) The verifier asks the prover to open a subset of the summation boxes S_i with $\sum S_i = W$. The verifier also asks to open all the pairs of boxes to check if $Result = 0$ within the subset of S_i.

3. P: The prover opens the appropriate boxes according to the two random requests from the verifier (either all the boxes describing the knapsack problem or those boxes where W is equal to some summation S_i with $Result = 0$ for each pair).

4. V: The verifier accepts the proof if the prover complies and rejects otherwise.

Figure 8.13 illustrates the ZKP for 0-1 simple knapsack. If the verifier has completed t iterations of the above steps, then he accepts.

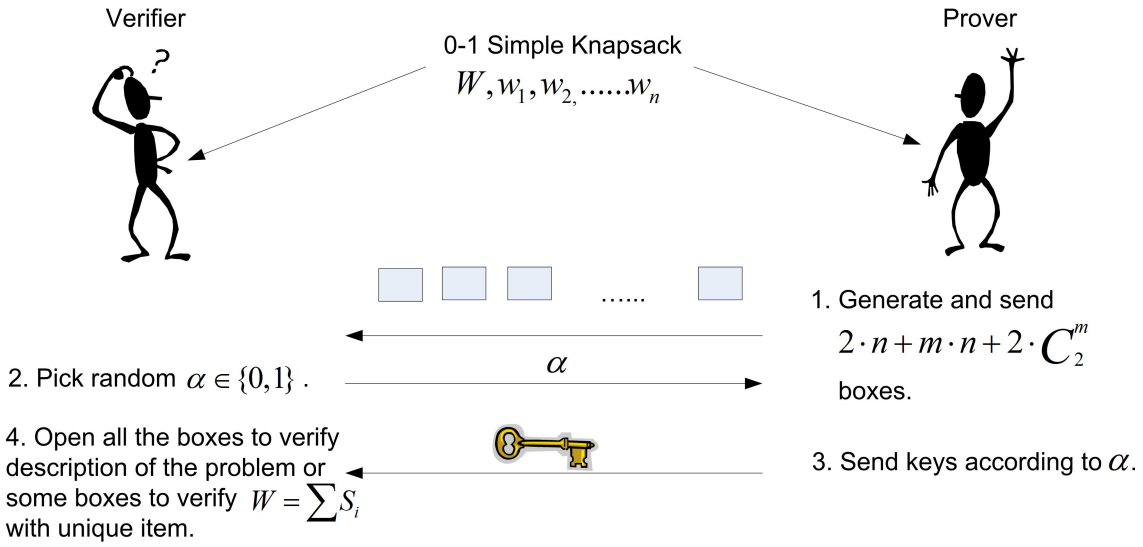


Figure 8.13: ZKP for 0-1 Simple Knapsack

This protocol is a ZKP because it upholds the properties of completeness, soundness and zero-knowledge.

- Completeness: If the prover knows the solution to the simple knapsack, he can encrypt the correct information in all the boxes. So the verifier will complete all rounds and accept with probability 1.

- Soundness: If the prover doesn't know the solution to the simple knapsack, the prover's chance of convincing the verifier that he does know the solution is $\frac{1}{2}$ in each round: either the simple knapsack problem or some summation equal to constraint W. His chance of convincing the verifier that he does know the exact cover are $\leq 1/2^k$ when there are k rounds.

- Zero-knowledge: In each round, the verifier either obtains a random matrix with "0" and "1" or obtains some random numbers. Thus, there is no useful information transferred.

8.5. ADVANCED TOPICS IN COMPOSING ZKPS

8.5.1. Composing ZKPs. ZKPs have many applications in cryptographic areas such as identity verification, authentication, key exchange and so on. Provided that one-way functions exist, ZKPs can be used to force parties to behave according to a predetermined protocol, such as multi-party secure computations [41]. The next natural question is whether the zero-knowledge condition is preserved under a variety of composition operations: sequential, parallel and concurrent. This question not only belongs to the theoretical area, but is crucial to the application of ZKPs in the cryptographic area. For example, we want to make sure the information obtained by the verifier during the execution of a ZKP will not enable him to extract any additional knowledge from subsequent executions of the same protocol.

For $T \in \{sequential, parallel, concurrent\}$, a protocol is T-zero-knowledge if it is zero-knowledge under a composition of type T [38]. The definitions of T-zero-knowledge are derived by considering that the verifiers can initiate a polynomial number of interactions with the prover using scheduling type T.

Protocol repetition, which repeats the basic protocol many times independently and links them together, is mainly used for error reduction. Protocol composition is different than protocol repetition; the prover is not assumed to coordinate its actions between different executions of the protocol [75]. If the verifier has some misbehavior in protocol repetition, the prover can refuse to continue execution. By contrast, in protocol composition, the prover is still obilgated to continue interaction with other executions of the protocol. The obvious way to perform repetition would be to execute the basic protocol sequentially, but it suffers from a high round-complexity, resulting in

a high number of message exchanges in the execution. Parallel repetition that conducts the basic protocol in parallel is used as a way of decreasing the error probability while maintaining the number of rounds of message.

1. In sequential composition, the ZKP is invoked (polynomially) many times, where each invocation follows the termination of the previous one [38]. Assuming a basic ZKP contains four-round $P1, V1, P2, V2$ message passing, Figure 8.14 shows the sequential composition. It is the most basic case of protocol composition. Every protocol that is zero-knowledge is sequential-zero-knowledge, which means zero-knowledge is closed under sequential composition.

2. In parallel composition, (polynomially) many instances of the ZKP are invoked at the same time and proceed at the same pace [38]. Under the assumption of a synchronous model of communication, the executions are totally synchronized so that the ith messages in all instances are sent exactly at the same time. As shown in Figure 8.15, assuming a basic ZKP contains four-round $P1(i = 1), V1(i = 2), P2(i = 3), V2(i = 4)$ message passing, the ith message in all ZKPs should be sent out at the same time.

   Goldreich and Krawczyk in [40] presented a simple protocol that is zero-knowledge, but is not closed under parallel composition. The example is described in Table 8.2:

   The prover's strategy is zero-knowledge because:

   - If the verifier chooses 0, then it is infeasible for the verifier to guess a passing pair (β, γ) since the α is randomly chosen by the prover. The verifier doesn't obtain anything from the interaction.
   - If the verifier chooses 1, then it obtains a pair $(\beta, f(\alpha\beta))$. Since β is selected by the prover, for any α, the value $f(\alpha\beta))$ is random to the verifier and $(\beta, f(\alpha\beta))$ is indistinguishable from a uniformly selected pair of n-bit long strings.

   However, if the verifier conducts two concurrent executions with the prover, there will be information leakage, illustrated in Table 8.3. In this case, the verifier chooses 0 in one session and 1 in another session. Upon receiving the prover's

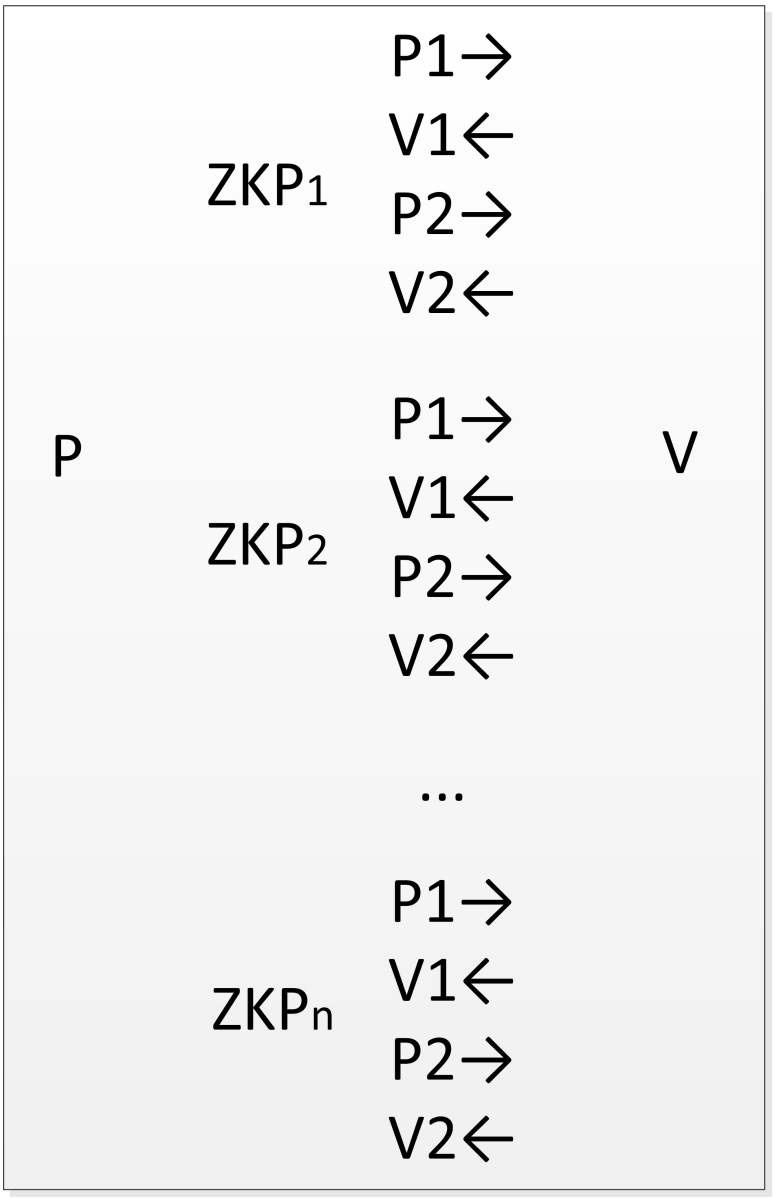


Figure 8.14: Sequential Composition of ZKP

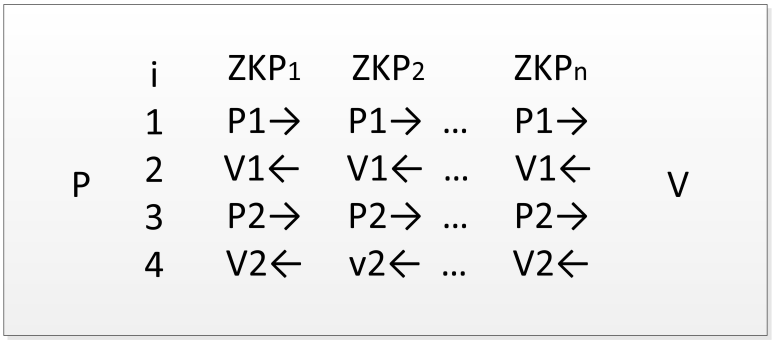


Figure 8.15: Parallel Composition of ZKP

Table 8.2: Protocol Not Closed Under Parallel Composition

Consider the prover holding a secret and a random function $f : \{0,1\}^{2n} \rightarrow \{0,1\}^n$, and willing to participate in the protocol. The verifier is supposed to send the prover a binary value $\{0,1\}$:
For choice 0, the prover uniformly selects $\alpha \in \{0,1\}^n$, and sends it to the verifier which is supposed to reply with a pair of n-bit long strings, denoted (β, γ). The prover checks whether or not $f(\alpha\beta) = \gamma$. The input of f function, $\alpha\beta$, is the concatenation operation of two n-bit string. If the equality is satisfied, the prover sends the verifier the secret he has. For choice 1, the verifier is supposed to uniformly select $\alpha \in \{0,1\}^n$, and send it to the prover, which selects uniformly $\beta \in \{0,1\}^n$, and replies with the pair $(\beta, f(\alpha\beta))$.

message α in the first session, the verifier sends α as its own message in the second session and obtains a pair $(\beta, f(\alpha\beta))$ from the prover's execution of the second session. Then the verifier sends the pair $(\beta, f(\alpha\beta))$ to the first session of the prover, which will satisfy the equation $f(\alpha\beta) = \gamma$ and the verifier will obtain the desired secret.

Table 8.3: Parallel Composition Leads to Information Leakage

V = 0	V = 1
$P \stackrel{\alpha}{\rightarrow} V$	
	$P \stackrel{\alpha}{\leftarrow} V$
	$P \stackrel{(\beta, f(\alpha\beta))}{\rightarrow} V$
$P \stackrel{(\beta,\gamma)}{\leftarrow} V$	
$P \stackrel{secret}{\rightarrow} V$ $(f(\alpha\beta) == \gamma)$	

Based on the above example, zero-knowledge is not closed under parallel composition in general [75]. However, assuming one-way functions exist, it has been

proven that there exists zero-knowledge protocols for NP that are closed under parallel composition and having a constant number of rounds [37]. Conceptually, the independently execution of one basic protocol will give no information to other execution of basic protocol since the prover will respond independently based on message from each execution.

Using a simulator is the technique to demonstrate the zero-knowledge property. The idea behind it is that whatever the verifier might have learned from interacting with the prover, the verifier could learn by himself by running the simulator. The simulator has two advantages over the prover to compensate for not knowing the secret as the prover does [5]. The first advantage is that the simulator knows and can determine the questions the verifier will ask. The second advantage is that the simulator has many choices to answer the verifier's questions. After it fails to answer a question from the verifier, the simulator simply choose not to output this interaction and goes back to some point to output only the successful interaction. This process is called rewind which helps the simulator to rewind back from the failure and try again. The black-box simulator, which simulates the interacting of the prover with the verifier without access to the strategy of that verifier, is used to demonstrate the zero-knowledge [37].

3. Concurrent composition is a more general protocol composition, which was first considered by Dwork, Naor and Sahai [25]. Here (polynomially) many instances of the protocol are invoked at an arbitrary time and proceed at an arbitrary pace [38]. This allows one or multiple verifiers to engage in many proofs with the prover and arbitrarily interleave the messages by running some of the protocols ahead in order to gain information that will enable it to attack some of the other protocols. Within each proof, the verifier must follow the proper order of the steps. Among different proofs, the verifier can interleave arbitrarily. For example, the verifier may execute the first step of proof 1, then execute all steps of proof 2 in order to obtain some knowledge in advance to execute the remaining steps in proof 1. Concurrent composition is shown in Figure 8.16.

   There are two models of concurrent composition: a purely asynchronous model and an asynchronous model with timing. The purely asynchronous model is a simpler model and requires no assumptions about the underlying communication

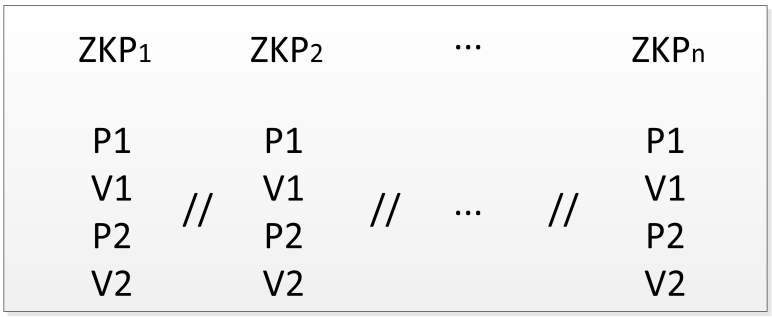


Figure 8.16: Concurrent Composition of ZKP

channels. While an asynchronous model with timing, each party is assumed to hold a local clock such that the relative clock rates are bounded by an *a priori* known constant.

The timing model assumes that each party holds a local clock such that the relative clock rates are bounded by an *a priori* known constant [25]. There are two constants involved in the timing assumption: ρ and Δ. ρ is a global time bound on the relative rates of the local clock and is known to all parties. Δ is an upper bound on the message handling and delivery time. Under this timing model, a constant-round concurrent ZKP can be constructed as shown in [25].

Some problems will arise if time-driven operations are considered. The use of time-driven operations, such as timeout of incoming messages, receipt and delay of outgoing messages, will slow down the execution of the protocol. However, in the absence of more appealing alternatives, the use of this timing model still is considered reasonable [38].

8.5.2. The Richardson-Kilian Concurrent ZKPs Protocol. Under standard intractability assumptions, concurrent ZKPs exist for NP in a purely asynchronous model by Richardson and Kilian (RK) [74]. In this model, a verifier is allowed to run up to k interactive proofs with the prover. The RK protocol consists of two stages: an $O(k)$ preamble messages and a main body. The first stage is independent of the common input, in which the verifier and the prover will be engaged in $O(k)$ message exchanges. First, the verifier commits to k random n-bit strings $v_1, v_2, ..., v_k \in \{0,1\}^n$, where n is the security parameter of the protocol that represents the number of concurrent executions and k is the parameter that determines the

number of rounds. In the following k iterations between the prover and verifier, the prover commits to a random n-bit string, p_j, and the verifier decommits to the corresponding v_j. In the second stage, the prover provides proof either for the statement is true or for some $j \in \{1, 2, ..., k\}$ $v_j = p_j$. Table 8.4 illustrates two stages.

Table 8.4: RK Protocol

First Stage: $V \rightarrow P$: Commit to $v_1, v_2, \cdots, v_k$ For $j = 1, 2, \cdots, k$ $P \rightarrow V$: Commit to p_j $V \rightarrow P$: Decommit to v_j *Second Stage:* $P \leftrightarrow V$: Zero Knowledge Proof for statement is true or $\exists j$ *s.t.* $v_j = p_j$

The technique used here is from WI in section 2.7: instead of proving a statement T, the prover proves a weaker theorem $T \vee W$, where W is a statement that will fail with extremely high probability [74]. In the actual interactions, there is little chance for the prover to guess v_j from the verifier. Thus, the prover has no choice but to provide the proof for the statement T.

It is necessary to examine whether the zero-knowledge property is still preserved in the protocol. Recall that in order to demonstrate the zero-knowledge property, a simulator has to be generated that can simulate the views of every polynomial-time adversary interacting with the prover. Under concurrent composition, the simulator's task becomes more complicated [75]. In the RK protocol, the preamble messages will be used by the simulator to its advantage. Whenever the simulator may cause $v_j = p_j$ to happen for some j, it can simulate the rest of the protocol and provide a WI proof. The larger the number of rounds in the preamble, the more resistant the result is to concurrent attacks, and the easier the simulation task is. However, the number of rounds in the protocol will also increase.

8.5.3. The Improved Concurrent ZKPs Protocol. Prabhakharan, Rosen and Sahai (PRS) proposed a more sophisticated concurrent ZKP in [73] with

$O(\alpha(n) \cdot logn)$ rounds of iteration, where $\alpha(\cdot)$ is any super-constant function such as $loglogn$. The PRS protocol also consists of two stages, which is illustrated in Table 8.5. In the first stage, the verifier will commit to a random n-bit string σ, and to two sequences $\{\sigma^0_{i,j}\}^k_{i,j=1}$, and $\{\sigma^1_{i,j}\}^k_{i,j=1}$. Each of the sequences consists of k^2 random n-bit strings, such that for every i, j the value of $\sigma^0_{i,j} \oplus \sigma^1_{i,j} = \sigma$. The total number of committed strings in this step is $2 \cdot k^2 + 1$. This is followed by k iterations between the prover and the verifier. In the jth iteration, the prover will send a random k-bit string, $r_j = r_{1,j}, r_{2,j}, ..., r_{k,j}$, and the verifier will decommit k strings of $\sigma^{r_{1,j}}_{1,j}, \sigma^{r_{2,j}}_{2,j}, ..., \sigma^{r_{k,j}}_{k,j}$. After first stage, the verifier has opened a total of k^2 strings in the two sequences.

In the second stage, the prover and the verifier will engage in a three-round protocol. After the prover commits secret values about the statement to the verifier, the challenge question σ will be decommitted with all the σ, $\{\sigma^{1-r_{i,j}}_{i,j}\}^k_{i,j=1}$ that were not revealed in the first stage. Upon checking that the strings satisfy the constraint $(\sigma^0_{i,j} \oplus \sigma^1_{i,j} = \sigma)$, the prover decommits the corresponding value to the verifier.

Table 8.5: PRS Protocol

First Stage:
$V \rightarrow P$: Commit to σ, $\{\sigma^0_{i,j}\}^k_{i,j=1}$, $\{\sigma^1_{i,j}\}^k_{i,j=1}$, for each i, j, $\sigma^0_{i,j} \oplus \sigma^1_{i,j} = \sigma$.
For $j = 1, 2, \cdots, k$:
$P \rightarrow V$: Send k-bit string $r_j = r_{1,j}, r_{2,j}, ..., r_{k,j}$
$V \rightarrow P$: Decommit k strings of $\sigma^{r_{1,j}}_{1,j}, \sigma^{r_{2,j}}_{2,j}, ..., \sigma^{r_{k,j}}_{k,j}$

Second Stage:
$P \Rightarrow V$: Perform n independent copies of commitments of secrets
$V \Rightarrow P$: Decommit to σ and to $\{\sigma^{1-r_{i,j}}_{i,j}\}^k_{i,j=1}$
$P \Rightarrow V$: Answer according to the value of σ if $\sigma^0_{i,j} \oplus \sigma^1_{i,j} = \sigma$ is satisfied

Both the RK and the PRS protocol follows the same structure of adding a preamble phase to the basic protocol. This phase is only used for sucessful simulation to prove the zero-knowledge property. Applying this protocol, Blum's Hamiltonian Cycle concurrent zero-knowledge proof has been demonstrated. Thus, every language

in NP has a concurrent zero-knowledge proof system. A concurrent ZKP for the 0-1 simple Knapsack of Table 8.6 that is shown in section 4.9, yields the PRS protocol in Table 8.7.

Table 8.6: Basic ZKP for 0-1 Simple Knapsack

Common input: n items, each item i with weight w_i and a weight constraint W. P1: The prover prepares all the black boxes. V1: Upon receiving $(2 \cdot n + m \cdot n + 2 \cdot C_2^m)$ black boxes, the verifier has two choices: $(\alpha = 0)$ The verifier asks the prover to open all the boxes to verify if they satisfy the original simple knapsack problem. $(\alpha = 1)$ The verifier asks the prover to open a subset of the summation boxes S_i with $\sum S_i = W$. The verifier also asks to open all the pair of boxes to check if $Result = 0$ within the subset of S_i. P2: The prover opens the appropriate boxes according to the two random requests from the verifier (either all the boxes describing the knapsack problem or whose boxes where W is equal to some summation S_i with unique item). V2: The verifier accepts the proof if the prover complies and rejects otherwise.

A detailed description is given as follows. In the first stage, the verifier will commit an n-bit random string σ and $2k^2$ n'-bit random strings as shown in Table 8.8 with the restriction that $\sigma_{i,j}^0 \oplus \sigma_{i,j}^1 = \sigma$ as shown in Table 8.9:

For the next $2k$ iterations, the verifier will decommit k strings in each iteration to the prover as shown in Table 8.10:

In the second stage, the prover will make independent copies of $P1$ in Table 8.6 with n' parallel verifiers. The verifier will decommit the σ and remaining k^2 $\sigma_{i,j}^{1-r_{i,j}}$ to the prover. The prover will execute n' copies of $P2$ in Table 8.6 if the k^2 strings in the first stage and k^2 strings in the second stage satisfy $\sigma_{i,j}^0 \oplus \sigma_{i,j}^1 = \sigma$. The verifier will accept only if all the conditions are satisfied.

In the actual execution of the protocol, since the prover does not know the value of σ, the protocol would be a proof system for 0-1 simple knapsack with soundness error $\frac{1}{2^{n'}}$. The sole purpose of the first stage is to allow the simulator to know the value of σ. As long as the simulator makes the verifier reveal both $\sigma_{i,j}^0$ and $\sigma_{i,j}^1$ for

Table 8.7: PRS Concurrent ZKP for 0-1 Simple Knapsack

Common input: n items, each item i with weight w_i and a weight constraint W. A parameter k is used for determining the number of rounds and also works as the security parameter that represents the number of concurrent executions.
First stage: $2k+2$ rounds interactions between the prover and the verifier (independent of the knapsack problem)

Verifier's preliminary step: the verifier selects and commits to an n'-bit string σ and two sequences of n'-bit strings: $\{\sigma^0_{i,j}\}^k_{i,j=1}$, $\{\sigma^1_{i,j}\}^k_{i,j=1}$, for each i, j, $\sigma^0_{i,j} \oplus \sigma^1_{i,j} = \sigma$. The total committed strings are $2k^2+1$.
For $j = 1, 2, \cdots, k$:
Prover's jth step: Send k-bit string $r_j = r_{1,j}, r_{2,j}, ..., r_{k,j}$ to the verifier
Verifier's jth step: Decommit k strings of $\sigma^{r_{1,j}}_{1,j}, \sigma^{r_{2,j}}_{2,j}, ..., \sigma^{r_{k,j}}_{k,j}$

Second stage: The prover and verifier engage in n' parallel executions of Basic ZKP for the 0-1 simple knapsack
$\hat{P}1$: The prover will prepare boxes for n' parallel independent executions, which means there will be $n' \cdot (2 \cdot n + m \cdot n + 2 \cdot C^m_2)$ black boxes. Each set of $(2 \cdot n + m \cdot n + 2 \cdot C^m_2)$ black boxes is generated independently and contains the different commitment of the same knapsack problem.
$\hat{V}1$: The verifiers decommit to σ and remaining $\{\sigma^{1-r_{i,j}}_{i,j}\}^k_{i,j=1}$, which are not revealed in the first stage.
$\hat{P}2$: The prover checks that the verifier has properly decommitted to the value by checking $\sigma^0_{i,j} \oplus \sigma^1_{i,j} = \sigma$. Each bit in σ is the question from the verifier for each execution. If so, the prover opens the appropriate boxes according to each bit value of σ for each execution(either all the boxes within one set which describes the knapsack problem correctly or under one set W is equal some summation of S_i with unique item) .
$\hat{V}2$: The verifier conducts the verification of the prover's proofs based on the result of $\hat{P}2$.

Table 8.8: $2k^2$ Randomly Generated Strings

$\sigma^0_{1,1}$	$\sigma^0_{1,2}$	...	$\sigma^0_{1,k}$	$\sigma^1_{1,1}$	$\sigma^1_{1,2}$	...	$\sigma^1_{1,k}$
$\sigma^0_{2,1}$	$\sigma^0_{2,2}$	...	$\sigma^0_{2,k}$	$\sigma^1_{2,1}$	$\sigma^1_{2,2}$	...	$\sigma^1_{2,k}$
...	...	...	...	...	...	...	...
$\sigma^0_{k,1}$	$\sigma^0_{k,2}$	...	$\sigma^0_{k,k}$	$\sigma^1_{k,1}$	$\sigma^1_{k,2}$	...	$\sigma^1_{k,k}$

Table 8.9: Constraints on Strings

$\sigma^0_{1,1} \oplus \sigma^1_{1,1} = \sigma$	$\sigma^0_{1,2} \oplus \sigma^1_{1,2} = \sigma$	...	$\sigma^0_{1,k} \oplus \sigma^1_{1,k} = \sigma$
$\sigma^0_{2,1} \oplus \sigma^1_{2,1} = \sigma$	$\sigma^0_{2,1} \oplus \sigma^1_{2,2} = \sigma$	...	$\sigma^0_{2,k} \oplus \sigma^1_{2,k} = \sigma$
...	...	...	...
$\sigma^0_{k,1} \oplus \sigma^1_{k,1} = \sigma$	$\sigma^0_{k,2} \oplus \sigma^1_{k,2} = \sigma$	...	$\sigma^0_{k,k} \oplus \sigma^1_{k,k} = \sigma$

Table 8.10: Preamble Messages

$(P1)$: The prover will send a random k-bit string $r_1 = r_{1,1}, r_{2,1}..., r_{k,1}$
$(V1)$: The verifier will decommit $\sigma_{1,1}^{r_{1,1}}, \sigma_{2,1}^{r_{2,1}}, ..., \sigma_{k,1}^{r_{k,1}}$
$(P2)$: The prover will send a random k-bit string $r_2 = r_{1,2}, r_{2,2}..., r_{k,2}$
$(V2)$: The verifier will decommit $\sigma_{1,2}^{r_{1,2}}, \sigma_{2,2}^{r_{2,2}}, ..., \sigma_{k,2}^{r_{k,2}}$
(...)
(Pk): The prover will send a random k-bit string $r_k = r_{1,k}, r_{2,k}..., r_{k,k}$
(Vk): The verifier will decommit $\sigma_{1,k}^{r_{1,k}}, \sigma_{2,k}^{r_{2,k}}, ..., \sigma_{k,k}^{r_{k,k}}$

some i, j, it can simulate the rest of the protocol by adjusting the $P1$ in Table 8.6 according to the value of $\sigma = \sigma_{i,j}^0 \oplus \sigma_{i,j}^1$.

8.5.4. Efficiency Considerations. Round-complexity is considered as a very important complexity measure for cryptographic protocols. It indicates the number of message exchanges taking place in the protocol. Typically, a constant number of rounds will be a desirable result. Under the assumptions of the existence of one-way functions, ZKPs can be constructed in constant number of rounds by Feigh and Shamir [28], Brassard, Crepeau and Yung [16], and Goldreich and Kahan [39] [38].

In [39], Goldreich and Kahan presented how to reduce the error probability of interactive proofs that having constant error probability without increasing the round-complexity and while preserving their zero-knowledge property. They are SZKP whose soundness condition requires that nobody, even with unbounded computational ability, can fool the verifier into accept the false statements except with negligible probability.

Recall that commitment schemes are commonly used to construct ZKPs with two phases. The first phase is called commit. The verifier will not know which value it is if the prover commits a value. The second phase is called reveal. The prover will send some auxiliary information that allows the verifier to reveal the uniquely committed value in order to assure that it is the original value. Two properties named secrecy and nonambiguity are involved [39]. Secrecy requires that at the end of the commit phase, the verifier does not gain any information of the prover's value. Nonambinguity means that in the reveal phase, with the transcript of the interaction in the commit phase, there exists only one value that the verifier may accept as a legal reveal of the commitment.

Two different commitment schemes have been presented to construct constant-round ZKPs [39].

- Standard commitment scheme: the nonambiguity requirement is absolute (makes no reference to the computational power of the adversary) whereas the secrecy requirement is computational (refers only to probabilistic polynomial-time adversaries).
- Perfect commitment schemes: the secrecy requirement is absolute, while the nonambiguity requirement is computational.

Goldreich and Kahan presented how to construct constant-round zero knowledge proofs for every set in NP, assuming the strong intractability assumptions (the existence of claw-free collections). The claw-free collection implies the existence of one-way functions, but the converse might not be true. Generally, four steps are involved in constructing a round-efficient ZKP [39]:

1. The verifier commits to a challenge. This is usually implemented by two rounds/messages.
2. The prover commits to a sequence of values.
3. The verifier decommits either properly or not.
4. Depending on the verifier's proper decommitment, the prover decommits to the corresponding values.

Based on this construction of constant-round ZKP for NP, Goldreich proved that security will be preserved in the two extreme schedulings of concurrent executions [37]: one is the case of parallel execution, and the other is of concurrent execution under the timing model. The blackbox simulator technique is used for zero-knowledge property verification.

8.5.5. Knowledge Complexity. Knowledge complexity is introduced by Goldwasser, Micali and Rackoff [45] to measure the computational advantage obtained by interaction. Anything obtained during the interaction will be considered as knowledge. How to quantify the amount of knowledge obtained by interaction is more challenging. One definitional approach is trying to bound the amount of knowledge by the number of bits that are communicated in an alternative interaction that

allows to simulate the original interaction. To be more concrete, one party is said to yield at most $k(|x|)$ bits of knowledge if the variation distance between interactive proof system denoted as $(P,V)(x)$ and a simulator denoted as is bounded above by $1-2^{-k(|x|)}+|x|^{-c}$. x is the input string. $k(|x|)=\frac{log(|x|)}{O(1)}$ [43].

8.5.6. Non-interactive Zero-Knowledge. The non-interactive zero-knowledge proof system was defined by Blum, Feldman and Micali. It consists of three entities: a prover, a verifier and a uniformly selected reference string. The reference string is not selected by either parties, but is selected by a trusted third party [39]. The reference string is public to the prover as well as the verifier. The interaction between the two parties is only a single message sent from the prover to the verifier. Then it is left for the verifier to reach the final decision.

Non-interactive zero-knowledge proof systems have many applications such as the construction of public-key encryption and signature schemes. What's more, ZKP can be derived by combining a secure coin-flipping protocol with a non-interactive zero-knowledge proof system [39]. Thus, the round-complexity of this ZKP depends on the round-complexity of the coin-flipping protocol and on whether it can be securely performed in parallel many times.

8.6. CONCLUSION

This chapter presented a survey on ZKPs with backgrounds, important concepts, existing applications for NP problems, composition operations and efficiency considerations. It constructed ZKPs for two NP problems: exact cover and 0-1 simple knapsack based on Blum's protocol for Hamiltonicity. Applying the PRS protocol, a concurrent ZKP for 0-1 simple knapsack is provided and explained in detail.

The notion of zero-knowledge originated with the efforts to formalize problems during the design of cryptographic protocols. ZKPs have been shown in many applications of the cryptographic area such as identity verification, authentication, key exchange and enforcing the honest behavior while maintaining privacy. In an authentication system, one party wants to prove its identity to a second party via some secret information but doesn't want the second party to learn anything about this secret. One of the classic authentications based on ZKPs is the Feige-Fiat-Shamir proof of identity [26]. The discovery of ZKPs for all of NP has played an important role in secure computation where several parties engage in a protocol to jointly compute a

function on their private inputs in a way that no party learns anything other than the output of the protocol. Since ZKPs can be viewed as a special case of secure two-party computation, every party can use ZKPs to convince each other that it is following the specific protocol without revealing its private input. This application was first developed by Goldreich, Micali and Wigderson in [42]. A general construction of a ZKP for an NP relation, which makes a black-box use of a secure protocol for a related multi-party functionality, was presented in [50].

ZKP can also be a good solution to some interesting problems. For example, in [51], the author proposed a new ZKP of identity scheme based on visual cryptography, visual zero knowledge proof of identity, that has been built with only Boolean OR operations. In [46], the author introduced cryptographic and physical ZKPs systems for solutions of Sudoku Puzzles, which are also known as NP problem.

Existing ZKPs are iterative in nature; their protocols require multiple communication rounds, making it not an appropriate security protocol for CPSs that information exchanges frequently but not iteratively between the cyber parts and the physical parts. Most of ZKPs are based on some complexity assumptions (e.g. the existence of one-way function such as quadratic residuosity, factoring, discrete log, etc.). Researchers still try to find a model in which ZKPs for all of NP could be obtained without any assumptions. In secure computation, it is realized that the complexity assumptions could be removed by working in a model with private communication channels.

The further work of ZKPs could go between cryptography and complexity theory. Non-black box ZKP has begun to inspire complexity theorists to reexamine whether known limitations of black-box reduction can be bypassed with various types of non-black box reductions. Another direction is to find common variants of ZKPs [84], such as noninteractive zero knowledge, proofs and arguments of knowledge and witness indistinguishable protocols.

BIBLIOGRAPHY

[1] Petersen Graph http://en.wikipedia.org/wiki/Petersen_graph accessed on Oct 12nd, 2011.

[2] Zero Knowledge Proof http://en.wikipedia.org/wiki/Zero_knowledge_proof accessed on Oct 12nd, 2011.

[3] R. Akella, F. Meng, D. Ditch, B. McMillin, and M. Crow. Distributed power balancing for the freedm system. *Smart Grid Communications (SmartGridComm), 2010 First IEEE International Conference on*, pages 7–12, 2010.

[4] L. Babai. Trading Group Theory for Randomness. *17th ACM Symposium on the Theory of Computing*, pages 421–429, 1985.

[5] B. Barak. How to go Beyond the Black-Box Simulation Barrier. *42nd IEEE Symposium on Foundations of Computer Science*, pages 106–115, 2001.

[6] H. Barringer. Levin & Gries: Verification of CSP. *A Survey of Verification Techniques for Parallel Programs*, pages 70–83, 1985.

[7] G. Barssard, D. Chaum, and C. Crepeau. Minumum Disclosure Proofs of Knowledge. *Computer and System Sciences*, 37(2):156–189, 1988.

[8] H.-J. Bckenhauer, D. Komm, R. Krlovi, and R. Krlovi. On the Advice Complexity of the Knapsack Problem. *LATIN'12 Proceedings of the 10th Latin American international conference on Theoretical Informatics*, pages 61–72, 2012.

[9] H.-J. Bckenhauer, D. Komm, R. Krlovi, R. Krlovi, and T. Mmke. On the Advice Complexity of Online Problems (Extended Abstract). *ISAAC '09 Proceedings of the 20th International Symposium on Algorithms and Computation*, pages 331–340, 2009.

[10] D. E. Bell and L. J. La Padula. Secure computer system: Unified exposition and multics interpretation. *Technical Report. ESD-TR-75-306*, 1976.

[11] K. J. Biba. Integrity considerations for secure computer systems. *Technical Report. ESD-TR-76-372*, 1977.

[12] M. Blum. How to Prove a Theorem So No One Else can Claim It. *Proceedings of the International Congress of Mathematicians*, 1(2):1444–1451, 1987.

[13] H.-J. Böckenhauer, D. Komm, R. Královic, R. Královic, and T. Mömke. On-line algorithms with advice. *Technical Report 614. ETH Zurich. Department of Computer Science*, 2009.

[14] M. Boreale, F. Pampaloni, and M. Paolini. Quantitative information flow, with a view. *Computer Security–ESORICS 2011*, pages 588–606, 2011.

[15] M. S. Branicky. Multiple lyapunov functions and other analysis tools for switched and hybrid systems. *Automatic Control, IEEE Transactions on*, 43(4):475–482, 1998.

[16] G. Brassard, C. Crepeau, and M. Yung. Everything in NP can be Argued in Perfect Zero-knowledge in a Bounded Number of Rounds. *Lecture Notes in Computer Science Advances in Cryptology EUROCRYPT 89*, pages 192–195, 1990.

[17] K. Chatzikokolakis, C. Palamidessi, and G. Smith. Measuring information leakage using generalized gain functions. *Computer Security Foundations Symposium (CSF), 2012 IEEE 25th*, pages 265–279, 2012.

[18] C.-T. Chen. *Linear system theory and design*. Oxford University Press, Inc., 1995.

[19] D. Clark, S. Hunt, and P. Malacaria. Quantitative information flow, relations and polymorphic types. *Journal of Logic and Computation*, 15(2):181–199, 2005.

[20] M. R. Clarkson, A. C. Myers, and F. B. Schneider. Belief in information flow. *Computer Security Foundations, 2005. CSFW-18 2005. 18th IEEE Workshop*, pages 31–45, 2005.

[21] S. D. A model of information. *Proceeding of the 9th National Computer Security Conference*, 1986.

[22] S. Dobrev, R. Královič, and D. Pardubská. Measuring the problem-relevant information in input. *RAIRO-Theoretical Informatics and Applications*, 43(03):585–613, 2009.

[23] S. Dobrev, R. Krlovi, and D. Pardubsk. Measuring the Problem-relevant Information in Input. *RAIRO - Theoretical Informatics and Applications*, 43(3):585–613, 2009.

[24] C. Dwork. Differential privacy. *Automata, languages and programming*, pages 1–12, 2006.

[25] C. Dwork, M. Naor, and A. Sahai. Concurrent Zero-Knowledge. *30th ACM symposium on the Theory of Computing*, pages 409–418, 1998.

[26] U. Feige, A. Fiat, and A. Shamir. Zero Knowledge Proof of Identity. *19th STOC*, pages 210–217, 1986.

[27] U. Feige and A. Shamir. Witness Indistinguishability and Witness Hiding Protocols. *22nd ACM Symposium on the Theory of Computing*, pages 416–426, 1990.

[28] U. Feige and A. Shamir. Zero Knowledge Proof of Knowledge in Two Rounds. *Lecture Notes of Computer Science Advances in CryptologyCRYPTO 89*, 435:526–544, 1990.

[29] U. Feigh, D. Lapidot, and A. Shamir. Multiple Non-Interactive Zero-Knowledge Proofs Under General Assumptions. *Journal on Computing*, 29(1):1–28, 1999.

[30] L. Feng and B. McMillin. A Survey on Zero-Knowledge Proofs. *Advances in Computers*, 94:25–69, 2014.

[31] L. Feng and B. McMillin. Quantification of information flow in a smart grid. *Computer Software and Applications Conference Workshops (COMPSACW), 2014 IEEE 38th International*, pages 140–145, 2014.

[32] L. Feng and B. McMillin. Information flow quantification framework for cyber physical system with constrained resources. *Computer Software and Applications Conference (COMPSAC), 2015 IEEE 39th Annual*, 2:50–59, 2015.

[33] L. Feng, B. McMillin, T. Paul, R. Thomas, and J. Kimball. *Information Flow Quantification in the Cyber-Physical System.* Submit to Dependable and Secure Computing, IEEE Transactions on, 2015.

[34] R. Focardi and R. Gorrieri. Classification of security properties:(part i: Information flow). *FOSAD 2000, LNCS 2171*, pages 331–396, 2001.

[35] T. Gamage and B. McMillin. Nondeducibility-based analysis of cyber-physical systems. *Critical Infrastructure Protection III*, pages 169–183, 2009.

[36] J. A. Goguen and J. Meseguer. Security policies and security models. *IEEE Symposium on Security and privacy*, 12, 1982.

[37] O. Goldreich. Concurrent Zero Knowledge with Timing, Revisited. *34th ACM Symposium on the Theory of Computing*, pages 332–340, 2002.

[38] O. Goldreich. Zero-knowledge twenty years after its invention. *IACR Cryptology ePrint Archive*, 2002:186, 2002.

[39] O. Goldreich and A. Kahan. How to Construct Constant-Round Zero-Knowledge Proof Systems for NP. *Journal of Cryptology*, 9(3):167–190, 1996.

[40] O. Goldreich and H. Krawczyk. On the Composition of Zero Knowledge Proof Systems. *SIAM Journal on Computing*, 25(1):169–192, 1996.

[41] O. Goldreich, S. Micali, and A. Wigderson. How to play any mental game. *Proceedings of the nineteenth annual ACM symposium on Theory of computing*, pages 218–229, 1987.

[42] O. Goldreich, S. Micali, and A. Wigderson. Proofs that Yield Nothing but Their Validity or All Languages in NP have Zero-Knowledge Proof Systems. *Journal of the ACM*, 38(3):691–729, 1991.

[43] O. Goldreich and E. Petrank. Quantifying Knowledge Complexity. *32nd FOCS*, pages 59–68, 1995.

[44] S. Goldwasser and S. Micali. Probabilistic encryption. *Journal of computer and system sciences*, 28(2):270–299, 1984.

[45] S. Goldwasser, S. Micali, and C. Rackoff. The Knowledge Complexity of Interactive Proof System. *SIAM Journal on Computing*, 18:186–208, 1989.

[46] R. Gradwohl, M. Naor, B. Pinkas, and G. N. Rothblum. Cryptographic and physical zero-knowledge proof systems for solutions of sudoku puzzles. *Theory of Computing Systems*, 44(2):245–268, 2009.

[47] M. A. Harrison, W. L. Ruzzo, and J. D. Ullman. Protection in operating systems. *Communications of the ACM*, 19(8):461–471, 1976.

[48] J. Hromkovi, R. Krlovi, and R. Krlovi. Information Complexity of Online Problems. *MFCS'10 Proceedings of the 35th international conference on Mathematical foundations of computer science*, pages 24–36, 2010.

[49] R. Impagliazzo and M. Yung. Direct minimum-knowledge computations. *Advances in CryptologyCrypto87*, pages 40–51, 1988.

[50] Y. Ishai, E. Kushilevitz, R. Ostrovsky, and A. Sahai. Zero-Knowledge from Secure Multiparty Computation. *STOC 07*, pages 21–30, 2007.

[51] A. M. Jaafar and A. Samsudin. Visual zero-knowledge proof of identity scheme: A new approach. *Computer Research and Development, 2010 Second International Conference on*, pages 205–212, 2010.

[52] G. Jain. Zero Knowledge Proofs: A Survey. *University of Pennsylvania*.

[53] W. Jiang and C. Clifton. A Secure Distributed Framework for Achieving k-anonymity. *Special Issue of the VLDB Journal on Privacy-Preserving Data Management*, pages 180–187, 2006.

[54] A. Juels and M. Sudank. A Fuzzy Vault Scheme. *International Symposium of Information Theory*, pages 408–425, 2002.

[55] R. M. Karp. Reducibility among Combinatorial Problems. *In Complexity of Computer Computations, Proc. Sympos*, pages 85–103, 1972.

[56] R. M. Karp. Reducibility among Combinatorial Problems. *Complexity of Computer Computations, Proc. Sympos*, pages 85–103, 1972.

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

[58] B. Köpf and D. Basin. An information-theoretic model for adaptive side-channel attacks. *Proceedings of the 14th ACM conference on Computer and communications security*, pages 286–296, 2007.

[59] B. Kopf and G. Smith. Vulnerability bounds and leakage resilience of blinded cryptography under timing attacks. *Computer Security Foundations Symposium (CSF), 2010 23rd IEEE*, pages 44–56, 2010.

[60] R. J. Lipton and L. Snyder. A linear time algorithm for deciding subject security. *Journal of the ACM (JACM)*, 24(3):455–464, 1977.

[61] G. Lowe. Quantifying information flow. *Computer Security Foundations Workshop, 2002. Proceedings. 15th IEEE*, pages 18–31, 2002.

[62] J. McLean. Security models and information flow. *Research in Security and Privacy, 1990. Proceedings., 1990 IEEE Computer Society Symposium on*, pages 180–187, 1990.

[63] J. McLean. Security Models and Information Flow. *Proceedings of the IEEE Symposium on Research in Security and Privacy*, pages 180–187, 1990.

[64] J. McLean. A general theory of composition for a class of possibilistic properties. *Software Engineering, IEEE Transactions on*, 22(1):53–67, 1996.

[65] B. McMillin and R. Akella. Verification and protection of confidentiality in an advanced smart grid. *System Science (HICSS), 2012 45th Hawaii International Conference on*, pages 2169–2175, 2012.

[66] R. C. Merkle, S. Member, Ieee, M. E. Hellman, and S. Member. Hiding information and signatures in trapdoor knapsacks. *IEEE Transactions On Information Theory*, 24:525–530, 1978.

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

[68] A. Mohr. A Survey of Zero Knowledge Proofs with Applications to Cryptography. *Southern University at Carbondale*, 2007.

[69] F. Pasqualetti, F. Dörfler, and F. Bullo. Attack detection and identification in cyber-physical systems–part i: Models and fundamental limitations. *arXiv preprint arXiv:1202.6144*, 2012.

[70] T. Paul. *Application of unified invariants for cyber physical systems in smart grids*. Missouri University of Science and Technology, 2015.

[71] T. Paul, J. W. Kimball, M. Zawodniok, T. P. Roth, B. McMillin, and S. Chellappan. Unified invariants for cyber-physical switched system stability. *Smart Grid, IEEE Transactions on*, 5(1):112–120, 2014.

[72] A. D. Pierro, C. Hankin, and H. Wiklicky. Approximate non-interference. *Journal of Computer Security*, 12(1):37–81, 2004.

[73] M. Prabhakaran, A. Rosen, and A. Sahai. Concurrent Zero Knowledge with Logarithmic Round-Complexity. *43rd FOCS*, pages 366–375, 2002.

[74] R. Richardson and J. Kilian. On the concurrent composition of zero-knowledge proofs. *Advances in CryptologyEUROCRYPT99*, pages 415–431, 1999.

[75] A. Rosen and O. Goldreich. *Concurrent zero-knowledge*. Springer, 2006.

[76] P. Y. Ryan and S. A. Schneider. Process algebra and non-interference. *Journal of Computer Security*, 9(1):75–103, 2001.

[77] A. Sabelfeld and A. C. Myers. Language-based information-flow security. *Selected Areas in Communications, IEEE Journal on*, 21(1):5–19, 2003.

[78] A. Shamir. How to Share a Secret. *Communications of the ACM*, 22:612–613, 1979.

[79] G. I. Simari. A Primer on Zero Knowledge Protocols http://cs.uns.edu.ar/ gis/publications/zkp-simari2002.pdf accessed on Oct. 12nd, 2011.

[80] D. D. Sleator and R. E. Tarjan. Amortized efficiency of list update and paging rules. *Communications of the ACM*, 28(2):202–208, 1985.

[81] G. Smith. On the foundations of quantitative information flow. *Foundations of Software Science and Computational Structures*, pages 288–302, 2009.

[82] G. Smith. Quantifying information flow using min-entropy. *Quantitative Evaluation of Systems (QEST), 2011 Eighth International Conference on*, pages 159–167, 2011.

[83] C. Tang and Z. Hao. Perfect Zero-Knowledge Argument of Knowledge with Negligible Error Probability in Two-Round for NP from Any One-way Permutation. *2010 International Conference on Communications and Mobile Computing*, 2010.

[84] S. Vadhan. The complexity of zero knowledge. *FSTTCS 2007: Foundations of Software Technology and Theoretical Computer Science*, pages 52–70, 2007.

[85] J. T. Wittbold and D. M. Johnson. Information flow in nondeterministic systems. *Research in Security and Privacy, 1990. Proceedings., 1990 IEEE Computer Society Symposium on*, pages 144–161, 1990.

[86] A. C. Yao. Theory and application of trapdoor functions. *Foundations of Computer Science, 1982. SFCS'08. 23rd Annual Symposium on*, pages 80–91, 1982.

[87] A. C. Yao. How to generate and exchange secrets. *Foundations of Computer Science, 1986., 27th Annual Symposium on*, pages 162–167, 1986.

VITA

Li Feng was born in Yinchuan, Ningxia Province, China. She received her Bachelor Degree in Information Engineering from Beijing University of Chemical Technology in 2002. Then, she graduated with honor and obtained her Master Degree in Computer Technology in the same university in 2005. She joined the Ph.D. program in Computer Science at the Missouri University of Science and Technology in August 2009. She received her Ph.D. degree in Computer Science from Missouri University of Science and Technology in December 2015. Her research advisor is Dr. Bruce McMillin. Li Feng's primary research interests are computer security in distributed systems, information flow analysis in cyber physical systems and interdisciplinary architecture.