國立臺灣大學管理學院資訊管理研究所

碩士論文

Graduate Institute of Information Management

College of Management

National Taiwan University

Master thesis

考慮隨機錯誤與惡意攻擊下資訊洩漏程度最小化之近似最佳化防禦資源分配與資訊切割配置策略

Near Optimal Defense Resource Allocation and Information Dividing-and-Allocation Strategies to Minimize Information Leakage Considering both Random Errors and Malicious Attacks

蘇至浩

Chih-Hao, Su

指導教授：林永松 博士

Advisor: Yeong-Sung, Lin, Ph.D.

中華民國 97 年 7 月

July, 2008

考慮隨機錯誤與惡意攻擊下資訊洩漏程度最小化之近似最佳化防禦資源分配與資訊切割配置策略

Near Optimal Defense Resource Allocation and Information Dividing-and-Allocation Strategies to Minimize Information Leakage Considering both Random Errors and Malicious Attacks

本論文係提交國立台灣大學
資訊管理學研究所作為完成碩士
學位所需條件之一部份

研究生：蘇至浩　撰

中華民國九十七年七月

博碩士論文電子檔案上網授權書

（提供授權人裝釘於紙本論文書名頁之次頁用）

本授權書所授權之論文為授權人在國立臺灣大學資訊管理學研究所 96 學年度第 二 學期取得博士、碩士學位之論文。

論文題目： 考慮隨機錯誤與惡意攻擊下資訊洩漏程度最小化之近似最佳化防禦資源配置與資訊切割配置策略

指導教授： 林永松 博士

茲同意將授權人擁有著作權之上列論文全文（含摘要），非專屬、無償授權本人畢業系所，不限地域、時間與次數，以微縮、光碟或其他各種數位化方式將上列論文重製，並得將數位化之上列論文及論文電子檔以上載網路方式，提供讀者基於個人非營利性質之線上檢索、閱覽、下載或列印。

□ 上列論文為授權人向經濟部智慧財產局申請專利之附件或相關文件之一（專利申請案號：　　　　），請於　　年　　月　　日後再將上列論文公開或上載網路。

☑ 因上列論文尚未正式對外發表，請於 99 年 7 月 23 日後再將上列論文公開或上載網路。

授權人簽名： 蘇至浩

中　華　民　國 97 年 7 月 15 日

國立臺灣大學碩士學位論文
口試委員會審定書

考慮隨機錯誤與惡意攻擊下資訊洩漏程度最小化之近似最佳化防禦資源分配與資訊切割配置策略

Near Optimal Defense Resource Allocation and Information Dividing-and-Allocation Strategies to Minimize Information Leakage Considering both Random Errors and Malicious Attacks

本論文係 蘇至浩 君（學號 R95725046）在國立臺灣大學資訊管理學系、所完成之碩士學位論文，於民國 97 年 7 月 15 日承下列考試委員審查通過及口試及格，特此證明

口試委員： 趙啟超

林永松

孫雅麗

所　　長： 陳靜枝

謝誌

隨著論文的完成，研究所生涯也將畫下句點。在這兩年的研究所生涯中，除了在學問方面有所增長外，待人處世的態度也學到了很多。在撰寫論文的過程中，培養了我嚴謹思考與認真負責的態度，雖然疲累卻也踏實，相信這會是一個寶貴且難得的經驗，也會在日後的回憶中，留下美好的一頁。

能夠完成研究所的學業，首先要感謝的是我的父母蘇振德先生與陳淑真女士，感謝他們無私無悔的付出，讓我能夠無後顧之憂的專注於我的論文之上，也由於他們的鼓勵，讓我在求學的道路上能更有動力的往前邁進。另外，特別要感謝的是我的指導教授林永松博士，林老師總是在我論文遇到困難時給予我適時的幫助，也教導了我面對困難時應該要有的態度。此外也感謝本系孫雅麗教授、莊裕澤教授、清大通訊趙啟超教授、輔大資工呂俊賢教授在口試時給予我論文的寶貴意見。

除此之外，我要感謝柏皓學長，從論文的方向以及模型的確立與問題的修正等等過程，學長都給予我非常大的幫助，也感謝學長在百忙之中還不忘關心我們的進度與適時的鼓勵。感謝中蓮學姊對於我的模型所提出的建議以及對於問題的解答。感謝家禎陪我度過了辛苦的研究所生涯，並且也在我論文的撰寫過程提供了寶貴的意見。感謝實驗室的戰友孜謙、奐廷、政祐、志元，因為有你們的提醒與互相鼓勵，給予了我往前邁進的動力。也謝謝研究所的朋友們子超、奕伃、小黑、以凡、立穎在苦悶的研究所生涯中給了我許多歡笑。感謝鼓勵我考台大研究所的朋友，讓我能有機會在這裡學習、成長。

最後要感謝身邊所有幫助過我的人，謝謝你們，讓我克服了人生中所遇到的困難。以此論文獻給我最尊敬的爺爺與奶奶，我愛你們。

論文摘要

論文題目：考慮隨機錯誤與惡意攻擊下資訊洩漏程度最小化之近似最佳化防禦資源配置與資訊切割配置策略

作者：蘇至浩　　九十七年七月

指導教授：林永松　博士

隨著資訊科技的進步以及儲存設備價格的遞減，不管是個人、企業體、或是政府單位皆大量使用電子化的方式儲存資訊。再者，伴隨著網路使用率的提升以及電子商務的出現，經由網路竊取資訊的犯罪行為也迅速的增加。像是釣魚行為或是安裝木馬程式於受害者的電腦以竊取資訊等的犯罪，對個人或企業體皆造成重大的傷害。因此，如何發展防禦策略以保護儲存在網路上的資訊，已經變成很重要的議題。

在這篇論文中，我們將攻防情境轉化成一個最小-最大化的雙層數學規劃問題。在內層的問題中，攻擊者想在有限的攻擊能量下藉由竊取資訊對網路造成最大的傷害；另一方面，在外層的問題中，防禦者想在有限的防禦預算下利用秘密分享的概念，最佳化防禦資源配置策略以及資訊切割與分配策略來最小化傷害。除此之外，防禦者也必須考慮到合法使用者對存取資訊的服務品質要求。為了解決這個問題，我們採用了拉格蘭日鬆弛法以及次梯度法。我們假設防禦策略已知下，先解決內層攻擊者的選徑問題，再根據內層解完後的結果藉由以次梯度法為基礎的演算法來調整防禦策略。

關鍵詞：最佳化、數學規劃、資訊竊取、拉格蘭日鬆弛法、資源配置、資訊切割、秘密分享

THESIS ABSTRACT

GRADUATE INSTITUTE OF INFORMATION MANAGEMENT NATIONAL TAIWAN UNIVERSITY

NAME：LILLIAN, CHUNG-LIEN TSENG   MONTH/YEAR：JULY 2006

ADVISER：YEONG-SUNG LIN

Near Optimal Defense Resource Allocation and Information Dividing-and-Allocation Strategies to Minimize Information Leakage Considering both Random Errors and Malicious Attacks

Information technology has been increasingly progressing, and the storage cost has been reducing. Thus, individuals, enterprises and government organizations are likely to store secret data through electronic way. Moreover, along with the rise of the use of network and the prevalence of e-commerce, the crime of information theft through network has grown in high-speed. Cyber crimes, like phishing or installing Trojan horse in victims' computers to steal information, will cause serious damage to individuals or enterprises. From the above reasons, to protect secret information stored on networks becomes an essential issue.

In this thesis, we formulate the attack-defense scenario as a min-max mathematical programming problem, which is a two-level mathematical problem. In the inner problem, the attacker wants to maximize the total damage by stealing information under limited attack power. In the outer problem, the defender wants to minimize the total damage by defense resource allocation and information-dividing under limited budget. In addition, the defender also has to consider QoS requirements of authorized users. In order to solve the considered problem, we use the Lagrangean Relaxation method and the subgradient method [14][15]. We solve the inner problem under a given defense strategy first, and then propose a subgrandient-based heuristic to adjust the defender’s strategy according to attacker’s attack strategy.

Keywords: Optimization, Mathematical Programming, Information Theft, Lagrangean Relaxation, Resource Allocation, Information Dividing, Secret Sharing

Table of Contents

Chapter 1 Introduction 1

1.1 Background 1

1.2 Motivation 5

1.3 Literature Survey 7

1.3.1 Survivability 7

1.3.2 Secret Sharing 11

1.3.3 Personal Data Objects Management 13

1.4 Proposed Approach 13

1.5 Thesis Organization 14

Chapter 2 Problem Formulation 16

2.1 Problem Description 16

2.2 Problem Formulation of the DRAID Model 19

2.3 Problem Formulation of the APS Model 31

Chapter 3 Solution Approach 35

3.1 Solution Approach for the APS Model 35

3.1.1 Lagrangean Relaxation Method 35

3.1.2 First-Stage Lagrangean Relaxation 39

3.1.3 Second-Stage Relaxation 51

3.1.4 Summary of the Solution Approach for APS Model 60

3.2 Solution Approach for the DRAID Model 62

Chapter 4 Computational Experiments 68

4.1 Computational Experiment with the APS Model 68

4.1.1 Simple Algorithm 1 68

4.1.2 Simple Algorithm 2 69

4.1.3 Experiment Environment 70

4.1.4 Experiment Results 73

4.1.5 Discussion of Results 99

4.2 Computational Experiment with the DRAID Model 102

4.2.1 Experiment Environment 102

4.2.2 Experiment Results 104

4.2.3 Discussion of Results 106

Chapter 5 Conclusion and Future Work 107

5.1 Conclusion 107

5.2 Future Work 109

References 111

List of Tables

Table 1-1 Survivability Definition Summary..9
Table 2-1 Problem Assumption and Description....................................23
Table 2-2 Given Parameters of the DRAID Model................................25
Table 2-3 Decision Variables of the DRAID Model................................26
Table 2-4 Given Parameters of the APS Model......................................31
Table 2-5 Decision Variables of the APS Model 32
Table 3-1 Algorithm to Solve (Sub 1.1') 41
Table 3-2 Algorithm to Solve (Sub 1.3)...44
Table 3-3 Heuristic_LR_1 Algorithm 49
Table 3-4Heuristic_LR_2 Algorithm 59
Table 3-5 LR Algorithm...61
Table 3-6 Heuristic_DRAID Algorithm 65
Talbe 3-7 Reallocate_Defense_Budget Algorithm..................................66
Table 3-8 Reallocate_Pieces Algorithm..66
Table 4-1 SA_1 Algorithm..68
Table 4-2 SA_2 Algorithm..69
Table 4-3 Experiment Parameter Settings for the APS Model 72
Table 4-4 Experiment Result under concave defense capability function (|*N*| = 25)..74
Table 4-5 Experiment Result under concave defense capability function (|*N*| = 64)..80
Table 4-6 Experiment Result under concave defense capability function (|*N*| = 100)..86
Table 4-7 Experiment of different defense capability function (|*N*| = 100) 92
Table 4-8 Experiment Parameter Settings for the DRAID Model 103
Table 4-9 Experiment Results of small networks (|N| = 25) 104
Table 4-10 Experiment Results of Medium-sized networks (|N| = 64) 104
Table 4-11 Experiment Results of Large networks (|N| = 100) 105

List of Figures

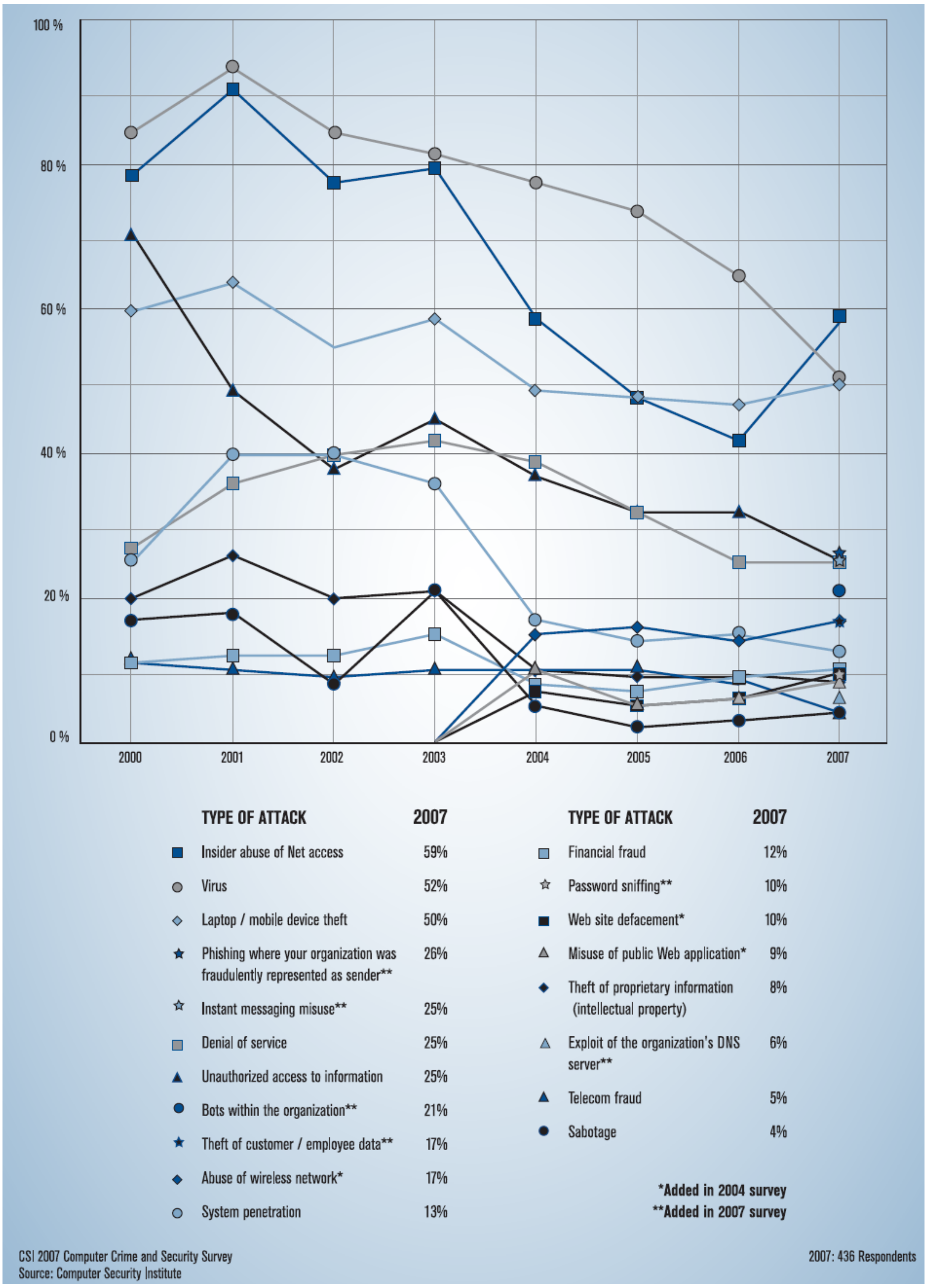
Figure 1-1Type of Attacks or Misuse (2007) 2
Figure 1-2 Dollar Amount losses by Type (2006).....................................3
Figure 1-3Dollar Amount losses by Type (2007)......................................4
Figure 2-1 Information Dividing-And-Allocation..................................20
Figure 2-2 Budget Allocation 20
Figure 2-3 Initial State of Attacker..20
Figure 2-4 Probing Neighbors..20
Figure 2-5 Attacking a Target..21
Figure 2-6 Final State of Network 21
Figure 2-7 Attack Tree..21
Figure 3-1 The Major Concepts of Lagrangean Relaxation Method.37
3-2 Flow Char for Lagrangean Relaxation Method 38
Figure 4-1 Susceptibility of Grid Networks under Different Scenarios (|N| = 25)..95
Figure 4-2 Susceptibility of Random Networks under Different Scenarios (|N| = 25) 95
Figure 4-3 Susceptibility of Scale-free Networks under Different Scenarios (|N| = 25) 95
Figure 4-4 Susceptibility of Grid Networks under Different Scenarios (|N| = 64)..96
Figure 4-5 Susceptibility of Random Networks under Different Scenarios (|N| = 64) 96
Figure 4-6 Susceptibility of Scale-free Networks under Different Scenarios (|N| = 64) 96
Figure 4-7 Susceptibility of Grid Networks under Different Scenarios (|N| = 100)..97
Figure 4-8 Susceptibility of Random Networks under Different Scenarios (|N| = 100) 97
Figure 4-9 Susceptibility of Scale-free Networks under Different Scenarios (|N| = 100) 97
Figure 4-10 Susceptibility of Grid Networks under Different defense capability function (|N| = 100)..98
Figure 4-11 Susceptibility of Random Networks under Different defense capability function (|N| = 100)..98
Figure 4-12 Susceptibility of Scale-free Networks under Different defense capability function (|N| = 100) 98
Figure 4-13 Susceptibility of diffrernt Networks and Topologies..........99

Figure 4-14 Survivability of Different Defense Budget Reallocation Strategy 105

Chapter 1 Introduction

1.1 Background

Following with the sizzling growth of Internet, the user group becomes more widespread, from exclusive military network (ARPANET) to academic network (NSF). Nowadays, network has expanded to commercial use (Internet) [12]. The inrush of all social class of users has not only built immeasurable business opportunities, but also made the network society a different kind of real-world society. Along with the rise of the use of network and the prevalence of e-commerce, the criminal act through network has grown in high-speed. Also, there are more and more types of attack of cyber crime, as shown on [1], which are listed in Figure 1-1. On the other hand, IT has been increasingly progressing, and the storage cost has been reducing. Thus, individuals, enterprises and government organizations are likely to store secret data through electronic way. However, enterprises may place the data on critical points of networks for users to access for the sake of information sharing. The convenience network brings usually accompanies threats. Cyber crimes, like phishing or installing Trojan horse in victims' computers to steal information, will cause serious damage to individuals or enterprises. From the former reasons, to protect the secret information stored on networks becomes an essential issue to discuss.



	TYPE OF ATTACK	2007		TYPE OF ATTACK	2007
■	Insider abuse of Net access	59%	□	Financial fraud	12%
●	Virus	52%	☆	Password sniffing**	10%
◆	Laptop / mobile device theft	50%	■	Web site defacement*	10%
★	Phishing where your organization was fraudulently represented as sender**	26%	△	Misuse of public Web application*	9%
☆	Instant messaging misuse**	25%	◆	Theft of proprietary information (intellectual property)	8%
□	Denial of service	25%	△	Exploit of the organization's DNS server**	6%
▲	Unauthorized access to information	25%	▲	Telecom fraud	5%
●	Bots within the organization**	21%	●	Sabotage	4%
★	Theft of customer / employee data**	17%			
◆	Abuse of wireless network*	17%			
○	System penetration	13%			

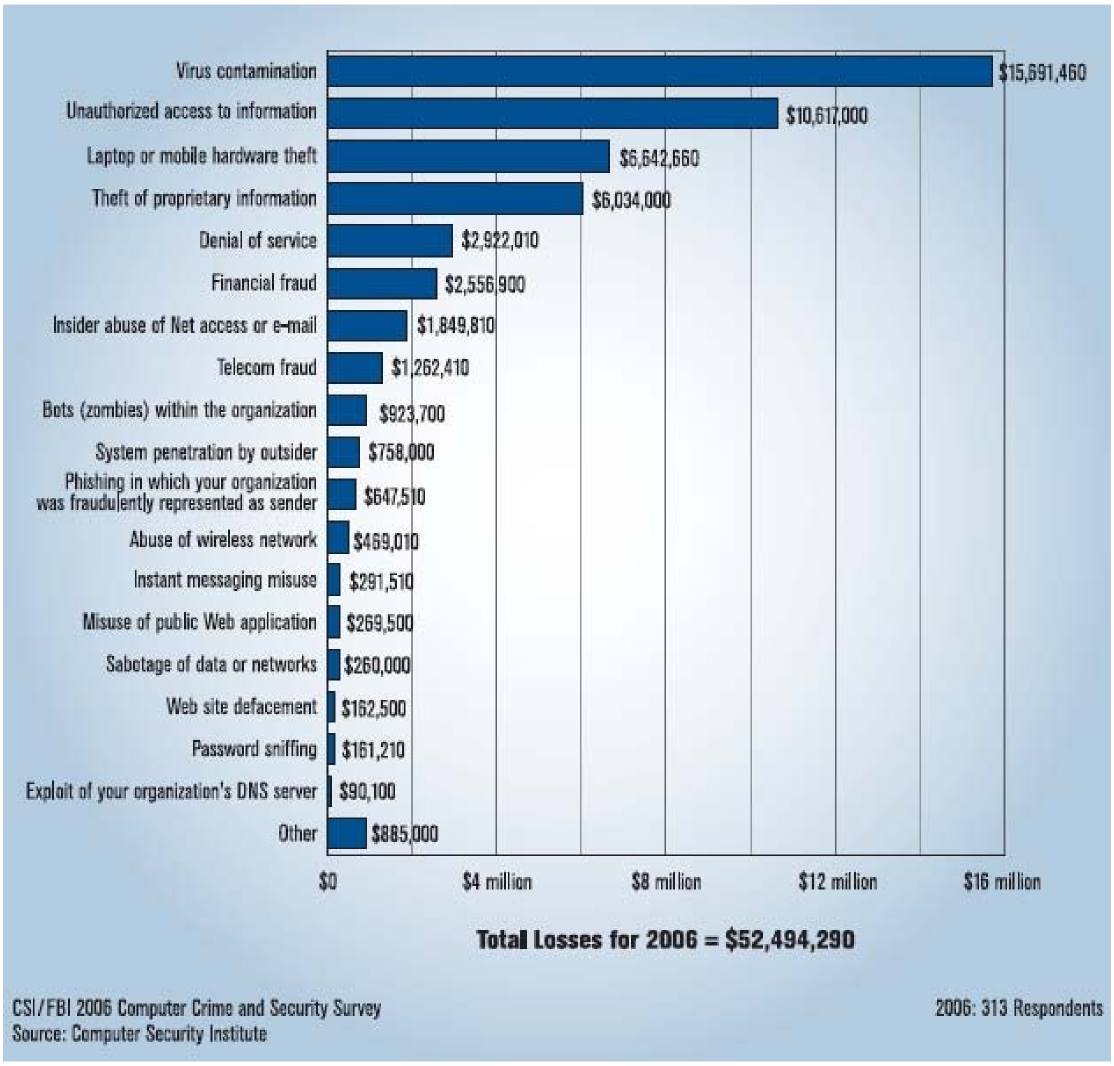
*Added in 2004 survey
**Added in 2007 survey

CSI 2007 Computer Crime and Security Survey
Source: Computer Security Institute

2007: 436 Respondents

Figure 1-1Type of Attacks or Misuse (2007)

Since the hackers can earn profits from stealing private properties and important information, almost every user is possibly attacked by all means of information stealing, like installing information-stealing components, such as backdoor programs, on victim computer or stealing information by alluring users to enter phishing website. In [2], it is indicated that Virus attacks, Unauthorized access to information, Theft of proprietary information, and Laptops or mobile hardware theft incidents have caused 74% of all losses, which are shown in Figure 1-2. In the following year, information leakage continuously causes great damage [1] as shown in Figure 1-3.

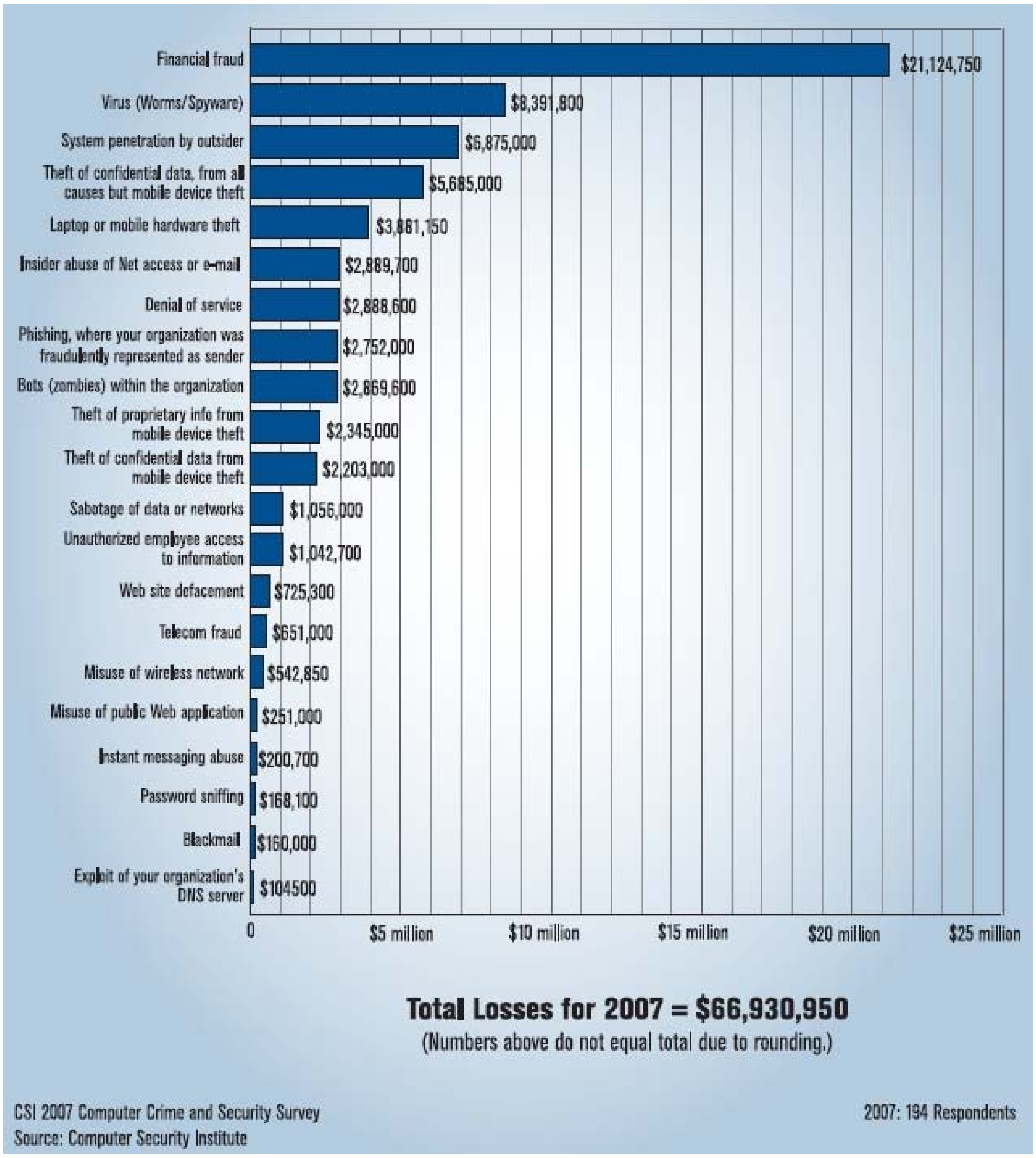


Total Losses for 2006 = $52,494,290

CSI/FBI 2006 Computer Crime and Security Survey
Source: Computer Security Institute

2006: 313 Respondents

Figure 1-2 Dollar Amount losses by Type (2006)



Total Losses for 2007 = $66,930,950
(Numbers above do not equal total due to rounding.)

CSI 2007 Computer Crime and Security Survey
Source: Computer Security Institute

2007: 194 Respondents

Figure 1-3Dollar Amount losses by Type (2007)

Because the attackers can successfully steal information without affecting the normal operation of system, the network operators are usually hard to notice that until the information has been revealed or used. Due to that reason, they must spend a lot to prevent such attack event, like setting up multilayer firewall to increase the difficulty to invade the system or secret sharing theorem [8][9][10][11], to decrease the damage when confronting information stealing. Although the defender can strengthen the system defense capability by resource allocation, the attacker can always find some vulnerability to attack. There is no so-called perfect-safe system. We need new metrics to describe the safeness of a system. In recent research, survivability [3][4][5][6][7] is one of the most general metric, which will be introduced in Chapter 1.3.

1.2 Motivation

With regard to the hazards to information security in recent years, information stealing was one of the most frequent incidents. Take Taiwan for example, there have been many serious cases [13], like 2 million accounts and passwords of HiNet host was stolen, and millions of personal information of users in PTT, which is one of the most popular BBS in Taiwan, was stolen, and the secret data of the Ministry of National Defense was stolen by hackers. Once the information was stolen, not only individual users may lose their rights and interests, the national security may also be victimized. In this commercial society, some technologies such as data mining are often used to enhance the relationship with customers. To make those technologies work, service providers will ask users to leave more detailed personal data, includes cell phone numbers, ID card numbers, and so on. As soon as user data was stolen, the attackers can do more things with that information, which make the damage more and more terrible. In [1], it is indicated that the loss of customer and proprietary data was the second-worst cause of financial loss. Moreover, retailers often speed up their transaction processes by wireless network due to its convenience. However, its lack of safeness makes it an evident target to hackers. Hackers would pilfer momentous information, like credit card numbers, through the vulnerability of wireless network. Take year 2007 for example, the American retailer TJX admitted the credit card data leakage event was probably through wireless network, and the victimized banks estimated that about 100 million accounts was stolen.

The information stealing will cause serious damage, which enforces the network operators to put more resources to prevent it. Nevertheless, attacker can always find a vulnerability to attack. Therefore, to the defenders, the main target is to allocate defense resources on network components to minimize the damage under limited budgets. The research [8] addressed that the secret sharing scheme in cryptography may deal with this problem. To sum up, in this paper we would like to discuss about how to exploit the concept of secret sharing and how to allocate defense resources to minimize the damage when information was stolen, and to solve it in mathematical way.

1.3 Literature Survey

1.3.1 Survivability

Nowadays, people depend more on IT system, so its dependability becomes a key point. Little harm may be likely to cause great deal of damage, especially for critical information systems [5]. The dependability is an integrated metric, which can represent reliability, availability, safety, confidentiality, and integrity [3]. It also has different meaning in different domain, for example, embedded control system needs highly available operation, database system needs highly availability operation, and weapons system needs high level of safety [5].

However, neither of those metrics can indicate if the system can still provide service under attack or threat. Presently, survivability [3][4][5][6][7] is usually used to evaluate the ability that the system can continuously provide basic service and recover on time from being under attack or damage. In commercial circumstances, it’s essential for an IT system to have such ability. Survivability has different meanings under different circumstances. For example, in telecommunications systems, its meaning is:

"A property of a system, subsystem, equipment, process, or procedure that provides a defined degree of assurance that the named entity will continue to function during and after a natural or man-made disturbance; e.g., nuclear burst. Note: For a given application, survivability must be qualified by specifying the range of conditions over which the entity will survive, the minimum acceptable level or post-disturbance functionality, and the maximum acceptable outage duration [4],"

while in aircraft combat, its meaning is:

"Aircraft combat survivability is the capability of an aircraft to avoid and/or withstand a man-made hostile environment. It can be measured by the probability the aircraft survives an encounter with the environment, PS. [4]".

The most common and general definition was proposed by Ellison in 1999: "Survivability is the capability of a system to fulfill its mission, in a timely manner, in the presence of attacks, failures or accidents [3]." Other definitions of survivability are listed in Table 1-1.

Table 1-1 Survivability Definition Summary

Researcher	Definition	Year
Westmark [6]	The ability of a given system with a given intended usage to provide a pre-specified minimum level of service in the event of one or more pre-specified threats	2004
Knight and Sullivan [4] [5]	Survivability is the ability [of a system] to continue to provide one or more alternate services (possibly degraded, less dependable, or different) in a given operating environment when various events cause damage to the system or its operating environment.	2000
Moitra, Soumyo, and Suresh [8]	The “degree to which a system has been able to withstand an attack or attacks, and is still able to function at a certain level in its new state after the attack.”	2002
Caldera and Jose [9]	Defined in terms of a survivable system where it is “available to fulfill its mission in a timely manner, in the presence of attacks, failures, or accidents.”	2000

Though there are many kinds of definitions of survivability, we can find there are some main components presented as follows [6]:

1. ***System**:*

The distributed network system environment for survivability, which has been defined, should be mentioned. In addition, whether the system is bounded or unbounded should be addressed.

2. ***Threat:***

Threats to a system can be categorized as accidental, intentional (malicious), or catastrophic. Accidental threats include software errors, hardware errors, and human errors. Intentional or malicious threats include sabotage, intrusion, or terrorist attacks. Catastrophic threats typically do not allow delivery of required service to the user, which includes acts of nature (thunderstorms, hurricanes, lightning, flood, earthquake, etc.), acts of war, and power failures.

3. ***Adaptability:***

In the event of a threat the system should have the capability to adapt to the threat and continue to provide the required service to the user.

4. ***Continuity of Service:***

Services should be available to the user as defined by the requirements of the system and expected by the user, even in the event of a threat.

5. ***Time**:*

Services should be available to the user within the time required by the system and expected by the user.

1.3.2 Secret Sharing

The concept of secret sharing was first developed and proposed by Adi Shamir and George Blakley in 1979 [9][11]. To simply describe it, this is a methodology to allocate secrets. The target is to separate secret *S* into *n* pieces to *n* specific individuals. We will say that each of those *n* individuals get a ‘share’ of secret *S*. In this mechanism, there is usually a role named *dealer* which is responsible for segmenting secrets. The dealer distributes the secret to *n* individuals of the group, and sets a *threshold t*, which means that we need at least *t* shares to get *S*, or we cannot get *S*. We usually call such system a (*t*, *n*)-threshold scheme or (*n*, *t*) secret sharing. From the above explanation, we will know that we can appropriately adjust system security and operating efficiency through altering threshold *t*. The system is safer when *t* is lager, and is more efficient otherwise.

In this era of technology, people exchange lots of information on computers and networks. But even in such public circumstance, some information just belongs to a specific individual and should not be easily acquired by other people. However, in some applications, we may not only need to let an individual to have a secret but also let many people could share this information together for safeness or fairness sake.

For example, assume there is a bank which stores its cash and gold in a vault with a combination to unlock it. For the business requirement and convenience, five managers of the bank can unlock the vault; for the safeness, there need to be at least three managers present when unlocking the vault. How should the system engineers design that combination of the vault of the bank? Now the concept of secret sharing may be helpful. One of the possible solutions is to design an algorithm and separate the cipher into five parts. The managers need to enter at least three parts of ciphers to unlock the vault. This solution thus solves the secret sharing problem.

In [9], the author shows how to divide data *D* into n pieces in such way *D* can be recombined from at least *k* pieces ($k \leq n$), but obtaining less than *k* pieces reveals no information about *D*. In addition, the author indicates some of the useful properties of this (*k*, *n*) threshold scheme presented as follow:

1. The size of each piece does not exceed the size of the original data.
2. When *k* is fixed, pieces can be dynamically added or deleted without affecting the other pieces.
3. It is easy to change the pieces without changing the original data *D*.
4. We can get a hierarchical scheme in which the number of pieces needed to determine *D* depends on their importance.

1.3.3 Personal Data Objects Management

Many services and application in the network need personal data of users for authentication and other commercial purposes when users connect to these services. However, with more Internet services offered, a user may have to maintain the same personal data in different places. To overcome this predicament, the methodology Personal Data Object Management was proposed [10]. It's also an application of secret sharing, which cuts personal data object into many chunks and restore it to complete data with some parts of chunks through encoding way. Moreover, the methodology proposed in [10] also can be treated as a global storage service [18], which people and applications can use to access required data anywhere and anytime. To achieve reliability of data, Erasure Coding can be used [17]. It divides data into *m* fragments and recode them into *n* fragments, where $n > m$. Any *m* of the coded fragments are sufficient to reconstruct the original data. The rate of encoding is

$r = m / n$, and the storage cost is multiplied by $1 / r$.

1.4 Proposed Approach

In this paper, we formulate the attack-defense scenario as a min-max mathematical programming problem, which is a two-level mathematical problem.

In the inner problem, the attacker wants to maximize the total damage by stealing information under limited attack power. We name the inner problem as “attack-path selecting problem (APS)”. In the outer problem, the defender wants to minimize the total damage by defense resource allocation and information-dividing under limited budget. In addition, the defender also has to consider QoS requirements of authorized users. We name the outer problem as “defense resource allocation and information dividing problem (DRAID).” This is a mixed and linear programming problem, which is complex and difficult.

In order to solve it, we use the Lagrangean Relaxation method and subgradient method [14][15]. We solve the inner problem under a given defense strategy first, and then propose a subgrandient-base heuristic to adjust the defender’s strategy according to attacker’s attack strategy.

We also evaluate a network’s survivability by calculating the percentage of un-stolen information. The higher the result is, the better the defense strategy is. The comparisons of networks’ survivability under different defense strategies and topologies are presented in Chapter 4.

1.5 Thesis Organization

The remainder of the thesis is organized as follows. In Chapter 2, the formulations of the APS and the DRAID problems are proposed. In Chapter 3, the solution approaches to the APS and the DRAID problems are presented. In Chapter 4, the computational results of the APS and the DRAID problems are presented. Finally, we present our conclusions and indicate possible directions of future research in Chapter 5.

Chapter 2 Problem Formulation

2.1 Problem Description

There is a lot of secret information in the real world, which often becomes the target of attackers because of its value. With consideration of security, the defender can use the technique of secret sharing to divide information. It will enforce attackers to make more efforts to recombine information. This makes information like a divided treasure map, and the attacker must collect all pieces of this treasure map to recombine it completely to find the treasury. But for authorized users, the risk that they cannot access information successfully will increase if they also have to get all pieces of information. Therefore, how to design the threshold of piece’s number to recombine information becomes an important issue.

Imagine that there are many connected nodes in a network. Each node constructs a domain and each edge represents the inter-domain connection. Namely, this network is an Autonomous-System (AS) level Internet. For an attacker, he/she must compromise a node first so that he/she can probe the neighbor nodes of this node to attack, and so on. This scenario implies the concept of defense-in-depth, which means that the attacker has to probe and attack nodes one by one instead of launching attack freely.

Therefore, the question that the defender wants to aim at is how to divide secret information appropriately and allocate those pieces (divided information) to nodes in the network. Additionally, the defender also has to design the threshold (piece's number) to recombine information and allocate defense resource to nodes to strengthen their defense capability under limited budget in order to minimize damage, which is incurred by information leakage. The defender must consider not only the security issue but also the authorized users' requirements. First of all, an authorized user will have different availability requirements for different information, but he/she may not access some pieces of information successfully because of random error occurring on nodes. Secondly, an authorized user will have different requirements of timeliness for different information, but he/she may not get complete information in time because of his/her distance to targeted nodes.. From above reasons, the defender must consider about both security and the user's QoS requirements at the same time when he/she designs a defense strategy.

For the attacker, he/she has to compromise nodes with target pieces one by one to recombine information under limited attack power to maximize the damage incurred by information leakage to the network.

In order to evaluate the performance of the defense strategy, we analyzed the survivability and susceptibility of the network. This calculation of these two metrics is shown in the following equations. S_g represents the value of information g, and then the metrics of network susceptibility and survivability can be presented as

$$\text{Susceptibility (\%)} = \frac{\sum_{g' \in \text{information that is recovered}} S_{g'}}{\sum_{g \in \text{all information in the network}} S_g} \times 100\%, \text{ and}$$

$$\text{Survivability (\%)} = 1 - \text{Susceptibility},$$

respectively. Susceptibility represents the percentage that information is recovered by the attacker and survivability represents the percentage that information is not recovered by the attacker.

2.2 Problem Formulation of the DRAID Model

In the DRAID Model, we try to transform the attack-defense scenario into an optimization problem. The attacker wants to maximize total damage to the network with limited attack power; the defender wants to minimize the maximized damage by allocating limited defense resource and dividing information. In this scenario, we assume that the attacker and the defender have global information about the target network. This is the worst case which makes the research more general.

Initially, the attacker is on a dummy node *o*, which is connected to the entry nodes of target network by artificial links. The attacker has to compromise one of the entry nodes and then interpenetrates the network until his/her attack power is exhausted. His/her ultimate goal is to maximize the total damage of the target network by stealing information.

The defender has to design an information-dividing-and-allocating strategy and a defense resource allocation strategy to minimize the maximized damage incurred by information leakage. The more defense resource a node is allocated to, the more robust it is. In other words, the attacker has to consume more attack power to compromise a node. Nevertheless, a node still has some defense capability even if no defense resource is allocated to it, so that the attack has to consume some attack power to compromise it. The attack-defense scenario is represented in Figures 2-1 ~

2-7.

The Attack-Defense Scenario

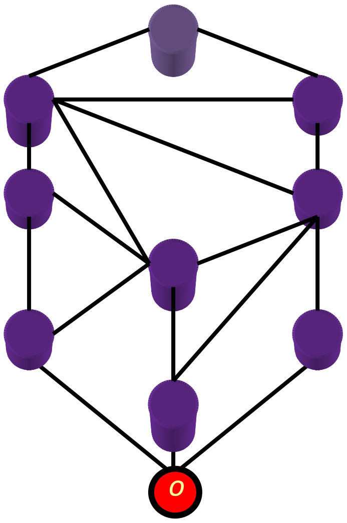


Figure 2-2 Budget Allocation

The defender allocates the defense resources to nodes to strengthen their defense capability.

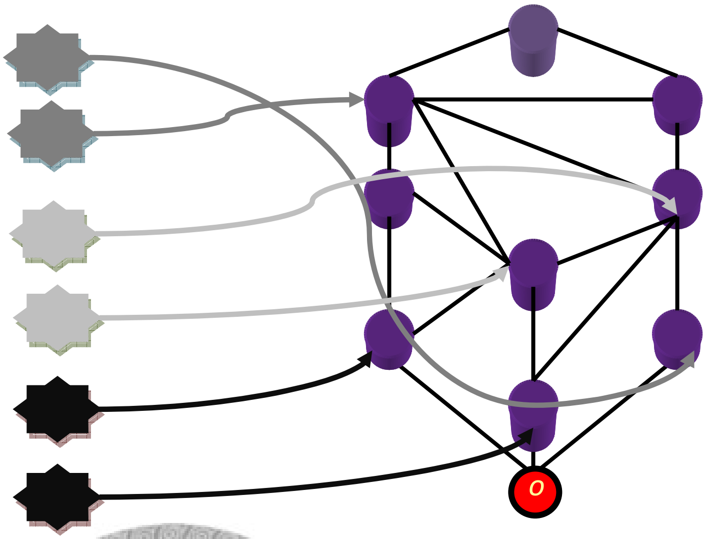


Figure 2-1 Information Dividing-And-Allocation

The defender divides information and allocates pieces to nodes in the target network.

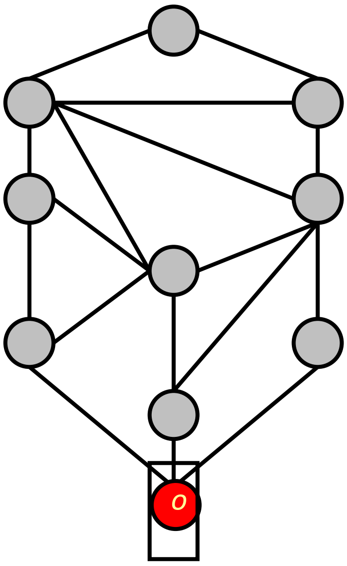


Figure 2-3 Initial State of Attacker

Initially, the attacker is on dummy node *o*, which is connected to entry nodes of the target network by artificial links.

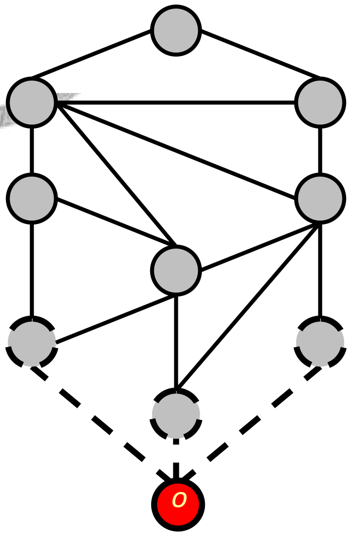
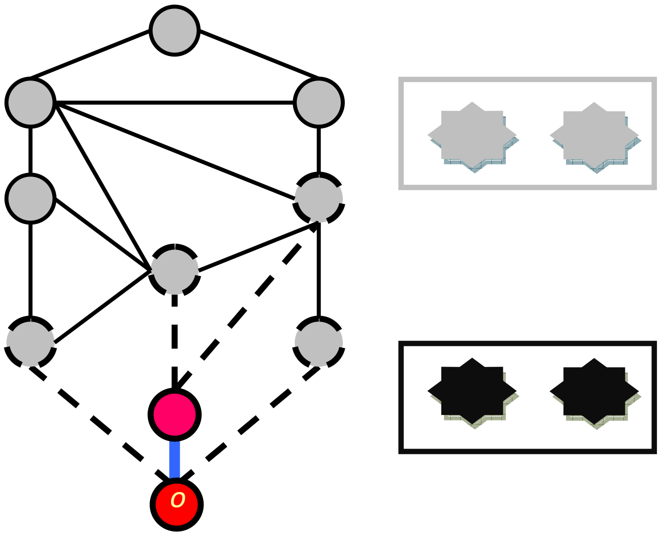


Figure 2-4 Probing Neighbors

The attacker probes the neighbors of node *o* to collect information about the threshold of attack power to compromise them.



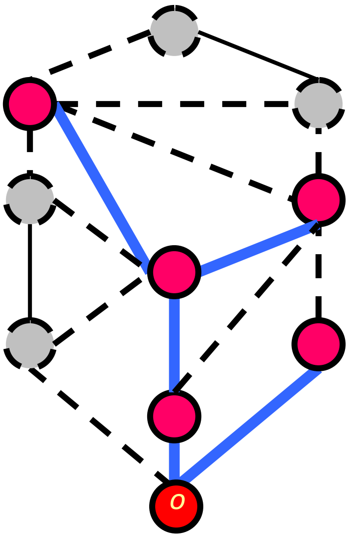


Figure 2-5 Attacking a Target

The attacker compromises a node, which has been probed, to steal pieces and then probes its neighbors.

Figure 2-6 Final State of Network

The attacker continuously launches attack to compromise nodes until his/her attack power is exhausted. In the meanwhile, the attacker recombines some information and causes damage to the network.

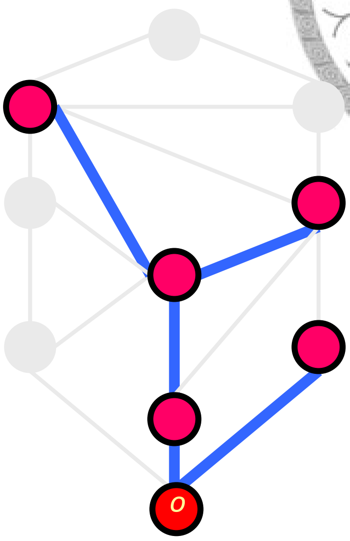
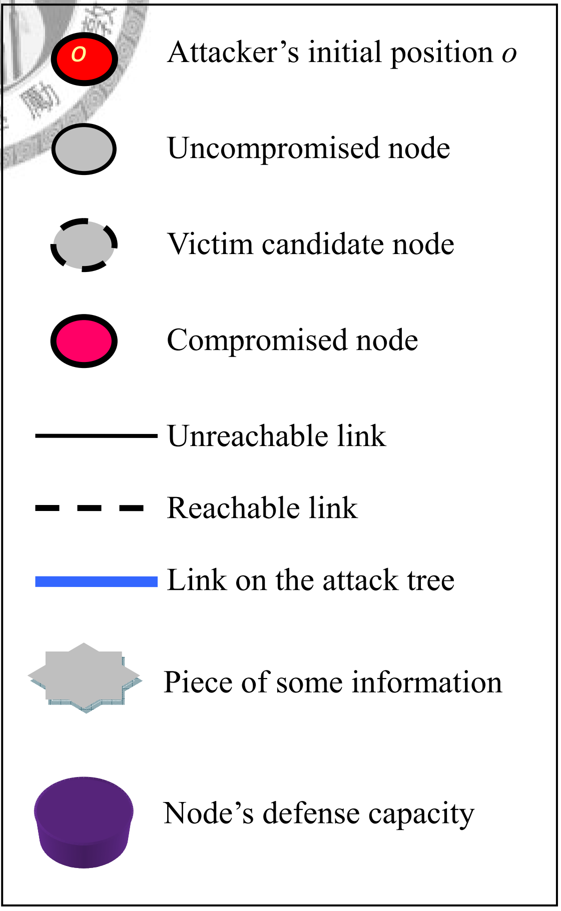


Figure 2-7 Attack Tree

The attack tree is constructed after the attacker terminates attacking.



Moreover, this problem involves not only malicious attack, but also random error. Therefore, there are two kinds of use of defense budget: one is on strengthening nodes' defense capability; the other is on decreasing the probability that random errors occur on nodes. Specifically, the defender must consider both users' QoS requirements and system safety at the same time when designing a defense strategy.

During the dynamic attack-defense process, the attacker will find the best attack path under limited attack power; the defender will adjust defense strategy according to attacker's action to minimize damage. The assumptions and description of this model are shown in Table 2-1.

Table 2-1 Problem Assumption and Description

Assumption：

- Both attacker and defender have complete information about the target network.
- Both the attacker and the defender have budget limitations.
- Only node attacks are considered.
- The target network is at AS-level.
- A node is only subject to attack if a path exists from attacker’s position to that node, and all the intermediate nodes on the path have been compromised.
- A node is compromised if the attack budget applied to the node is equal to or more than the defense capability of the node.
- Both random error and malicious attacks are considered.
- A user has different availability requirement to different information.
- A user has different requirement of the maximal tolerable waiting-time to different information when recover information.
- Each node has a random access error probability to legitimate users.
- An attacker can recover information only when obtaining at least the fixed number of pieces of information.
- Each node has a capacity limitation.

Given：

- The defense budget that prevents a node from being compromised and decreases the nodal random access error probability
- Attack budget
- Damage incurred by information leakage, i.e., the information value
- Attacker’s initial position *o*
- The network topology and the network size
- The set of all sensitive information
- The capacity of each node

- The minimal time to access pieces on each node
- Each information’s size

Objective：

- To minimize the maximized total damage

Subject to：

- The node to be attacked must be connected to the existing attack tree
- The total defense cost to prevent node from being compromised and to decrease random access error on nodes must be no more than a given value
- The total attack cost must be no more than a given value
- The attack budget applied to a node must equal to the node’s defense capability
- Attacker must obtain at least a given number of pieces when recovering information
- A node contains at most one piece of each information
- The total number of information pieces on all nodes must equal to the number of pieces that information is divided into
- The threshold of number of pieces to recover information must be no more than number of pieces that information is divided into
- The sum of pieces’ size on a node must be no more than its capacity
- Each information’s availability must be at least a given value without attacking
- The minimal waiting time to recover information must be no more than a given value without attacking and random access error on nodes

To determine：

- Defender: budget allocation and information-dividing-and-allocation strategy
- Attacker: which information to steal, which nodes to attack and how much attack budget to apply to nodes

We formulate this problem as a min-max mathematical programming problem. The given parameters of this model are shown in Table 2-2.

Table 2-2 Given Parameters of the DRAID Model

Given Parameters	
Notation	**Description**
N	The index set of all nodes in the network
W	The set of all O-D pairs, where the origin is the attacker's initial position (a dummy node o), which is connected to the entry nodes of target network by artificial links, and the destinations are the nodes in the target network
P_w	The index set of all candidate paths of an O-D pair w, where $w\in W$
δ_{pi}	An indicator function, which is 1 if node i is on path p, and 0 otherwise (where $i\in N$, $p\in P_w$)
G	The index set of all sensitive information in the network
S_g	Damage incurred by leaking at least k_g pieces of information g, where $g\in G$
m_g	The size of information g, where $g\in G$
n_g	The number of piece that information g will be divided into, where $g\in G$
R_g	The availability requirement of information g without attacking, where $g\in G$
T_g	The maximal tolerable waiting time for a user to recover information g without attacking, where $g\in G$
$H(k_g,m_g)$	The size of each piece of information g, where $g\in G$
A	The total attack budget
B	The total defense budget to enforce the nodal defense capability and to decrease the nodal random access error probability
v_i	The capacity of node i, $i\in N$
t_i	The minimal time to access pieces on node i, $i\in N$
$O(D)^j$	The function to retrieve the j^{th} largest object in D, where D is a set with comparable objects.

In this problem, if the attacker gets at least the threshold of information pieces to recombine information g which will cause damage, S_g, to the target network. The goal of an attacker is to collect at least threshold of information pieces to maximize the sum of S_g. The decision variables of this model are shown in Table 2-3.

Table 2-3 Decision Variables of the DRAID Model

Decision Variables	
Notation	**Description**
a_i	The attack budget applied to node *i,* where $i \in N$
x_p	1 if path p is selected as the attack path, and 0 otherwise (where $p \in P_w$)
Z_g	1 if at least k_g pieces of information g are stolen, and 0 otherwise (where $g \in G$)
y_i	1 if node i is compromised; and 0 otherwise (where $i \in N$)
$b^c{}_i$	The budget allocated to protect node i from being compromised, where $i \in N$
$b^e{}_i$	The budget allocated to node i to make the probability that random access error occurs on node i decrease, where $i \in N$
$\hat{a}_i(b^c{}_i)$	The threshold of the attack budget required to compromise node i, i.e., the defense capability of node i, where $i \in N$
$P(b^e{}_i)$	The probability that random access error occurs on node i, $i \in N$
k_g	The threshold of the number of information pieces required to recover information g, where $g \in G$
σ_{gi}	1 if node i contains one piece of information g, and 0 otherwise (where $i \in N$, $g \in G$)

The objective function and constraints are as follows:

Objective function:

$$\min_{k_g, b^c{}_i, b^e{}_i, \sigma_{gi}} \max_{y_i, a_i, z_g} \sum_{g \in G} S_g Z_g \qquad \text{(IP 1)}$$

Subject to:

$$\sum_{p \in P_w} x_p = y_i \qquad \forall i \in N, w = (o,i) \qquad \text{(IP 1.1)}$$

$$\sum_{p \in P_w} x_p \leq 1 \qquad \forall\, w \in W \qquad \text{(IP 1.2)}$$

$$\sum_{w \in W} \sum_{p \in p_w} x_p \varsigma_{pi} \leq |N-1| y_i \qquad \forall i \in N \qquad \text{(IP 1.3)}$$

$$x_p = 0 \; or \; 1 \qquad \forall\, p \in P_w, w \in W \qquad \text{(IP 1.4)}$$

$$y_i = 0 \; or \; 1 \qquad \forall\, i \in N \qquad \text{(IP 1.5)}$$

$$0 \leq b^c{}_i \leq B \qquad \forall\, i \in N \qquad \text{(IP 1.6)}$$

$$0 \leq b^e{}_i \leq B \qquad \forall\, i \in N \qquad \text{(IP 1.7)}$$

$$\sum_{i \in N} b^c{}_i + \sum_{i \in N} b^e{}_i \leq B^c \qquad \text{(IP 1.8)}$$

$$0 \leq a_i \leq \hat{a}_i(b^c{}_i) \qquad \forall\, i \in N \qquad \text{(IP 1.9)}$$

$$\sum_{i \in N} a_i \leq A \qquad \text{(IP 1.10)}$$

$$\hat{a}_i(b^c{}_i) y_i \leq a_i \qquad \forall\, i \in N \qquad \text{(IP 1.11)}$$

$$k_g Z_g \leq \sum_{i \in N} \sigma_{gi} y_i \qquad \forall\, g \in G \qquad \text{(IP 1.12)}$$

$$\sum_{i \in N} \sigma_{gi} = n_g \qquad \forall\, g \in G \qquad \text{(IP 1.13)}$$

$$k_g \le n_g \qquad \forall g \in \mathrm{G} \qquad \text{(IP 1.14)}$$

$$Z_g = 0 \text{ or } 1 \qquad \forall\, g \in G \qquad \text{(IP 1.15)}$$

$$\sigma_{gi} = 0 \text{ or } 1 \qquad \forall g \in G, i \in N \qquad \text{(IP 1.16)}$$

$$\sum_{g\in G} H(k_g, n_g, m_g)\sigma_{gi} \le v_i \qquad \forall\, i \in N \qquad \text{(IP 1.17)}$$

$$\sum_{q=k_g}^{n_g} (C_q^{n_g} \prod_{j=n_g-q+1}^{n_g} (\underset{i\in N}{O}((1-p(b^e{}_i))\sigma_{gi})^j) \prod_{l=1}^{n_g-q} \underset{i\in N}{O}(p(b^e{}_i)\sigma_{gi})^l) \ge R_g \qquad \forall g \in G \quad \text{(IP 1.18)}$$

$$\sum_{j=0}^{k_g-1} \underset{i\in N}{O}(t_i\sigma_{gi})^{n_g-j} \le T_g \qquad \forall g \in G. \quad \text{(IP 1.19)}$$

Explanation of the mathematical formulation：

- Objective function: The objective is to minimize the maximized total damage, $\sum_{g\in G} S_g Z_g$, incurred by compromising at least k_g nodes, which have pieces belong to information *g*, where k_g is the threshold to recombine complete information g. In the inner problem, an attacker tries to maximize the damage to the target network by deciding which nodes or groups to attack, i.e., the y_i value of each node *i* and the z_g value of each group *g*. In the outer problem, the defender tries to minimize the damage caused by the attacker by allocating the defense resources, $b^c{}_i$ and $b^e{}_i$, to each node appropriately and designing the information dividing-and-allocating strategy (k_g, σ_{gi}).

- Constraint (IP 1.1) restricts that a node is chosen for attack if and only if the attacker finds a path between his initial position *o* and the target node.

- Constraint (IP 1.2) restricts that at most one of the candidate paths of an OD pair w is selected.

- Constraint (IP 1.3) restricts that a node can be transited at most |*N*|-1 times. This constraint also ensures that there is no cycle and all nodes on the attack path are compromised.

- Constraints (IP 1.4) and (IP1.5) are integer constraints, both of which restrict that the value of x_p, y_i to be 0 or 1.

- Constraints (IP 1.6), (IP 1.7), and (IP 1.8) restrict the amount of defense budget, $b^c{}_i$ and $b^e{}_i$, that can be allocated to each node *i*. The total allotted defense budget, $\sum_{i\in N} b^c{}_i$ and $\sum_{i\in N} b^e{}_i$, must not exceed the defense budget *B*.

- Constraint (IP 1.9) restricts that attack budget a_i can be applied to node *i* cannot exceed the node's defense capacity $\hat{a}_i(b^c{}_i)$.

- Constraint (IP 1.10) restricts that the total allotted attack cost, $\sum_{i\in N} a_i$, must not exceed the attack budget *A*.

- Constraint (IP 1.11) restricts that a node is compromised if and only if attack budget applied to it is no less than its defense capability.

- Constraint (IP 1.12) restricts that information *g* is recover successfully if and only if the attacker compromises at least k_g nodes which have pieces of information *g*.

- Constraint (IP 1.13) restricts that a node can contain at most one piece of the same group.

- Constraint (IP 1.14) restricts that the threshold of number of information pieces to recover information cannot exceed the number of pieces that information is divided into.

- Constraints (IP 1.15) and (IP 1.16) are integer constraints both of which restrict that the value of Z_g, σ_{gi} to be 0 or 1.

- Constraint (IP 1.17) restricts that the total size of pieces on node *i* cannot exceed its capacity.

- Constraint (IP 1.18) is a QoS constraint, which restricts the information's availability .

Constraint (IP 1.19) is a QoS constraint, which restricts the maximal tolerable waiting time to recover information.

2.3 Problem Formulation of the APS Model

In APS model, we formulate an attacker’s behavior as a mathematical optimization problem, which is the inner problem of DRAID model. We can predict the future actions of an attacker by resolving this problem and then design the best defense strategy (budget allocation strategy and pieces allocation strategy). After this problem has been solved, its outcome is regarded as an input for DRAID problem. We can use the outcome from solving APS problem to develop a better defense strategy for DRAID model.

The assumptions and attack procedures of APS problem are the same as those of DRAID problem. We formulate APS problem as a mathematical maximization programming problem. Table 2-4 shows the given parameters.

Table 2-4 Given Parameters of the APS Model

Given Parameters	
Notation	**Description**
N	The index set of all nodes in the network
W	The set of all O-D pairs, where the origin is a dummy node o, which is connected to the entry nodes of target network by artificial links, and the destinations are the nodes in the target network
G	The index set of all sensitive information in the network
P_w	The index set of all candidate paths of an O-D pair w, where $w \in W$
A	The total attack power
S_g	Damage incurred by stealing at least k_g pieces of information g, where $g \in G$
$\hat{a}_i(b^c{}_i)$	The threshold of the attack power required to compromise node i, i.e., the defense capability of node i, where $i \in N$
k_g	The threshold of information pieces required to recombine complete

	information g, where $g \in G$
δ_{pi}	An indicator function, which is 1 if node i is on path p; and 0 otherwise (where $i \in N$, $p \in P_w$)
σ_{gi}	An indicator function, which is 1 if node i has one piece of information g, and 0 if node i has no piece of information g (where $i \in N$, $g \in G$)

Note that $\hat{a}_i(b^c{}_i)$, k_g, and σ_{gi} are decision variables in DRAID model but given parameters in APS model. $\hat{a}_i(b^c{}_i)$ is a function of $b^c{}_i$ increasing with $b^c{}_i$, which represents the defense capability of node i. k_g is a threshold of piece's number required to get complete information g. σ_{gi} indicates if node i has one piece of information g.

Table 2-5 shows the decision variables of the APS model, and the formulation of the APS model is shown below it.

Table 2-5 Decision Variables of the APS Model

Decision Variables	
Notation	**Description**
a_i	Attack power applied to node i, where $i \in N$
y_i	1 if node i is compromised; and 0 otherwise, where $i \in N$
Z_g	1 if at least k_g pieces of information g are stolen, and 0 otherwise, where $g \in G$
x_p	1 if path p is selected as the attack path; and 0 otherwise, where $p \in P_w$

Objective function:

$$\max_{y_i, a_i, z_g} \sum_{g\in G} S_g Z_g = \min_{y_i, a_i, z_g} -\sum_{g\in G} S_g Z_g$$ (IP 2)

Subject to:

$$\sum_{p\in P_w} x_p = y_i \qquad \forall i \in N, w=(o,i)$$ (IP 2.1)

$$\sum_{p\in P_w} x_p \le 1 \qquad \forall\, w \in W$$ (IP 2.2)

$$\sum_{w\in W}\sum_{p\in p_w} x_p \varsigma_{pi} \le |N-1| y_i \qquad \forall i \in N$$ (IP 2.3)

$$x_p = 0\ or\ 1 \qquad \forall\, p \in P_w, w \in W$$ (IP 2.4)

$$y_i = 0\ or\ 1 \qquad \forall\, i \in N$$ (IP 2.5)

$$0 \le a_i \le \hat{a}_i(b^c{}_i) \qquad \forall\, i \in N$$ (IP 2.6)

$$\sum_{i\in N} a_i \le A$$ (IP 2.7)

$$\hat{a}_i(b^c{}_i) y_i \le a_i \qquad \forall\, i \in N$$ (IP 2.8)

$$k_g Z_g \le \sum_{i\in N} \sigma_{gi} y_i \qquad \forall\, g \in G$$ (IP 2.9)

$$Z_g = 0 \text{ or } 1 \qquad \forall g \in G.$$ (IP 2.10)

Explanation of the mathematical formulation：

- Objective function: The objective of the formulation is to maximize the total damage by getting more complete information. First the attacker selects which information to steal and then compromises nodes which have pieces of related information.

- Constraints (IP 2.1) ~ (IP 2.5) are the same as Constraints (IP 1.1) ~ (IP1.5) in the DRAID model. Combination of these constraints restrict attacker must compromise nodes one by one and ensure there is no cycle.

- Constraints (IP 2.6), (IP 2.7), and (IP 2.8) are the same as Constraints (IP 1.10), (IP 1.11), and (IP 1.12) in the DRAID model. These constraints restrict the attack budget which is applied to each node.

- Constraints (IP 2.9) and (IP 2.10) are the same as Constraints (IP 1.13) and (IP 1.16) in the DRAID model. These constraints restrict attacker must get at least the threshold of number of information pieces to recover information.

Chapter 3 Solution Approach

3.1 Solution Approach for the APS Model

3.1.1 Lagrangean Relaxation Method

In order to solve large-scale mathematical programming problems, many approaches had been proposed in the 1970s [16]. Most of them used the divide-and-conquer technique to separate a complex problem into several relatively uncomplicated sub-problems and solve them respectively, and Lagrangean relaxation method is one of those excellent approaches. Due to Lagrangean relaxation method's flexibility and ability to help us to find the bounds of the optimal objective value, it has become one of the most popular tools for solving optimization problems. Besides, it can also assist us to develop effective heuristic algorithms to our problems. It can be used to solve integer programming, linear programming, combinatorial optimization, and non-linear programming problems [15].

The basic concept of Lagrangean relaxation method is to remove some complicated constraints to objective function of the primal problem (P) with associated multipliers (μ), and then the original primal problem (P) will be transformed into a so-called Lagrangean relaxation problem (LR_μ). This concept would be inspired because of the observation that many complex integer programming problems can be formulated as a relatively easy problem that is complicated by a set of side constraints. After the transformation, we can decompose the mathematical model into several independent sub-problems which can be solved optimally by any well-known algorithm or methodology. Figure 3-1 illustrates the major concepts of the Lagrangean relaxation method.

As mentioned above, we can use Lagrangean relaxation method to get some hints about the boundary of the objective function value. For the minimization problems, the optimal value, Z_D (μ), of the (LR_μ) is always a lower bound (LB) of the original problem. In order to get the tighter LB to close the optimal value of (P), we can repeatedly tune the multipliers to make Z_D (μ) as large as possible, which is so-called “Lagrangean dual problem.” During this process, we can obtain values of the decision variable and multipliers, which can help us to design appropriate heuristics to tune the infeasible solution to a feasible one. This step is called “getting primal feasible solution,” and each feasible solution we found is the upper bound (UB) of the original problem. Therefore, the optimal value of (P) is guaranteed to be between the LB and UB. There will be an area between the LB and the UB, call “gap”. In order to make the gap tightest, we have to design the best algorithm for each sub-problem and best heuristic for original problem.

In order to tune the multipliers, the subgrandient method is often to be used. Initially, the scalar which can modulate the step size of tuning multipliers in iteration is relatively bigger, so the vibration of multipliers is bigger. However, it will reduce in later period and at last tend to be stable and converges to one value, and thus Lagrangean relaxation method will be stopped. Figure3-2 shows the detail process.

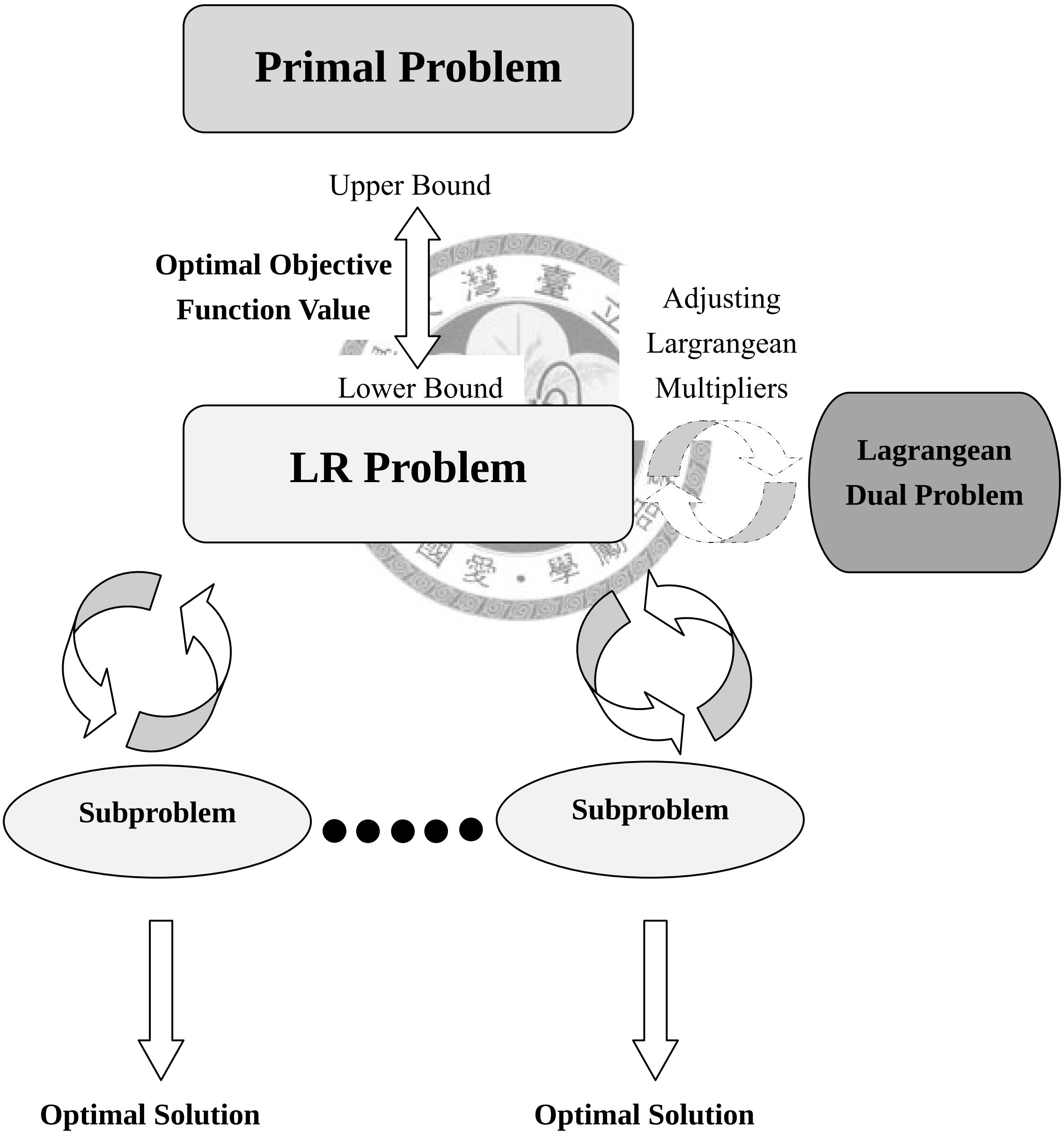
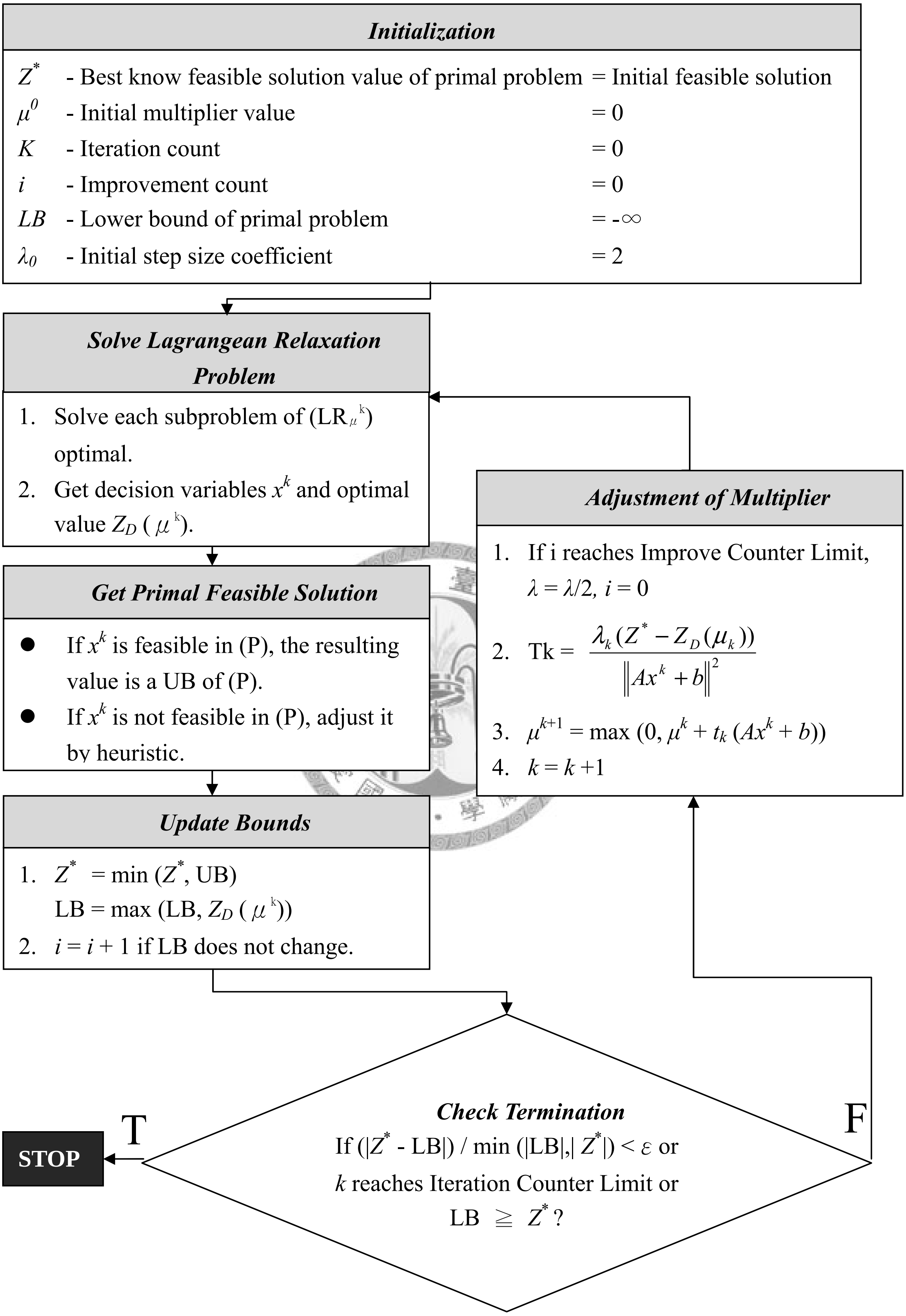


Figure 3-1 The Major Concepts of Lagrangean Relaxation Method


Initialization
Z^* - Best know feasible solution value of primal problem = Initial feasible solution
μ^0 - Initial multiplier value = 0
K - Iteration count = 0
i - Improvement count = 0
LB - Lower bound of primal problem = -∞
λ_0 - Initial step size coefficient = 2
Solve Lagrangean Relaxation Problem
1. Solve each subproblem of (LR_{μ}^{k}) optimal.
2. Get decision variables x^k and optimal value Z_D (μ^k).
Get Primal Feasible Solution
● If x^k is feasible in (P), the resulting value is a UB of (P).
● If x^k is not feasible in (P), adjust it by heuristic.
Update Bounds
1. Z^* = min (Z^*, UB)
LB = max (LB, Z_D (μ^k))
2. $i = i + 1$ if LB does not change.
Adjustment of Multiplier
1. If i reaches Improve Counter Limit, $\lambda = \lambda/2$, $i = 0$
2. Tk = $\dfrac{\lambda_k(Z^* - Z_D(\mu_k))}{\left\| Ax^k + b \right\|^2}$
3. μ^{k+1} = max (0, $\mu^k + t_k (Ax^k + b)$)
4. $k = k + 1$
Check Termination
If (|Z^* - LB|) / min (|LB|,| Z^*|) < ε or
k reaches Iteration Counter Limit or
LB ≧ Z^* ?
T
F
STOP

3-2 Flow Char for Lagrangean Relaxation Method

3.1.2 First-Stage Lagrangean Relaxation

In order to obtain tighter UBs and LBs, we adopt two-stage Largrangean relaxation procedure. In the first stage, we relax four constraints of (IP 2), and the details are described in 3.2.1.1.

3.1.2.1 Lagrangean Relaxation

As applying Lagrangean relaxation method to solve the problem (IP 2) of the APS model, we relax Constraints (IP 2.1), (IP 2.3), (IP 2.8), (IP 2.9) and replace them to objective function with the associated Lagrangean multipliers, μ_i^1, μ_i^2, μ_i^3, μ_g^4, respectively. Therefore, the original problem (IP 2) of the APS model is transformed into the following Lagrangean relaxation problem (LR 1).

Optimization problem：

$Z_D\ (\mu_1, \mu_2, \mu_3, \mu_4)$ (LR 1)

$$= \min - \sum_{g\in G} S_g Z_g + \sum_{i\in N}\mu_i^1[\sum_{p\in P_{(o,i)}} x_p - y_i] + \sum_{i\in N}\mu_i^2[\sum_{w\in W}\sum_{p\in P_w} x_p \varsigma_{pi} - (|N-1|)y_i] +$$

$$\sum_{i\in N}\mu_i^3[\hat{a}_i(b^c{}_i)y_i - a_i] + \sum_{g\in G} u_g^4[k_g Z_g - \sum_{i\in N}\sigma_{gi} y_i]$$

Subject to:

$\sum_{p\in P_w} x_p \le 1$ $\quad \forall\ w\in W$ (LR 1.1)

$x_p = 0\ or\ 1$ $\quad \forall\ p\in P_w, w\in W$ (LR 1.2)

$y_i = 0\ or\ 1$ $\quad \forall\, i \in N$ (LR 1.3)

$0 \le a_i \le \hat{a}_i(b^c{}_i)$ $\quad \forall\, i \in N$ (LR 1.4)

$\sum_{i \in N} a_i \le A$ (LR 1.5)

$Z_g = 0$ or 1 $\quad \forall g \in G.$ (LR 1.6)

Constraints (IP 2.2), (IP 2.4) ~ (IP 2.7), and (IP 2.10) of the objective function (IP2) of the APS model are not relaxed, but denoted as (LR 1.1) ~ (LR 1.6) in the LR problem. The Lagrangean multipliers μ_1, μ_2, μ_3, and μ_4 are the vectors of $\{\mu_i^1\}$, $\{\mu_i^2\}$, $\{\mu_i^3\}$, and $\{\mu_g^4\}$, in which μ_2, μ_3, and μ_4 are non-negative and the variable μ_1 is unrestricted. In order to solve (LR 1), we decompose it into four independent sub-problems, which can be respectively solved optimally as showing below.

Subproblem 1.1 (related to decision variable x_p)

$Z_{\text{Sub1}}(\mu_1, \mu_2) = \min \sum_{i \in N} \sum_{p \in P_{(o,i)}} \mu_i^1 x_p + \sum_{i \in N} \sum_{w \in W} \sum_{p \in p_w} \mu_i^2 \varsigma_{pi} x_p$ (Sub 1.1)

Subject to：

$\sum_{p \in P_w} x_p \le 1$ $\quad \forall\, w \in W$ (Sub 1.1.1)

$x_p = 0\ or\ 1$ $\quad \forall p \in P_w, w \in W.$ (Sub 1.1.2)

In this problem, we want to determine the value of x_p individually for each O-D pair. Note that Constraint (Sub 1.1.1) allows only one path to be chosen for an O-D pair. As described in the notations, each O-D pair w originates from an attacker's position o and ends at one target node i, where $i \in N$. Thus, $\sum_{i \in N} \sum_{p \in P_{(o,i)}} \mu_i^1 x_p$ can be transformed into $\sum_{w \in W} \sum_{p \in P_w} \mu_i^1 x_p$ As described in the notations, each O-D pair w originates from an attacker's position o and ends at one target node i, where $i \in N$. Thus, we can further decompose (Sub 1) into $|W|$ independent sub-problems. For each O-D pair $w = (o, i)$, $i \in N$ and $w \in W$,

$$Z_{\text{sub1'}} (\mu_1, \mu_2) = \min \sum_{p \in P_w} (\mu_i^1 + \sum_{j \in N} \mu_j^2 \varsigma_{pj}) x_p \qquad \textbf{(Sub 1.1')}$$

Subject to:

$$\sum_{p \in P_w} x_p \le 1 \qquad \forall\, w \in W \qquad \textbf{(Sub 1.1.1')}$$

$$x_p = 0 \; or \; 1 \qquad \forall\, p \in P_w,\, w \in W. \qquad \textbf{(Sub 1.1.2')}$$

In order to solve (Sub 1.1'), we design an algorithm as Table 3-1 shows.

Table 3-1 Algorithm to Solve (Sub 1.1')

Step 1: For each O-D pair $w \in W$, we find the minimum cost shortest path using μ_j^2 as the node weight by Dijkstra's minimum cost shortest path algorithm. The total cost of a path is the sum of the weights of the nodes on that path.

Step 2: For each O-D pair $w \in W$, we set the x_p value of each path p to zero except for the one already chosen to be the minimum cost shortest path for some O-D pair w, since not more than one path can exist between them.

Step 3: For each O-D pair $w \in W$, we examine the sum of its minimum path cost and the μ_i^1 value of its destination node. If the resulting value is non-positive, the x_p value of the minimum cost shortest path p between the O-D pair is set to one, because this is a minimization problem. The value of x_p is set to zero if its associated parameter is positive.

The time complexity of Dijkstra's algorithm is $O(|N|^2)$. Since the source of each path is the same, Dijkstra's algorithm only needs to be implemented once since its outcome is the minimum cost shortest path tree; thus, the total time complexity of (Sub 1.1) is $O(|N|^2)$.

Subproblem 1.2 (related to decision variable Z_g)

$$Z_{\text{Sub2}}(\mu_4) = \min \sum_{g \in G} [-S_g + \mu_g^{\ 4} k_g] Z_g \qquad \text{(Sub 1.2)}$$

Subject to：

$$Z_g = 0 \text{ or } 1 \qquad \forall g \in G. \qquad \text{(Sub 1.2.1)}$$

(Sub 1.2) can be further decomposed into $|G|$ independent sub-problems, for which we must decide the Z_g value of different information $g \in G$. Since (Sub 1.2) is a minimization problem, and the value of each Z_g is either zero or one, we can solve the problem by examining the associated parameters of Z_g easily and optimally. For each information $g\in G$, if ($-S_g+\mu_g^4k_g$) is positive, the value of Z_g is set to zero. On the other hand, if the sum of the parameters is non-positive, Z_g is set to one. Therefore, the value of this sub-problem can be minimized. To sum up, the rule to decide the value of Z_g is shown below.

$$Z_g = \begin{cases} 1, \text{ if } (-S_g+\ \mu_g^4k_g) < 0 \\ 0, \text{ if } (-S_g+\ \mu_g^4k_g) \geq 0 \end{cases}$$

The time complexity of (Sub 1.2) is O ($|G|$).

Subproblem 1.3 (related to decision variable a_i)

$$Z_{\text{Sub3}}\ (\mu_3) = \min \sum_{i\in N}(-\mu_i^3)a_i \qquad \text{(Sub 1.3)}$$

Subject to：

$$0 \leq a_i \leq \hat{a}_i(b^c{}_i) \qquad \forall\, i\in N \qquad \text{(Sub 1.3.1)}$$

$$\sum_{i\in N} a_i \leq A. \qquad \text{(Sub 1.3.2)}$$

By its nature, (Sub 1.3) is a fractional knapsack problem, in which the original maximized positive profit is replaced by minimized negative loss. In order to solve (Sub 1.3) optimally, we propose an algorithm which is shown in Table 3-2.

Table 3-2 Algorithm to Solve (Sub 1.3)

Step 1: We sort each node $i \in N$ by the parameter of each a_i and a_i itself in ascending order with ($-\mu_i^3$) as the primary key. Because of the non-negativity of μ_i^3, the parameter of each a_i will be non-positive.

Step 2: We check the array of sorted nodes from the left, and set the value of each a_i to $\hat{a}_i(b^c{}_i)$.

Step 3: We stop once the sum of a_i reaches A, or there is insufficient space to set the next a_i to to $\hat{a}_i(b^c{}_i)$. In such a case, the next a_i is set to (A − the summation of a_i that have already been given a value), and the remainder are set to zero.

The time complexity of (Sub 3) is $O(|N|^2)$.

Subproblem 1.4 (related to decision variable y_i)

Z_{Sub4} (μ_1, μ_2, μ_3, μ_4) (Sub 1.4)

$$= \min \sum_{i \in N} (-\mu_i{}^1 - \mu_i{}^2 |N-1| + \mu_i{}^3 \hat{a}_i(b^c{}_i) - \sum_{g \in G} \mu_g{}^4 \sigma_{gi}) y_i$$

Subject to:

$$y_i = 0 \; or \; 1 \qquad i \in N. \qquad \text{(Sub 1.4.1)}$$

(Sub 1.4) can be further decomposed into |N| independent sub-problems, for which we must decide the y_i value of different information $i \in N$ Since (Sub 1.4) is a minimization problem, and the value of each y_i is either zero or one, we can solve the problem by examining the associated parameters of y_i easily and optimally. For each information $i \in N$, if $(-\mu_i{}^1 - \mu_i{}^2 |N| + \mu_i{}^3 \hat{a}_i(b^c{}_i) - \sum_{g \in G} \mu_g{}^4 \sigma_{gi})$ is positive, the value of y_i

is set to zero so that the value of this sub-problem can be minimized. On the other hand, if the sum of the parameters is non-positive, y_i is set to one. To sum up, the rule to decide the value of y_i is shown below.

$$y_i = \begin{cases} 1, \text{ if } \ (-\mu_i^1 - \mu_i^2 |N-1| + \mu_i^3 \hat{a}_i (b^c{}_i) - \sum_{g \in G} \mu_g{}^4 \sigma_{gi}) < 0 \\ \\ 0, \text{ if } \ (-\mu_i^1 - \mu_i^2 |N-1| + \mu_i^3 \hat{a}_i (b^c{}_i) - \sum_{g \in G} \mu_g{}^4 \sigma_{gi}) \geq 0 \end{cases}$$

The time complexity of (Sub 1.4) is O (|N|).

3.1.2.2 The Dual Problem and the Subgradient Method

According to the weak duality theorem of Lagrangean relaxation method [16], for any set of the multipliers (μ_1, μ_2, μ_3, μ_4) is a lower bound on $Z_{\text{IP 2}}$. Therefore, we construct a dual problem (D 1) to calculate the tightest LB and solve it by the subgradient method [14][15].

Dual Problem (D 1),

$Z_D = \max Z_D (\mu_1, \mu_2, \mu_3, \mu_4)$ (D 1)

Subject to:. $\mu_2 \geq 0, \mu_3 \geq 0, \mu_4 \geq 0$.

Let a vector m be a subgradient of Z_D (μ_1, μ_2, μ_3, μ_4) . Then, in iteration k of the subgradient procedure, the multiplier vector $\mu^k = (\mu_1^{k,} \mu_2^k, \mu_3^k, \mu_4^k)$ is updated by

$$\mu^{k+1} = \mu^k + t^k m^k,$$

where

$m^k (\mu_1^{k,} \mu_2^k, \mu_3^k, \mu_4^k) =$

$$(\sum_{p\in P_w} x_p - y_i, \sum_{w\in W}\sum_{p\in P_w} x_p \varsigma_{pi} - (|N|-1)y_i, \hat{a}_i(b^c{}_i)y_i - a_i, k_g Z_g - \sum_{i\in N}\sigma_{gi} y_i\ );$$

and the step size, t^k, is determined by

$$t^k = \lambda \frac{Z^*_{\text{IP 2}} - Z_{\text{D}}(v^k)}{\left\| m^k \right\|^2}.$$

$Z^*_{\text{IP 2}}$ represents the best UB on the primal objective function value found by iteration k. and λ is a scalar between 0 and 2. it is usually initiated with the value of 2 and halved if the best objective function value does not improve within a given iteration count.

3.1.2.3 Getting Primal Feasible Solutions

During first stage lagrangean relaxation, LB rises again and again. However, we also have to lower UB by improving our heuristic to the APS model. Thus, we use the multipliers and the result of solving subproblem in (LR1) to help us design the algorithm for solving the APS model. The algorithm, denoted as Heuristic_LR_1, is described below.

Step 1. For each path p, if it is selected to attack in (Sub 1.1), add each node on *p* to the attack tree.

Step 2. Calculate the total attack cost on the attack tree. If it is less than total attack budget, go to step 3; otherwise go to step 7.

Step 3. Initiate *victim_candidate_set*.

Step 4. Select node *i* whose defense capacity is no more than the difference between total attack budget and total attack cost from *victim_candidate_set* by the node's weight. If we can not find any one of which, terminate.

Step 5. Compromise node *i* and add it to the attack tree.

Step 6. Update the total attack cost, *victim_candidate_set*, and node's weight. Go to Step 4.

Step 7. Recursively remove leaf nodes and update each node's weight until the total attack cost is no more than total attack budget.

In the heuristic, we use the result of solving (Sub 1.1) to derive the final solution. Initially, we add the nodes on paths which are selected in (Sub 1.1) to the attack tree. If the total attack cost of the attack tree is less than the total attack budget, the tree can be expanded further; otherwise we have to remove some nodes from the attack tree.

In the first case, the problem is which node is better for the attacker to choose in the current run. To answer this question, we design a dynamic weight of each node *i*, which is $\mathbf{((}\hat{a}_i(b^c{}_i) + |\mathrm{N}| \mathbf{*} v^2{}_i\mathbf{)} \mathbf{/} \mathbf{(}nodeDamage^2 + nodeDamage \mathbf{/} a_i\mathbf{))}$. Note that only the unrecovered information will be calculated into $nodeDamage_i$. This formula implies that the node which has the best cost-benefit ratio will be the target. Also, it drives the attacker to compromise nodes which have pieces of to-be-recovered information.

In the second case, we have to recursively remove leaf nodes until the total attack cost is no more than the total attack budget. Similarly, the node's dynamic weight is used to help the attacker make decision. Note that the damage of recovered information, whose number of stolen pieces is equal to its threshold, will be enhanced. It drives the attacker not to remove nodes which contain pieces of weak information, which is the information whose number of stolen pieces is equal to its threshold.

The time complexity of calculating each node's weight is $O(|N|*|G|)$. The times of calculating each node's weight are $|N|$. Thus the time complexity of the algorithm is $\mathrm{O}(|N|^2*|G|)$.

Table 3-4 describes the detail of the heuristic.

Table 3-3 Heuristic_LR_1 Algorithm

```
//Initialization
FOR each attack_path p {
    IF ($x_p$ in Subproblem 1.1 is set to one) {
        Add each node i on p to the attack_tree;
    }
}
FOR each information g {
    g.currentDiffToThreshold = $k_g$;
}
FOR each node i which is on attack_tree {
    FOR each information g {
        IF ( node i contains g's piece ) {
            g.currentDiffToThreshold--;
        }
    }
}
Calculate total_attack_cost of the attack_tree;
Initialize victim_candidate_set;
IF (total_attack_cost < TOTAL_ATTACK_BUDGET) {
    WHILE (total_attack_cost < TOTAL_ATTACK_BUDGET AND there are still
      uncompromised nodes) {
        FOR each node i which is in victim_candidate_set {
            FOR each information g whose currentDiffToThreshold > 0{
                i.nodeDamage += $S_g$ / currentDiffToThreshold;
            }
            IF (nodeDamage == 0)
                $weight = (\hat{a}_i(b^c{}_i) + |N| * v^2{}_i);$
            ELSE
                $weight = (\hat{a}_i(b^c{}_i) + |N| * v^2{}_i)/(nodeDamage^2 + nodeDamage / a_i);$
        }
        Find node i, whose weight is the smallest among all other nodes' weight in
          victim_candidate_set AND whose defense_capability is no more than
        (TOTAL_ATTACK_BUDGET – total_attack_cost);
        Compromise node i and add it to the attack_tree;
        FOR each information g {
                IF (node i contains g’s piece)
                    g.currentDiffToThreshold--;
            }
            total_attack_cost += defense_capability of node i;
            Update victim_candidate_set;
        }
}
ELSE {
        WHILE (total_attack_cost > TOTAL_ATTACK_BUDGET) {
            FOR each leaf_node i {
                FOR each information g {
                    IF (currentDiffToThreshold = = 0)
                        i.nodeDamage += $S_g^2$;
                    ELSE
                        i.nodeDamage += $S_g$;
                }
                IF (nodeDamage == 0)
                    $weight = (\hat{a}_i(b^c{}_i) + |N| * v^2{}_i);$
                ELSE
                    $weight = (\hat{a}_i(b^c{}_i) + |N| * v^2{}_i)/(nodeDamage^2 + nodeDamage / a_i);$
            }
            Find node i, which is a leaf_node of the attack_tree AND whose weight is
              the largest among all leaf_nodes;
            Remove node i from the attack_tree;
            total_attack_cost -= defense_capability of node i;
            FOR each information g {
                IF (node i contains g’s piece)
                    g.currentDiffToThreshold++;
            }
        }
}
```

3.1.3 Second-Stage Relaxation

After finishing the first stage relaxation procedure, we obtain an UB and a legitimate LB. In order to narrow the gap between UB and LB, we take the second stage relaxation. In this stage, the initial UB and the initial LB are the best UB and the best LB of the first-stage relaxation respectively. The details are described in 3.2.2.1.

3.1.3.1 Lagrangean Relaxation

As applying Lagrangean relaxation method to solve the problem (IP 2) of the APS model, we relax Constraints (IP 2.1), (IP 2.3), (IP 2.7), (IP 2.9) and replace them to objective function with the associated Lagrangean multipliers, v_i^1, v_i^2, v^3, v_g^4, respectively. Therefore, the original problem (IP 2) of the APS model is transformed into the following Lagrangean relaxation problem (LR 2).

Z_D (v_1, v_2, v_3, v_4)

$$= \min - \sum_{g\in G} S_g Z_g + \sum_{i\in N} v_i^{1}[\sum_{p\in P_{(o,i)}} x_p - y_i] + \sum_{i\in N} v_i^{2}[\sum_{w\in W}\sum_{p\in P_w} x_p \varsigma_{pi} - (|N-1|)y_i] +$$

$$v^3[\sum_{i\in N} a_i - A] + \sum_{g\in G} v_g^{4}[k_g Z_g - \sum_{i\in N} \sigma_{gi} y_i] \qquad \text{(LR 2)}$$

Subject to:

$$\sum_{p\in P_w} x_p \le 1 \qquad \forall\ w \in W \qquad \text{(LR 2.1)}$$

$$x_p = 0 \; or \; 1 \qquad \forall\, p \in P_w,\, w \in W \qquad \text{(LR 2.2)}$$

$$y_i = 0 \; or \; 1 \qquad \forall\, i \in N \qquad \text{(LR 2.3)}$$

$$0 \le a_i \le \hat{a}_i(b^c{}_i) \qquad \forall\, i \in N \qquad \text{(LR 2.4)}$$

$$\hat{a}_i(b^c{}_i)y_i \le a_i \qquad \forall\, i \in N \qquad \text{(LR 2.5)}$$

$$Z_g = 0 \text{ or } 1 \qquad \forall g \in G. \qquad \text{(LR 2.6)}$$

The Lagrangean multipliers v_1, v_2, and v_4 are the vectors of $\{v_i^1\}$, $\{v_i^2\}$, and $\{v_g^4\}$, in which v_2 and v_4 are non-negative and the variable v_1 is unrestricted. The Lagrangean multipliers v_3 is non-negative. In order to solve (LR 2), we decompose it into three independent sub-problems, which can be respectively solved optimally as showing below.

Subproblem 2.1 (related to decision variable x_p)

$$Z_{\text{Sub1}}\,(\nu_1, \nu_2) = \min \sum_{i\in N}\sum_{p\in P_{(o,i)}} \nu_i{}^1 x_p + \sum_{i\in N}\sum_{w\in W}\sum_{p\in p_w} \nu_i{}^2 \varsigma_{pi} x_p \qquad \text{(Sub 2.1)}$$

Subject to：

$$\sum_{p\in P_w} x_p \le 1 \qquad \forall\, w \in W \qquad \text{(Sub 2.1.1)}$$

$$x_p = 0 \; or \; 1 \qquad \forall p \in P_w, w \in W. \qquad \text{(Sub 2.1.2)}$$

This sub-problem is the same as (Sub 1.1) in the first stage relaxation, so we can adopt the algorithm proposed in Section 3.1.2.1 to solve (Sub 2.1).

The time complexity of (Sub 2.1) is O(|N|^2).

Subproblem 2.2 (related to decision variable Z_g)

$$Z_{\mathrm{Sub2}}\ (\nu_4) = \min \sum_{g\in G}[-S_g + \nu_g{}^4 k_g]Z_g \qquad \text{(Sub 2.2)}$$

Subject to：

$$Z_g = 0 \text{ or } 1 \qquad \forall g \in G. \qquad \text{(Sub 2.2.1)}$$

The problem is the same as (Sub 1.2) in the first relaxation, so we can adopt the rule in Subproblem 1.2 to decide the value of Z_g.

Subproblem 2.3 (related to decision variable a_i, y_i)

$$Z_{\mathrm{Sub4}}\ (\nu_1, \nu_2, \nu_3, \nu_4) \qquad \text{(Sub 2.3)}$$

$$= \min \sum_{i\in N}(-\nu_i{}^1 - \nu_i{}^2|N-1| - \sum_{g\in G}\nu_g{}^4\sigma_{gi})y_i + \nu^3\sum_{i\in N}a_i$$

Subject to:

$$y_i = 0\ or\ 1 \qquad \forall i \in N. \qquad \text{(Sub 2.3.1)}$$

$$0 \le a_i \le \hat{a}_i(b^c{}_i) \qquad \forall\, i \in N \qquad \text{(Sub 2.3.2)}$$

$$\hat{a}_i(b^c{}_i)y_i \le a_i \qquad \forall\, i \in N \qquad \text{(Sub 2.3.3)}$$

This problem contains two decision variables (y_i and a_i) and they are bound by constraint (Sub 2.3.3). We can design a rule to decide their values according to (Sub

2.3.2) and (Sub 2.3.3). First of all, we decide the value of y_i depending on examining the associated parameters of y_i. Since this is a minimization problem, and the value of each y_i is either zero or one, so the rule to decide the value of each y_i is shown as below.

$$y_i = \begin{cases} 1, \text{ if } (-v_i^1 - v_i^2\,|N\text{-}1| - \sum v_g{}^4 \sigma_{g_i}) < 0 \\ 0, \text{ if } (-v_i^1 - v_i^2\,|N\text{-}1| - \sum v_g{}^4 \sigma_{g_i}) \geq 0 \end{cases}$$

After deciding the value of y_i, we can determine the value of a_i. Since v_3 is non-negative, and we want to minimize (Sub 2.3), the rule to determine a_i is as below.

$$a_i = \begin{cases} \hat{a}_i(b^c{}_i), \text{ if } (y_i = 1 \text{ and } -v_i^1 - v_i^2\,|N\text{-}1| - \sum v_g{}^4 \sigma_{g_i} + v^3 \hat{a}_i(b^c{}_i) < 0) \\ 0, \quad \text{if } (y_i = 0 \text{ or } (y_i = 1 \text{ and } - v_i^1 - v_i^2\,|N\text{-}1| - \sum v_g{}^4 \sigma_{g_i} + v^3 \hat{a}_i(b^c{}_i) \geq 0)) \end{cases}$$

The time complexity of (Sub 2.3) is O(|N|).

3.1.3.2 The Dual Problem and the Subgradient Method

According to the weak duality theorem of Lagrangean relaxation method [16], for any set of the multipliers (v_1, v_2, v_3, v_4) is a lower bound on $Z_{\text{IP }2}$. Therefore, we construct a dual problem (D 2) to calculate the tightest LB and solve it by the subgradient method [14][15].

Dual Problem (D 1),

$$Z_D = \max Z_D\,(v_1, v_2, v_3, v_4) \tag{D 1}$$

Subject to:. $v_2 \geq 0, v_3 \geq 0, v_4 \geq 0.$

Let a vector *m* be a subgradient of Z$_D$ (v_1, v_2, v_3, v_4). Then, in iteration *k* of the subgradient procedure, the multiplier vector $v^k = (v_1^{k,} v_2^k, v_3^k, v_4^k)$ is updated by

$$v^{k+1} = v^k + t^k m^k,$$

where

$m^k\,(v_1^{k,}\ v_2^k, v_3^k, v_4^k) =$

$$(\sum_{p\in P_w} x_p - y_i, \sum_{w\in W}\sum_{p\in P_w} x_p \varsigma_{pi} - (|N|-1)y_i, \sum_{i\in N} a_i - A, k_g Z_g - \sum_{i\in N} \sigma_{gi} y_i);$$

and the step size, t^k, is determined by

$$t^k = \lambda \frac{Z^*_{\mathrm{IP}\,2} - Z_{\mathrm{D}}(v^k)}{\left\| m^k \right\|^2}.$$

$Z^*_{\mathrm{IP}\,2}$ represents the best UB on the primal objective function value found by iteration *k*. and λ is a scalar between 0 and 2. It is usually initiated with the value of 2 and halved if the best objective function value does not improve within a given iteration count.

3.1.3.3 Getting Primal Feasible Solutions

In order to lower the UB further, we design another heuristic, denoted as Heuristic_LR_2, which is described as below.

Since the attacker wants to construct an attack tree to maximize damage under limited attack budget, he/she must try to get the number of information's pieces which exactly equals to threshold to recover information as possible as he/she can. In order to carry out this objective we mentioned above, we take the following steps:

Step 1. Construct a minimum cost spanning tree.

Step 2. Select information by its weight to recover and mark it

Step 3. Recursively remove unmarked leaf nodes which do not contain pieces belonging to the information selected in Step 2.

Step 4. If the attack tree contains exactly the number of selected information's pieces which equals to threshold or each leaf node is marked, go to Step 5; otherwise remove unmarked leaf nodes by its weight and go to Step 3.

Step 5. If the cost of attack tree is no more than total attack budget, mark each node on the attack tree and set selected information obtainable.

Step 6. Calculate the number of other unmarked information's pieces which are on the current attack tree. If there is information whose number of pieces on the current attack tree equals or more than the threshold to recover it, mark it and set it obtainable.

Step 7. If there is unmarked information, go to Step 1.

Step 8. Compromise marked nodes and add them to the attack tree, and recover obtainable information.

In Step 2, we have to select information each by each, and the question is how to determine information's priority. Thus, we assign each information *g* a dynamic weight, ($\frac{S_g}{currentDiffToThreshold_g}+\frac{n_g * Z_g}{k_g}$). Z_g is the solution obtained from (LR 2); $currentDiffToThreshold_g$ is the difference between the number of pieces you have gotten and the threshold of number of pieces to recover information *g*. This formula implies that information which has the best cost-benefit ratio in current run would be the target to steal. In addition, the ratio of total number of information's pieces to the threshold to recover information is considered. That means the more pieces information is divided into and the less pieces information needs to recover will be good for information's weight. Moreover, the formula also considers the hints obtained from the result of solving (LR 2). If information *g* is selected in (LR 2), the weight of which will be improved.

In Step 4, we have to remove leaf node with piece of selected information.

Thus, we design a rule to decide which one would be removed. Similar to the above, the design of dynamic node's weight is concerned, which is $(\frac{pathCost_i + |N| * \mu^2{}_i}{pathDamage_i + nodeDamage_i / a_i})$ and $nodeDamage_i$ / a_i is set to zero if a_i is equal to zero. $pathCost_i$ represents the cost that the attack compromise node i have to consume on path; $nodeDamage_i$ is the sum of *pieceDamage* on node i, where *pieceDamage* is equal to S_g / $currentDiffToThreshold_g$; $pathDamage_i$ is the sum of *nodeDamage* on path. Note that weight of node i is set to ($pathCost_i$ + $|N|*\mu^2{}_i$) if $pathDamage_i$ is equal to zero. This formula implies that node i whose cost-benefit ratio on its attack path is the worst would be remove. Moreover, the formula considers the hints obtained from the result of solving (LR 2). If the attacker applies non-zero attack budget to a node in (LR 2), the node is tending to be a target. $|N|*u^2{}_i$ reflects the penalty of inconsistency between x_p and y_i, which means node i is selected to be a target but there is no attack path to it.

In Step 1, we can duplicate this spanning tree and use it repeatedly, so we only have to create it one time. The total time complexity of Prim's algorithm to construct a minimum cost spanning tree is $O(|N|\log|N|)$. The complexity of Checking all information is $O(|G|)$. The complexity of calculating each node's weight is $O(|N|*|G|)$. Thus, the complexity of the algorithm is $O(|G|^2*|N|)$.

Table 3-3 describes the detail of this heuristic.

Table 3-4 Heuristic_LR_2 Algorithm

```
//Initialization
FOR each information g {
    Set g to unmarked;
    Set g to unobtainable;
    currentDiffToThreshold = $k_g$; //set initial threshold
    containPieceNum = $n_g$; //set initial number of pieces
}

// check all information
WHILE (there exists unmark information) {
    Prim(); //construct the minimum cost spanning tree rooted at o
    FOR each information g which is unmarked {
        weight = pieceDamage = $S_g$ / currentDiffToThreshold;
        IF ($Z_g$ in Subproblem 2.2 is set to one)
            weight += $n_g / k_g$;
    }
    FOR each node i {
        FOR each information g which is unmarked {
            IF (i has g's piece) {
                nodeDamage += pieceDamage;
            }
        }
    }
    FOR each node i {
        pathDamage = summation of nodeDamage of all nodes which is
            unmarked on i's path;
        pathCost = summation of defense capability of all nodes which is unmarked
            on i's path;
        weight = (pathCost + $|N|*u^2_i$) / (pathDamage + nodeDamage/$a_i$);
    }
    Find information g, which is unmarked AND whose weight is the largest among
        all unmarked information;
    Set g to marked;
    Recursively remove unmarked leaf_nodes without g's pieces from the
        attack_tree until all leaf_nodes of the attack_tree contain g's pieces;

    WHILE (g's containPieceNum > g's currentDiffToThreshold) {
            Find node i, which is an unmarked leaf_node of the attack_tree AND
              whose weight is the largest among all leaf_nodes;
            Remove i from attack_tree;
            g.containPieceNum--;
            Recursively remove unmarked leaf_nodes without g's pieces from the
              attack_tree until all leaf_nodes of the attack_tree contain g's pieces;
    }
    Calculate the total_attack_cost of the attack_tree;
    IF (total_attack_cost $\le$ TOTAL_ATTACK_BUDGET) {
        FOR each information g {
            currentDiffToThreshold = $k_g$; //set initial threshold
            containPieceNum = $n_g$; //set initial number of pieces
        }
        FOR each node i which is on attack_tree {
            Set node i to marked;
            FOR each information g {
                IF (i has g's piece) {
                    currentDiffToThreshold--;
                    containPieceNum--;
                }
            }
        }
        FOR each information g {
            IF (currentDiffToThreshold $\le$ 0) {
                Set g to marked;
                Set g to obtainable;
            }
        }
    }
}
Compromise each marked node and add them to the attack tree;
```

3.1.4 Summary of the Solution Approach for APS Model

3.1.4.1 Lagrangean Relaxation-based Algorithm

We solve the APS model by adopting Lagrangean relaxation-based algorithms we propose, and denote it as LR. First of all, we relax some complex constrains and put them into objective function with respective multipliers. Secondly, we decompose the complex mathematical model into several stand-along subproblems, and optimally solve them to get a LB for the primal problem. Lastly, two heuristic are adopted to derive the feasible solutions to the primal problem, which is a UB for the primal problem. In addition, we tighten the gap between the LB and the UB by solving dual problem. The LR procedure is repeated until the stop condition is satisfied. Table 3-5 describes the detail of LR algorithm.

Table 3-5 LR Algorithm

```
Initialize the Lagrangean multiplier vectors ($\mu_1$, $\mu_2$, $\mu_3$, $\mu_4$) and ($v_1$, $v_2$, $v_3$, $v_4$) to be zero
   vectors;
UB = 0; LB = -SUM_OF_VALUE_OF_ALL_INFORMATION;
improvement_counter = 0
$\lambda$ = 2; //step size coefficient
Initiate_Budget_Allocation_Strategy();

//Main LR procedure
FOR iteration = 1 TO ITERATION_COUNTER_LIMIT {
    IF (iteration $\leq$ (ITERATION_COUNTER_LIMIT / 2)) {
        Solve (Sub 1.1);
        Solve (Sub 1.2);
        Solve (Sub 1.3)
        Solve (Sub 1.4);
        $Z^*_{IP\,2}$ = –Heuristic_LR_1();
    }
    ELSE {
        Solve (Sub 2.1);
        Solve (Sub 2.2);
        Solve (Sub 2.3);
        $Z^*_{IP\,2}$ = –Heuristic_LR_2();
    }
    Calculate $Z_D$;

    //Update bounds
    IF ($Z_D > LB$) {
        $LB = Z_D$;
        improvement_counter = 0;
    }
    ELSE {
        improvement_counter ++;
    }
    IF ($Z^*_{IP\,2} < UB$) {
        $UB = Z^*_{IP\,2}$;
    }

    //Update step size and Lagrangean multipliers
    IF (improvement_counter = IMPROVEMENT_COUNTER_LIMIT) {
        improvement_counter = 0;
        $\lambda = \lambda / 2$;
    }
    Update_Step_Size();
    Update_Lagrangean_Multiplier();
}
```

3.2 Solution Approach for the DRAID Model

After solving the APS model, we can obtain the best strategy for the attacker. In order to solve the DRAID model, the optimal solution of the APS model can be used as the input of it. We adjust defense budget allocation strategy and pieces allocation strategy according to the current attack strategy. After the adjustment, we solve the

APS model again and obtain another attack strategy which is aimed at the new defense strategy. The interaction between the attacker and the defender continues until it reaches a balance. Table 3-6 describes the detail of the adjustment procedure.

Since we have to design both budget allocation strategy and pieces allocation strategy, we take the two stages adjustment. First of all, we reallocate defense budget to nodes. The budget reallocation strategy which we adopt is based on the concept of the subgrandient method. If nodes are uncompromised, it means we allot too much budget to them; on the other hand, nodes are allotted insufficient budget if they are compromised. Thus, we take back parts of defense budget from uncompromised nodes and reallocate it to compromised nodes. The question is how to decide the proportion of defense budget which is taken back from uncompromised node. Obviously, if a node has more number of times used to be a hop-site, the more important it is. So an impact factor is used to normalize the number of times a node was used as a hop-site. The formula of the factor is (w_i / w_{max}), where w_i is the average frequency that node I has been used as a hop-site up to now, and w_{max} is the average number of nodes which are compromised by the attacker up to now. The higher the impact factor of a node, the lower the proportion of the defense budget will be taken from it if it is uncompromised in current run. How to reallocate the extra budget to compromised nodes is also important. Here we propose two heuristics for comparison.

One, denoted as damage_based reallocation algorithm, reallocates the extra budget according to number of pieces on compromised nodes. The more the number of pieces on a compromised node is, the more important it is. The other one, denoted as bonus_base reallocation algorithm, reallocates the extra budget according to the bonus that nodes bring when allocating additional budget to them. Table 3-7 describes the detail of the defense budget reallocation strategy.

If there is no improvement under defense budget reallocation procedure during a period, we take the pieces reallocation procedure. The concept of reallocating pieces is to make the attacker collect pieces more difficultly. For information, if it is recovered successfully by the attacker, that means the pieces allocation of it is easy for the attacker to collect these pieces. Thus, we reallocate some pieces to other nodes which are uncompromised and have few pieces. After reallocating pieces of recovered information, we verify if it satisfies QoS requirement of legitimate users and correct it. We do the adjustment and then solve APS model repeatedly until the improvement does not occur during a period, and switch to the defense budget reallocation procedure. The switch between two procedures will be terminated when maximal iteration is reached. The detail of pieces reallocation strategy is described in Table 3-8. The time complexity of pieces_reallocation algorithm is $O(|N|*|G|)$. The time complexity of defense budget reallocation is $O(|N|)$.

Table 3-6 Heuristic_DRAID Algorithm

```
//Initialization
UB = –LR(); //the return value of LR() is negative due to the objective function
   transformation in the AS model
improvement_counter = 0
reallocate_budget = true;

//Main Heuristic_DRAS procedure
FOR iteration = 1 TO ITERATION_COUNTER_LIMIT {
      IF (reallocate_ budget) {
            Reallocate_Defense_Budget();//as shown in Table 3-7
      }
      ELSE {
                  Reallocate_Pieces(); //as shown in Table 3-8
                  Initiate_Budget_Allocation_Strategy();
      }

      $Z^*_{\mathrm{IP}\ 1}$ = –LR();

      //Update UB
      IF ($Z^*_{\mathrm{IP}\ 1}$ < UB) {
            UB = $Z^*_{\mathrm{IP}\ 1}$;
            improvement_counter = 0;
      }
      ELSE {
            improvement_counter ++;
      }
      //Update step size
      IF improvement_counter = IMPROVEMENT_COUNTER_LIMIT {
            improvement_counter = 0;
            IF (reallocate_ budget) //change to pieces_ reallocation strategy
                  reallocate_ budget = false;
            ELSE //change to budget_ reallocation strategy
                  reallocate_ budget = true;
      }
}
```

Talbe 3-7 Reallocate_Defense_Budget Algorithm

```
//Initialization
total_defense_cost = 0;
FOR each node i {
    IF (node i is uncompromised) {
        $b_i = b_i * (1- w_i / w_{max})$
    }
    total_defense_cost += $b_i$;
}
remaining_defense_budget = TOTAL_DEFENSE_BUDGET – total_defense_cost

//Reallocation of defense budget
FOR each node i {
    IF (node i is compromised) {
        $b_i$ += remaining_defense_budget * Budget_Reallocation_Strategy();
          to compromised node according to reallocation strategy
    }
}
```

Table 3-8 Reallocate_Pieces Algorithm

```
//Initialization
FOR each compromised node i {
    FOR each information g which is recombined successfully by attacker {
        IF (node i contain g's piece)
            Take g's piece back from node i and put it into PieceBasket;
    }
}
FOR each information g whose pieces are in PieceBasket {
    Update each uncompromised node's nodeDamage;
    WHILE (there are g's pieces in PieceBasket AND there exists uncompromised
      nodes which do not contain g's piece and capable for g's piece) {
        Find uncompromised node i, which do not contain g's piece AND whose
          nodeDamage is smallest AND whose remaining capacity is capable for
          g's piece;
        Allocate one piece which belong to g to i;
    }
}
```

```
FOR each information g whose pieces are in PieceBasket {
    Update each compromised node’s nodeDamage;
    WHILE (there are g’s pieces in PieceBask ) {
      Find compromised node i, which do not contain g’s piece AND whose
        nodeDamage is smallest AND whose remaining capacity is capable for
        g’s piece;
    }
}
VerifyQoS();
Initial_budget_allocation();
```

Chapter 4 Computational Experiments

4.1 Computational Experiment with the APS Model

In order to show our heuristics' superiority, we design two simple algorithms for comparison purpose.

4.1.1 Simple Algorithm 1

Simple algorithm 1 is based on the concept of the greedy method, in which the node has the smallest weight will be selected as a target. However, the formula of node's weight is different from which we proposed in former heuristics. Table 4-1 shows the pseudo code of simple algorithm1, which is denoted as SA$_1$.

Table 4-1 SA$_1$ Algorithm

```
FOR each node i {
    FOR each information g {
        IF (node i contain g's piece)
            $nodeDamage_i$ += $S_g$ / $k_g$;
    }

    $weight = \frac{a_i(b^{c}{}_i)}{nodeDamage_i^{\,2}}$;

}

Add source o to attack_tree;

//Construction of attack_tree
WHILE (total_attack_cost < TOTAL_ATTACK_BUDGET AND there are still
  uncompromised nodes) {
    Find uncompromised node i, whose weight is the smallest;
    path_cost = summation of defense capability of all uncompromised nodes on i's
       path;
    IF (path_cost + total_attack_cost ≤ TOTAL_ATTACK_BUDGET) {
        Compromised nodes on i's path and add them to the attack_tree;
        total_attack_cost += path_cost of node i;
    }
}
```

4.1.2 Simple Algorithm 2

The concept of simple algorithm 2 is derived from the heuristic of second-stage Lagrangean relaxation shown in Section 3.1.3.3. The differences between them are the formula of the weight and the initial attack tree. The formula of the weight is $\frac{a_i(b^c{}_i)}{nodeDamage_i^2}$, and the node in *victim_candidate_set* has the smallest weight will be selected as a target. The initial attack tree only contains the attacker's initial position *o*, and then expends by degrees. Table 4-2 shows the pseudo code of simple algorithm2, which is denoted as SA_2.

Table 4-2 SA_2 Algorithm

```
WHILE (total_attack_cost < TOTAL_ATTACK_BUDGET AND there are still
   uncompromised nodes) {
    FOR each node i which is in victim_candidate_set {
        FOR each information g whose currentDiffToThreshold > 0 {
            i.nodeDamage += $S_g$ / currentDiffToThreshold;
        }
        IF (nodeDamage == 0)
            weight = ($\hat{a}_i(b^c{}_i)$);
        ELSE
```

```
            $weight = ( \hat{a}_i (b^c{}_i) )/(nodeDamage^2)$;
    }
    Find node $i$, whose weight is the smallest among all other nodes' weight in
      victim_candidate_set AND whose defense_capability is no more than
      (TOTAL_ATTACK_BUDGET – total_attack_cost);
    Compromise node $i$ and add it to the attack_tree;
    FOR each information $g$ {
        IF (node $i$ contains $g$'s piece)
            g.currentDiffToThreshold--;
    }
    total_attack_cost += defense_capability of node i;
    Update victim_candidate_set;
}
```

4.1.3 Experiment Environment

We transform the proposed algorithms for solving the APS model into codes in Visual C++ and execute them in a PC with an INTEL Pentium 4, 3GHz CPU. The Iteration Counter Limit and Improve Counter Limit are set to 1000 and 100 respectively. The first-stage relaxation procedure and the corresponding getting primal algorithm are executed in iteration 1~500; the second-stage relaxation procedure and the corresponding getting primal algorithm are executed in iteration 501~1000. We initiate the step size scalar, λ, as 2 and halve it if the improvement of the objective function value, Z_D, does not occur during a period of Improve Counter Limit.

We adopt three different types of network topology as attack targets. The first is a grid network, which is composed by $k \times k$ nodes; the second is a random network, in which nodes connect to nodes randomly and the average degree of each node is set to four; the third is a scale-free network, in which each newly added node connects to two different nodes which have the biggest degree in the network, and the scale-free network is most close to the network of the real world.

To choose a better strategy of total defense budget allocation, we design ten different ratios of the budget, B1, for decreasing random access error on nodes to the budget, B2, for enhancing nodes’ defense capability. In addition, each ratio is collocated with three budget allocation strategy of B2, which are uniform based, degree based, and damage based. We choose the best one as our budget allocation strategy.

We evaluate the effects of three different functions of defense capability on the budget allocation strategy. The first is a concave function; the second is a linear function; the third is a convex function.

The parameters and scenarios used in our experiments are described in Table 4-3

Table 4-3 Experiment Parameter Settings for the APS Model

Parameters of LR

Parameters	**Value**
Iteration Counter Limit	1000
Improve Counter Limit	100
Initial UB	0
Initial Multiplier Value	$\mu_1^{0} = \mu_2^{0} = \mu_3^{0} = \mu_4^{0} = 0$ $v_1^{501} = \mu_1^{500}$, $v_2^{501} = \mu_2^{500}$, $v_4^{501} = \mu_4^{500}$ $v_3^{501} = 0$
Initial Scalar of Step Size $\boldsymbol{\lambda}$	2
Test Platform	CPU: INTEL Pentium 4, 3GHz RAM: 1GB OS: Microsoft Windows XP

Parameters of the APS Model

Parameters	**Value**
Testing Topology	Grid network, Random network, Scale-free network
Number of Nodes $\lvert N \rvert$	25, 64, 100
The Total Defense Budget B	Equal to Number of Nodes
The Ratio of B1 to B2	0, 0.1 ,0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1
Total Attack Budget A	Equal to Total Defense Budget
Budget Allocation Strategy	Uniform allocation (b1), Degree-based allocation (b2), Piece-based allocation (b3)
Defense Capability $\hat{a}_i(b^c{}_i)$	Concave: $\hat{a}_i(b^c{}_i) = \log(10b^c{}_i+1) + \varepsilon$, Linear: $\hat{a}_i(b^c{}_i) = 3b^c{}_i + \varepsilon$, Convex: $\hat{a}_i(b^c{}_i) = (b^c{}_i)^{2.5} + \varepsilon$, $b^c{}_i$ is the budget allocated to node i to protect it from being compromised, $\forall i \in N$
Error probability on node *i* P(*b^e_i*)	$P(b^e{}_i) = p^0{}_i \times e^{-(b^e{}_i \times 0.5)}$, $p^0{}_i$ is initial random access error probability on node i; $b^e{}_i$ is the budget allocated to node i to decrease random access error probability on it, $\forall i \in N$

4.1.4 Experiment Results

In order to compare the attack effectiveness under different attack scenarios, we use the network susceptibility metric, which is defined in chapter 2.1. To evaluate the quality of LR based algorithms, we calculate the gap between LR and LB by $\frac{LB-LR}{LR}\times 100\%$, where LR represents the susceptibility under executing LR based algorithms. We also calculate the improvement ratio of LR to SA_1 and SA_2 by $\frac{LR-SA_1}{SA_1}\times 100\%$, $\frac{LR-SA_2}{SA_2}\times 100\%$.

Table 4-4 show the susceptibility under concave defense capability function, different network topologies, different budget allocation strategies and different ratios of B1 to B2.

Table 4-4 Experiment Result under concave defense capability function (|*N*| = 25)

Network Topology	Ratio of B1 to B2	Budget Allocation	LR (%)	Gap (%)	Improvement Ratio to SA1 (%)	Improvement Ratio to SA2 (%)
Grid Networks	0	b1	82.42%	5.04%	74.36%	25.93%
		b2	82.42%	10.3%	74.36%	103%
		b3	65.45%	2.858%	38.46%	61.19%
	0.1	b1	47.27%	0.009938%	9.859%	110.8%
		b2	64.24%	0.008459%	58.21%	58.21%
		b3	47.27%	0.009402%	9.859%	110.8%
	0.2	b1	65.45%	0.009974%	0%	38.46%
		b2	64.24%	0.009155%	158.5%	186.5%
		b3	47.27%	0.009264%	110.8%	110.8%
	0.3	b1	65.45%	0.009217%	38.46%	38.46%
		b2	64.24%	0.009565%	35.9%	58.21%
		b3	47.27%	0.009133%	90.24%	110.8%
	0.4	b1	65.45%	0.009391%	38.46%	0%
		b2	64.24%	0.009987%	35.9%	58.21%
		b3	64.24%	0.009346%	38.46%	38.46%
	0.5	b1	65.45%	0.1514%	52.11%	66.15%

Network Topology	Ratio of B1 to B2	Budget Allocation	LR (%)	Gap (%)	Improvement Ratio to SA1 (%)	Improvement Ratio to SA2 (%)
Grid Networks	**0.5**	**b2**	65.45%	0.4709%	191.9%	66.15%
		b3	64.24%	0.01307%	49.3%	63.08%
	0.6	**b1**	82.42%	17.41%	0%	0%
		b2	82.42%	19.3%	0%	0%
		b3	82.42%	0.7499%	0%	0%
	0.7	**b1**	100%	0%	83.33%	83.33%
		b2	100%	0%	83.33%	83.33%
		b3	54.55%	0.5336%	0%	0%
	0.8	**b1**	54.55%	0.8394%	0%	0%
		b2	54.55%	1.57%	0%	0%
		b3	54.55%	0.00973%	0%	0%
	0.9	**b1**	100%	0%	55.66%	0%
		b2	100%	0%	55.66%	0%
		b3	64.24%	47.1%	0%	0%
	1	**b1**	100%	0%	0%	21.32%
		b2	100%	0%	0%	21.32%
		b3	82.42%	7.139%	0%	0%

Network Topology	Ratio of B1 to B2	Budget Allocation	LR (%)	Gap (%)	Improvement Ratio to SA1 (%)	Improvement Ratio to SA2 (%)
Random Networks	**0**	**b1**	96.67%	0.8537%	58.9%	22.11%
		b2	96.67%	2.901%	58.9%	22.11%
		b3	79.17%	11.13%	30.14%	30.14%
	0.1	**b1**	79.17%	0.8337%	0%	0%
		b2	78.33%	1.24%	28.77%	28.77%
		b3	64.17%	0.009593%	5.479%	5.479%
	0.2	**b1**	79.17%	0.2539%	30.14%	30.14%
		b2	78.33%	2.906%	28.77%	28.77%
		b3	64.17%	0.009554%	5.479%	5.479%
	0.3	**b1**	79.17%	0.6143%	30.14%	30.14%
		b2	79.17%	1.976%	30.14%	30.14%
		b3	64.17%	0.009842%	5.479%	5.479%
	0.4	**b1**	81.18%	2.539%	30.14%	30.14%
		b2	84.98%	7.342%	30.14%	30.14%
		b3	70.35%	0.5007%	15.07%	15.07%
	0.5	**b1**	79.17%	3.86%	30.14%	30.14%

Network Topology	Ratio of B1 to B2	Budget Allocation	LR (%)	Gap (%)	Improvement Ratio to SA1 (%)	Improvement Ratio to SA2(%)
Random Networks	0.5	b2	79.17%	9.141%	30.14%	30.14%
		b3	79.17%	0.6328%	30.14%	30.14%
	0.6	b1	96.67%	2.848%	0%	0%
		b2	100%	0%	3.448%	3.448%
		b3	96.67%	1.841%	84.13%	84.13%
	0.7	b1	100%	0%	44.58%	44.58%
		b2	100%	0%	44.58%	44.58%
		b3	69.17%	1.541%	0%	0%
	0.8	b1	69.17%	0.4669%	0%	0%
		b2	69.17%	16.4%	0%	0%
		b3	69.17%	0.009619%	0%	0%
	0.9	b1	100%	0%	0%	0%
		b2	100%	0%	42.86%	0%
		b3	70%	32.77%	0%	0%
	1	b1	100%	0%	3.448%	0%
		b2	100%	0%	3.448%	0%
		b3	100%	0%	3.448%	0%

Network Topology	Ratio of B1 to B2	Budget Allocation	LR (%)	Gap (%)	Improvement Ratio to SA1 (%)	Improvement Ratio to SA2 (%)
Scale-free Networks	**0**	**b1**	86.89%	4.343%	0%	26.19%
		b2	86.89%	7.401%	26.19%	26.19%
		b3	71.31%	3.952%	3.571%	50%
	0.1	**b1**	53.28%	0.009052%	66.67%	12.07%
		b2	68.85%	0.09828%	115.4%	44.83%
		b3	53.28%	0.009158%	66.67%	12.07%
	0.2	**b1**	68.85%	0.009971%	115.4%	44.83%
		b2	68.85%	0.4998%	115.4%	44.83%
		b3	54.92%	0.009651%	71.79%	15.52%
	0.3	**b1**	68.85%	0.00993%	115.4%	44.83%
		b2	68.85%	6.29%	115.4%	44.83%
		b3	54.92%	0.009609%	71.79%	15.52%
	0.4	**b1**	68.85%	0.3342%	0%	44.83%
		b2	68.85%	7.198%	115.4%	44.83%
		b3	68.85%	0.009326%	0%	44.83%
	0.5	**b1**	79.17%	3.86%	30.14%	30.14%

Network Topology	Ratio of B1 to B2	Budget Allocation	LR (%)	Gap (%)	Improvement Ratio to SA1 (%)	Improvement Ratio to SA2 (%)
Scale-free Networks	0.5	**b2**	79.17%	9.141%	30.14%	30.14%
		b3	79.17%	0.6328%	30.14%	30.14%
	0.6	**b1**	78.69%	22.35%	0%	0%
		b2	78.69%	25.45%	57.38%	0%
		b3	78.69%	1.48%	174.3%	0%
	0.7	**b1**	100%	0%	125.9%	125.9%
		b2	100%	0%	125.9%	79.41%
		b3	55.74%	0.6083%	∞	25.93%
	0.8	**b1**	55.74%	1.945%	0%	0%
		b2	55.74%	7.043%	0%	∞
		b3	55.74%	0.009686%	∞	∞
	0.9	**b1**	100%	0%	64.86%	0%
		b2	100%	0%	64.86%	0%
		b3	60.66%	40.86%	0%	0%
	1	**b1**	100%	0%	27.08%	0%
		b2	100%	0%	27.08%	0%
		b3	100%	0%	27.08%	0%

Table 4-5 Experiment Result under concave defense capability function (|*N*| = 64)

Network Topology	Ratio of B1 to B2	Budget Allocation	LR (%)	Gap (%)	Improvement Ratio to SA1 (%)	Improvement Ratio to SA2 (%)
Grid Networks	0	b1	84.03%	18.02%	25.48%	25.48%
		b2	100%	0%	49.33%	49.33%
		b3	66.96%	21.9%	45.16%	56.61%
	0.1	b1	87.9%	10.92%	46.93%	46.93%
		b2	87.9%	12.07%	46.93%	46.93%
		b3	59.82%	5.677%	0%	102.3%
	0.2	b1	87.9%	10.96%	0%	0%
		b2	87.9%	12.31%	0%	0%
		b3	59.82%	3.974%	97.7%	0%
	0.3	b1	84.03%	11.33%	0%	0%
		b2	84.03%	17.23%	0%	0%
		b3	42.76%	19.22%	3.606%	0%
	0.4	b1	84.03%	13.41%	0%	0%
		b2	84.03%	18.3%	0%	0%
		b3	42.76%	20.72%	0%	0%
	0.5	b1	100%	0%	0%	0%

Network Topology	Ratio of B1 to B2	Budget Allocation	LR (%)	Gap (%)	Improvement Ratio to SA1 (%)	Improvement Ratio to SA2 (%)
Grid Networks	0.5	b2	100%	0%	91.63%	0%
		b3	52.18%	2.611%	0%	0%
	0.6	b1	100%	0%	99.1%	0%
		b2	100%	0%	0%	0%
		b3	50.23%	3.182%	∞	0.9042%
	0.7	b1	100%	0%	0%	0%
		b2	100%	0%	0%	0%
		b3	57.04%	0.8398%	0%	0%
	0.8	b1	100%	0%	35.85%	0%
		b2	100%	0%	35.85%	0%
		b3	73.61%	0.07861%	0%	0%
	0.9	b1	100%	0%	19.01%	0%
		b2	100%	0%	∞	∞
		b3	84.03%	0.7088%	∞	∞
	1	b1	100%	0%	∞	0%
		b2	100%	0%	∞	0%
		b3	100%	0%	526.1%	0%

Network Topology	Ratio of B1 to B2	Budget Allocation	LR (%)	Gap (%)	Improvement Ratio to SA1 (%)	Improvement Ratio to SA2 (%)
Random Networks	**0**	**b1**	82.99%	19.87%	30.41%	30.41%
		b2	82.99%	19.91%	30.41%	30.41%
		b3	63.64%	29.11%	45.77%	45.77%
	0.1	**b1**	90.46%	8.101%	43.37%	51.58%
		b2	90.46%	9.726%	0%	51.58%
		b3	59.68%	7.57%	84.68%	118.1%
	0.2	**b1**	90.46%	6.9%	43.37%	0%
		b2	90.46%	8.566%	43.37%	0%
		b3	63.1%	5.026%	95.26%	105%
	0.3	**b1**	80.65%	14.28%	0%	0%
		b2	80.65%	15.73%	0%	0%
		b3	42.48%	17.23%	0%	11.32%
	0.4	**b1**	80.65%	16.52%	0%	0%
		b2	80.65%	20.78%	0%	0%
		b3	42.48%	17.24%	0%	11.32%
	0.5	**b1**	100%	0%	99.1%	0%

Network Topology	Ratio of B1 to B2	Budget Allocation	LR (%)	Gap (%)	Improve-ment Ratio to SA1 (%)	Improve-ment Ratio to SA2(%)
Random Networks	0.5	b2	100%	0%	0%	0%
		b3	50.23%	3.182%	∞	0%
	0.6	b1	100%	0%	99.1%	0%
		b2	100%	0%	0%	0%
		b3	50.23%	3.182%	∞	0.9042%
	0.7	b1	100%	0%	0%	0%
		b2	100%	0%	82.73%	0%
		b3	54.73%	1.01%	0%	0%
	0.8	b1	100%	0%	∞	0%
		b2	100%	0%	49.93%	0%
		b3	66.7%	0.06017%	∞	0%
	0.9	b1	100%	0%	24%	0%
		b2	100%	0%	24%	0%
		b3	80.65%	0.2883%	0%	0%
	1	b1	100%	0%	10.55%	0%
		b2	100%	0%	0%	0%
		b3	90.46%	9.133%	0%	0%

Network Topology	Ratio of B1 to B2	Budget Allocation	LR (%)	Gap (%)	Improvement Ratio to SA1 (%)	Improvement Ratio to SA2 (%)
Scale-free Networks	**0**	**b1**	87.77%	13.22%	21.35%	21.35%
		b2	87.77%	12.44%	21.35%	21.35%
		b3	72.33%	18.91%	31.82%	31.82%
	0.1	**b1**	84.92%	13.9%	36.97%	36.97%
		b2	77.08%	14.49%	24.33%	24.33%
		b3	62%	5.913%	68.93%	68.93%
	0.2	**b1**	84.92%	10.89%	0%	36.97%
		b2	84.92%	8.397%	36.97%	36.97%
		b3	62%	3.671%	0%	0%
	0.3	**b1**	71.85%	23.38%	0%	0%
		b2	71.85%	11.68%	70.42%	0%
		b3	42.16%	6.746%	0%	0%
	0.4	**b1**	71.85%	27.64%	0%	0%
		b2	71.85%	11.31%	2.196%	0%
		b3	42.16%	8.575%	0%	0%
	0.5	**b1**	100%	0%	∞	0%

Network Topology	Ratio of B1 to B2	Budget Allocation	LR (%)	Gap (%)	Improvement Ratio to SA1 (%)	Improvement Ratio to SA2 (%)
Scale-free Networks	0.5	b2	51.19%	71.69%	∞	4.866%
		b3	51.19%	3.227%	∞	4.866%
	0.6	b1	100%	0%	∞	0%
		b2	51.19%	71.69%	∞	4.866%
		b3	51.19%	3.227%	∞	4.866%
	0.7	b1	100%	0%	69.42%	0%
		b2	59.03%	59.52%	0%	0%
		b3	59.03%	0.8287%	0%	0%
	0.8	b1	100%	0%	54.78%	0%
		b2	64.61%	22.24%	∞	0%
		b3	64.61%	0.06414%	0%	0%
	0.9	b1	100%	0%	39.17%	0%
		b2	71.85%	32.7%	0%	0%
		b3	71.85%	0.2233%	0%	0%
	1	b1	100%	0%	17.76%	0%
		b2	100%	0%	0%	0%
		b3	84.92%	8.659%	0%	0%

Table 4-6 Experiment Result under concave defense capability function ($|N|$ = 100)

Network Topology	Ratio of B1 to B2	Budget Allocation	LR (%)	Gap (%)	Improvement Ratio to SA1 (%)	Improvement Ratio to SA2 (%)
Grid Networks	0	b1	91.26%	7.809%	51.94%	2.711%
		b2	92.81%	7.079%	34.68%	16.57%
		b3	77.25%	17.85%	41.16%	11.32%
	0.1	b1	76.1%	15.01%	39.44%	4.317%
		b2	74.77%	25.54%	74.27%	2.489%
		b3	69.25%	20.11%	44.86%	23.59%
	0.2	b1	85.92%	10.32%	67.92%	0%
		b2	85.92%	11.41%	67.92%	0%
		b3	68.58%	19.24%	85.66%	0.817%
	0.3	b1	84.44%	17.29%	4.589%	0%
		b2	84.44%	17.85%	29.56%	0%
		b3	65.17%	16.27%	42.66%	50.6%
	0.4	b1	84.44%	18.1%	29.56%	0%
		b2	84.44%	18%	29.56%	0%
		b3	65.17%	17.33%	42.66%	4.206%
	0.5	b1	88.85%	5.685%	98.51%	0%

Network Topology	Ratio of B1 to B2	Budget Allocation	LR (%)	Gap (%)	Improvement Ratio to SA1 (%)	Improvement Ratio to SA2 (%)
Grid Networks	0.5	b2	88.85%	6.289%	0%	0%
		b3	69.77%	6.398%	171.7%	46.99%
	0.6	b1	88.85%	7.607%	98.51%	0%
		b2	88.85%	10.62%	33.52%	0%
		b3	69.77%	10.04%	55.88%	0%
	0.7	b1	88.85%	7.607%	98.51%	0%
		b2	88.85%	10.62%	33.52%	0%
		b3	69.77%	10.04%	55.88%	0%
	0.8	b1	88.85%	12.13%	98.51%	0%
		b2	88.85%	12.09%	27.35%	0%
		b3	69.77%	13.85%	55.88%	0%
	0.9	b1	100%	0%	87.69%	0%
		b2	100%	0%	221.7%	0%
		b3	100%	0%	87.69%	0%
	1	b1	100%	0%	88.09%	0%
		b2	100%	0%	31.53%	0%
		b3	100%	0%	309.6%	0%

Network Topology	Ratio of B1 to B2	Budget Allocation	LR (%)	Gap (%)	Improvement Ratio to SA1 (%)	Improvement Ratio to SA2 (%)
Random Networks	**0**	**b1**	91.37%	6.319%	29.36%	8.378%
		b2	90%	10.61%	27.65%	12.03%
		b3	78.93%	14.78%	37.98%	31.07%
	0.1	**b1**	76.41%	15.18%	115.9%	18.97%
		b2	76.41%	27.93%	67.12%	4.757%
		b3	65.63%	18.38%	85.41%	2.186%
	0.2	**b1**	79.72%	14.89%	0%	0%
		b2	85.21%	16.03%	31.23%	6.891%
		b3	65.06%	18.41%	0.1908%	0%
	0.3	**b1**	81.74%	20.59%	33%	0%
		b2	81.74%	21.9%	0%	0%
		b3	61.46%	19.95%	0.2695%	0%
	0.4	**b1**	81.74%	21.84%	2.539%	0%
		b2	100%	0%	25.57%	22.33%
		b3	61.46%	20.9%	50.15%	0.1346%
	0.5	**b1**	88.31%	5.826%	33.96%	0%

Network Topology	Ratio of B1 to B2	Budget Allocation	LR (%)	Gap (%)	Improvement Ratio to SA1 (%)	Improvement Ratio to SA2(%)
Random Networks	0.5	b2	88.31%	7.703%	29.42%	0%
		b3	68.24%	8.215%	49.1%	0%
	0.6	b1	88.31%	8.895%	33.96%	0%
		b2	88.31%	12.25%	29.42%	0%
		b3	68.24%	11.98%	48.83%	0%
	0.7	b1	88.31%	8.895%	33.96%	0%
		b2	88.31%	12.25%	29.42%	0%
		b3	68.24%	11.98%	48.83%	0%
	0.8	b1	88.31%	12.8%	29.42%	0%
		b2	100%	0%	25.12%	13.24%
		b3	68.24%	15.79%	0%	0%
	0.9	b1	100%	0%	28.91%	0%
		b2	100%	0%	0%	0%
		b3	100%	0%	28.91%	0%
	1	b1	100%	0%	154%	0%
		b2	100%	0%	46.82%	0%
		b3	100%	0%	154%	0%

Network Topology	Ratio of B1 to B2	Budget Allocation	LR (%)	Gap (%)	Improvement Ratio to SA1 (%)	Improvement Ratio to SA2 (%)
Scale-free Networks	**0**	**b1**	91.48%	6.312%	28.4%	0%
		b2	91.48%	8.97%	43.12%	10.06%
		b3	78.38%	14.37%	41.08%	9.394%
	0.1	**b1**	77.5%	13.53%	69.36%	0%
		b2	77.5%	25.42%	62.03%	1.143%
		b3	72.7%	16.68%	48.47%	17.48%
	0.2	**b1**	84.79%	10.59%	60.2%	0%
		b2	86.06%	13.58%	69.76%	2.03%
		b3	71.96%	12.81%	84.58%	45.14%
	0.3	**b1**	83.91%	18.17%	25.49%	1.152%
		b2	83.91%	18.3%	96.18%	1.152%
		b3	66.87%	15.11%	49.51%	0%
	0.4	**b1**	83.91%	18.74%	25.49%	1.152%
		b2	83.91%	18.41%	87.62%	1.152%
		b3	66.87%	15.37%	49.51%	0%
	0.5	**b1**	87.89%	6.39%	33.84%	0%

Network Topology	Ratio of B1 to B2	Budget Allocation	LR (%)	Gap (%)	Improvement Ratio to SA1 (%)	Improvement Ratio to SA2 (%)
Scale-free Networks	0.5	b2	87.89%	7.125%	82.4%	0%
		b3	70.41%	5.585%	46.12%	0%
	0.6	b1	87.89%	8.7%	253.7%	0%
		b2	87.89%	9.623%	82.4%	0%
		b3	70.41%	9.252%	183.3%	0%
	0.7	b1	87.89%	8.7%	253.7%	0%
		b2	87.89%	9.623%	82.4%	0%
		b3	70.41%	9.252%	183.3%	0%
	0.8	b1	87.89%	13.12%	33.84%	0%
		b2	87.89%	12.75%	295.5%	0%
		b3	76.66%	8.366%	59.09%	8.88%
	0.9	b1	100%	0%	88.09%	0%
		b2	100%	0%	31.53%	0%
		b3	100%	0%	309.6%	0%
	1	b1	100%	0%	43.08%	0%
		b2	100%	0%	55.77%	0%
		b3	100%	0%	43.08%	0%

Table 4-7 Experiment of different defense capability function (|*N*| = 100)

Network Topology	Ratio of B1 toB2	defense capability function	LR (%)
Grid Networks	0	concave	77.25%
		linear	39.46%
		convex	26.45%
	0.1	concave	69.25%
		linear	39.24%
		convex	39.13%
	0.2	concave	68.58%
		linear	36.38%
		convex	19.04%
	0.3	concave	65.17%
		linear	23.79%
		convex	21.9%
	0.4	concave	65.17%
		linear	43.28%
		convex	35.05%
	0.5	concave	69.77%
		linear	40.87%
		convex	41.39%
	0.6	concave	69.77%
		linear	47.98%
		convex	88.85%
	0.7	concave	69.77%
		linear	47.98%
		convex	88.85%
	0.8	concave	69.77%
		linear	69.77%
		convex	100%
	0.9	concave	100%
		linear	100%
		convex	100%
	1	concave	100%
		linear	100%
		convex	100%

Network Topology	Ratio of B1 toB2	defense capability function	LR (%)
Random Networks	**0**	**concave**	78.93%
		linear	39.03%
		convex	31.72%
	0.1	**concave**	65.63%
		linear	35.89%
		convex	35.89%
	0.2	**concave**	65.06%
		linear	36.38%
		convex	30.19%
	0.3	**concave**	61.46%
		linear	33.17%
		convex	20.65%
	0.4	**concave**	61.46%
		linear	41.1%
		convex	38.91%
	0.5	**concave**	68.24%
		linear	45.85%
		convex	45.85%
	0.6	**concave**	68.24%
		linear	45.85%
		convex	88.31%
	0.7	**concave**	68.24%
		linear	45.85%
		convex	88.31%
	0.8	**concave**	68.24%
		linear	68.24%
		convex	100%
	0.9	**concave**	100%
		linear	100%
		convex	100%
	1	**concave**	100%
		linear	100%
		convex	100%

Network Topology	Ratio of B1 toB2	defense capability function	LR (%)
Scale-free Networks	0	concave	78.38%
		linear	39.71%
		convex	35.48%
	0.1	concave	72.7%
		linear	42.45%
		convex	43.05%
	0.2	concave	71.96%
		linear	39.15%
		convex	34.49%
	0.3	concave	66.87%
		linear	42.77%
		convex	39.19%
	0.4	concave	66.87%
		linear	44.72%
		convex	41.14%
	0.5	concave	70.41%
		linear	45.56%
		convex	54.44%
	0.6	concave	70.41%
		linear	64.56%
		convex	87.89%
	0.7	concave	70.41%
		linear	64.56%
		convex	87.89%
	0.8	concave	76.66%
		linear	70.41%
		convex	100%
	0.9	concave	100%
		linear	100%
		convex	100%
	1	concave	100%
		linear	100%
		convex	100%

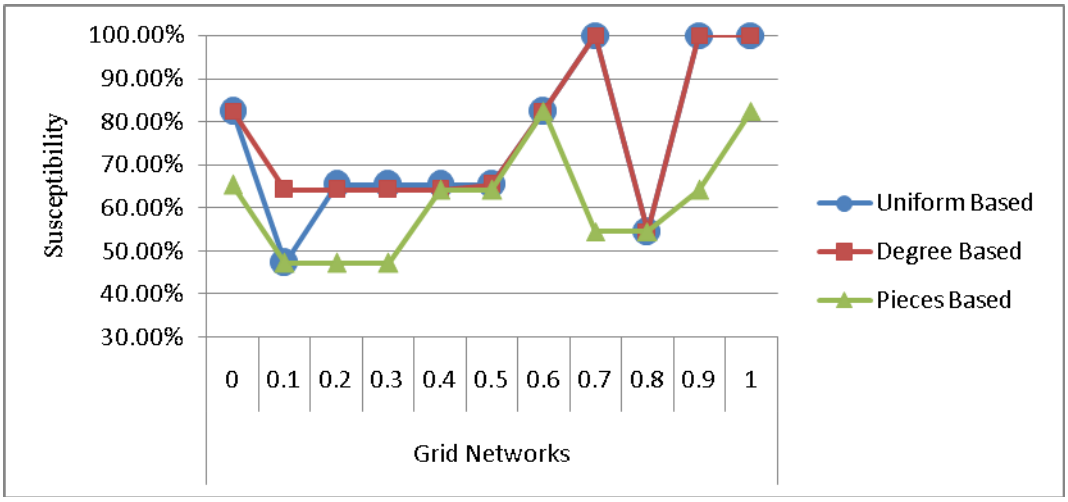


Figure 4-1 Susceptibility of Grid Networks under Different Scenarios (|N| = 25)

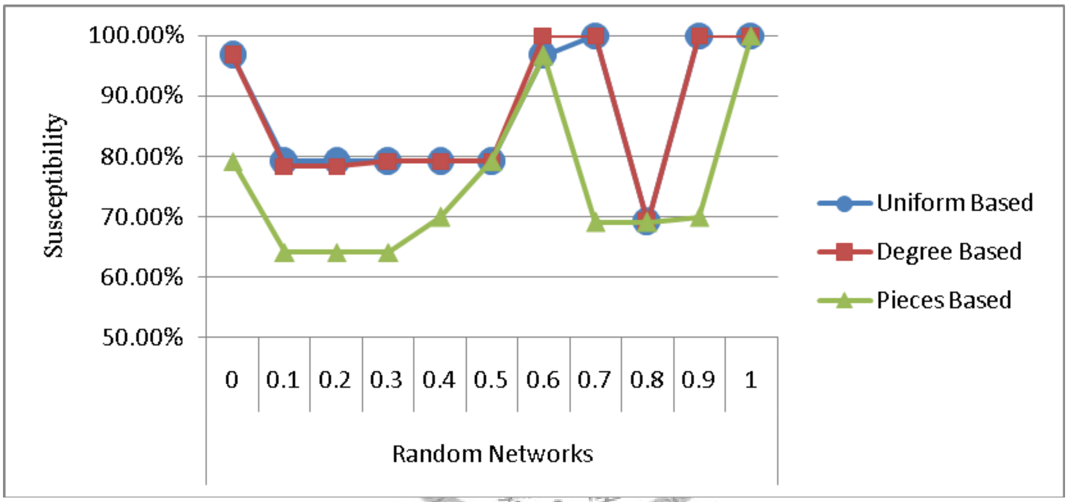


Figure 4-2 Susceptibility of Random Networks under Different Scenarios (|N| = 25)

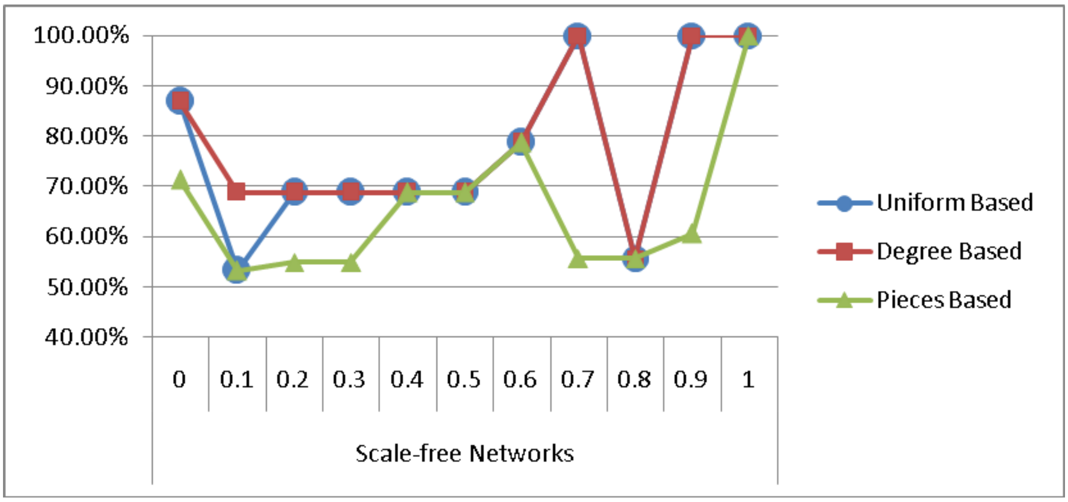


Figure 4-3 Susceptibility of Scale-free Networks under Different Scenarios (|N| = 25)

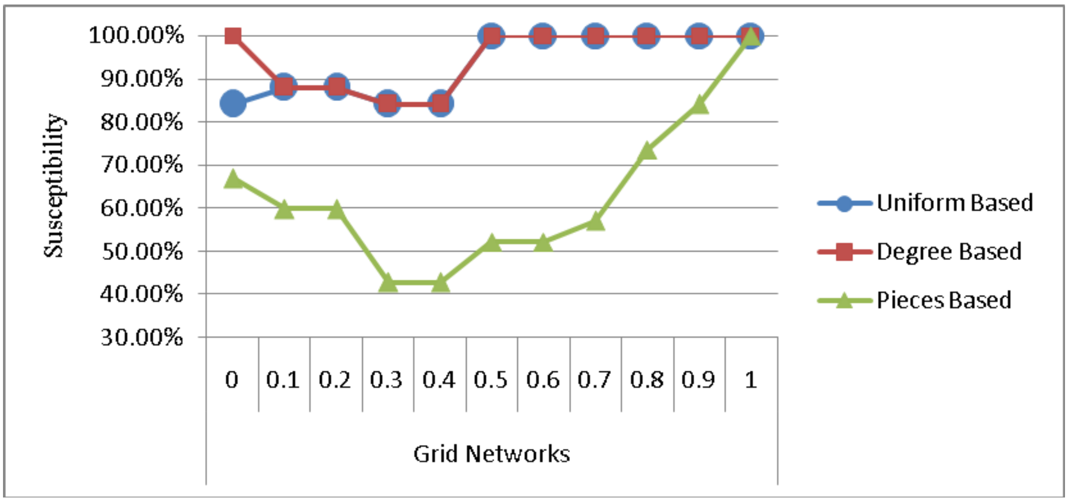


Figure 4-4 Susceptibility of Grid Networks under Different Scenarios (|N| = 64)

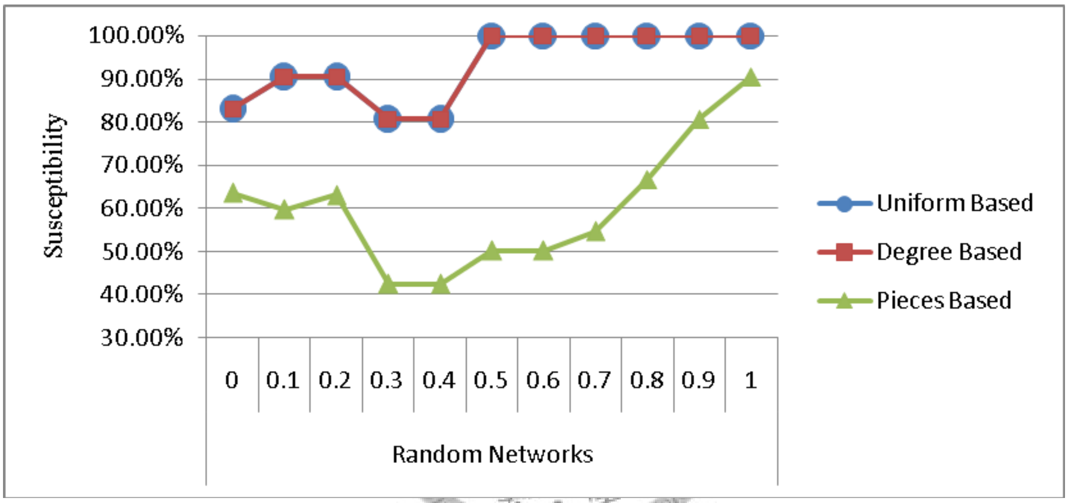


Figure 4-5 Susceptibility of Random Networks under Different Scenarios (|N| = 64)

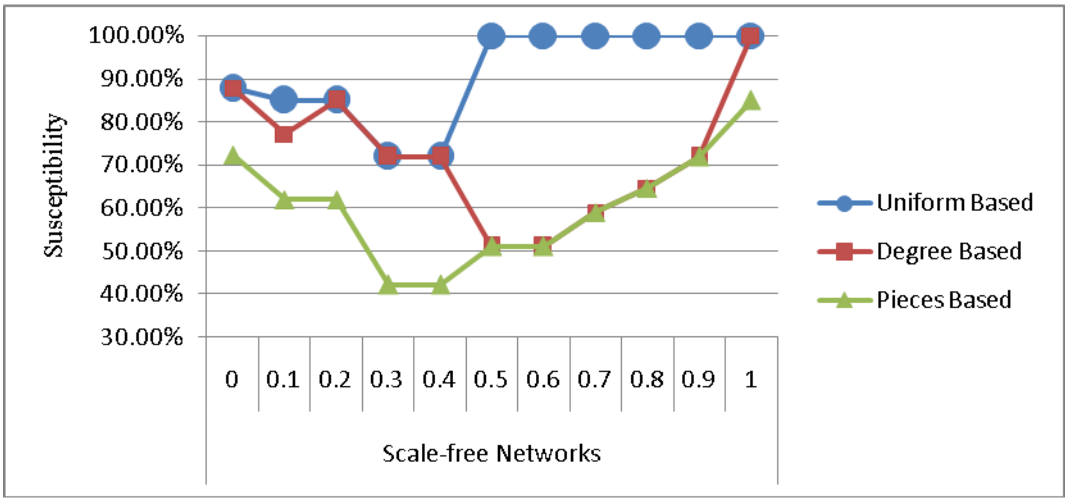


Figure 4-6 Susceptibility of Scale-free Networks under Different Scenarios (|N| = 64)

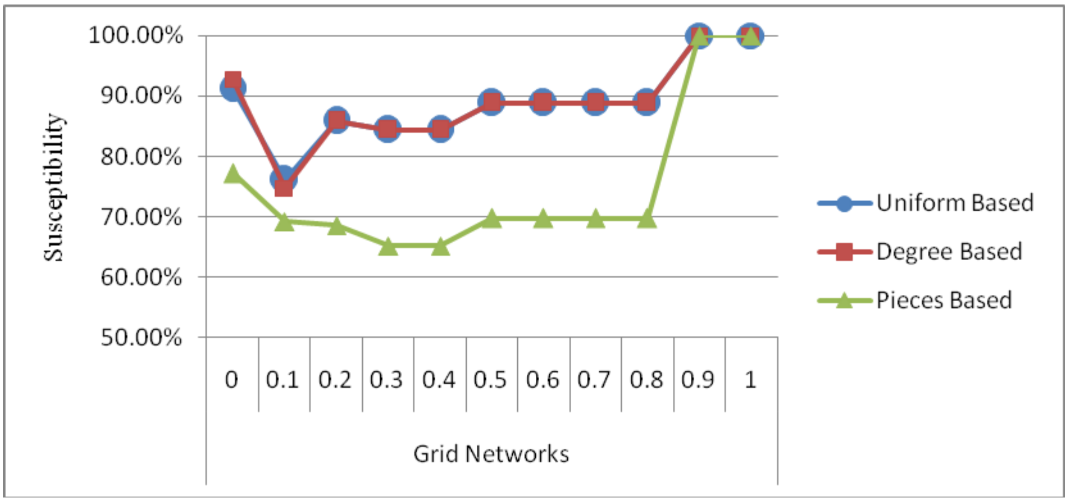


Figure 4-7 Susceptibility of Grid Networks under Different Scenarios (|N| = 100)

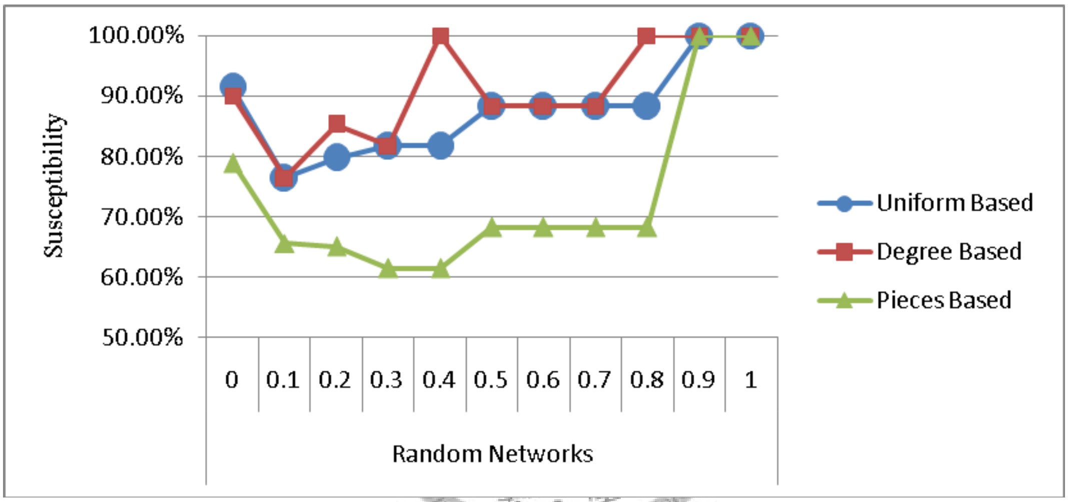


Figure 4-8 Susceptibility of Random Networks under Different Scenarios (|N| = 100)

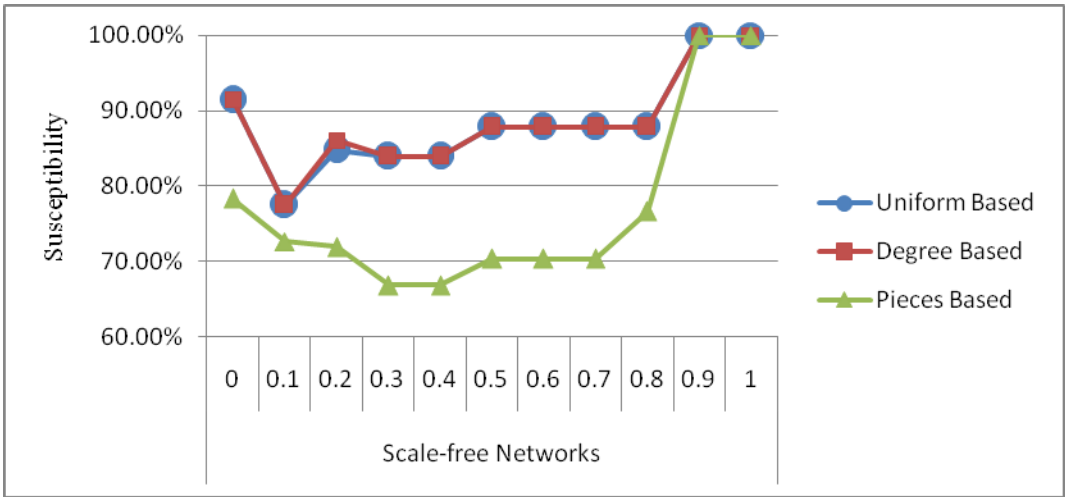


Figure 4-9 Susceptibility of Scale-free Networks under Different Scenarios (|N| = 100)

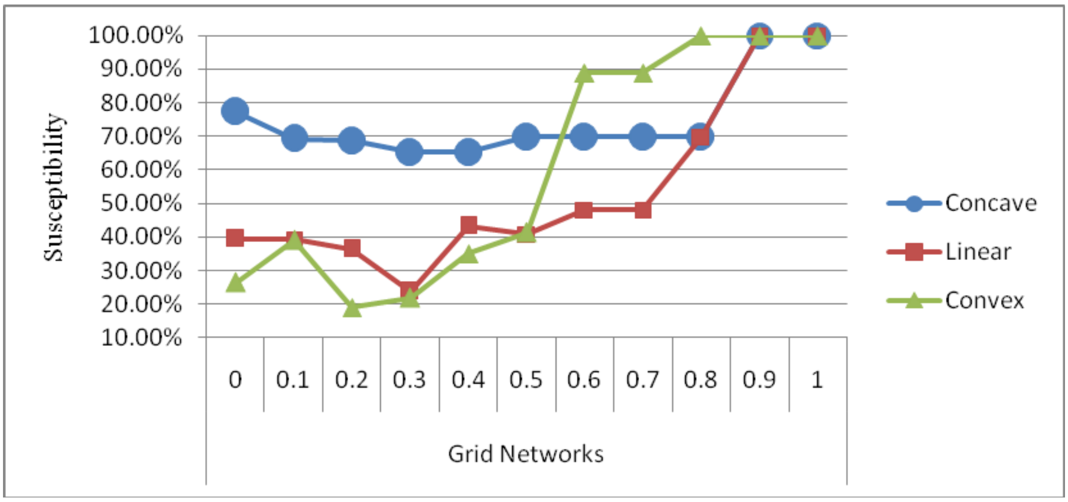


Figure 4-10 Susceptibility of Grid Networks under Different defense capability function (|N| = 100)

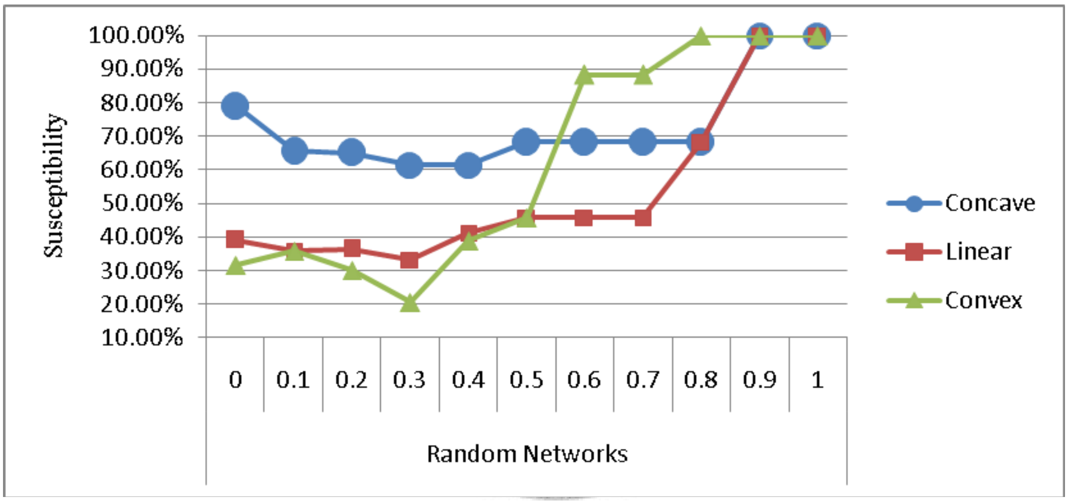


Figure 4-11 Susceptibility of Random Networks under Different defense capability function (|N| = 100)

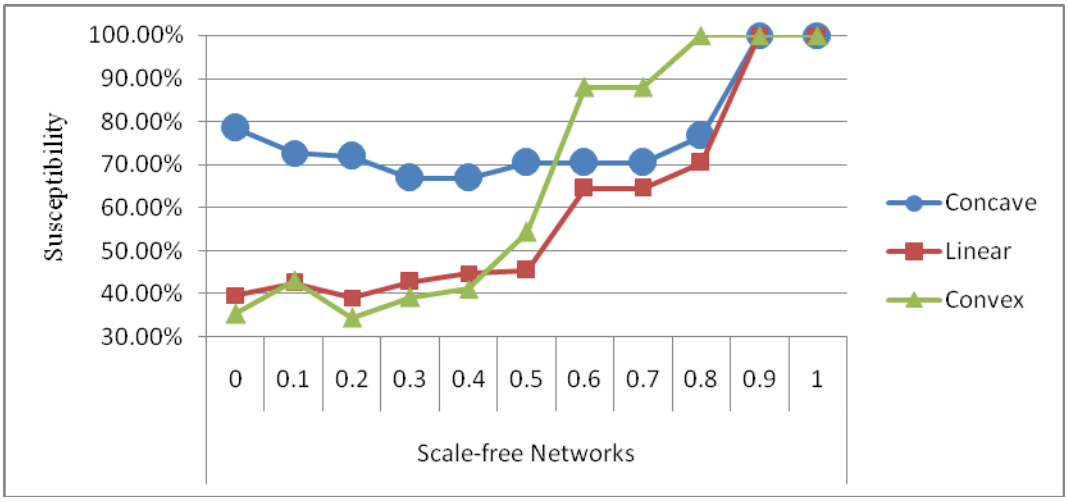


Figure 4-12 Susceptibility of Scale-free Networks under Different defense capability function (|N| = 100)

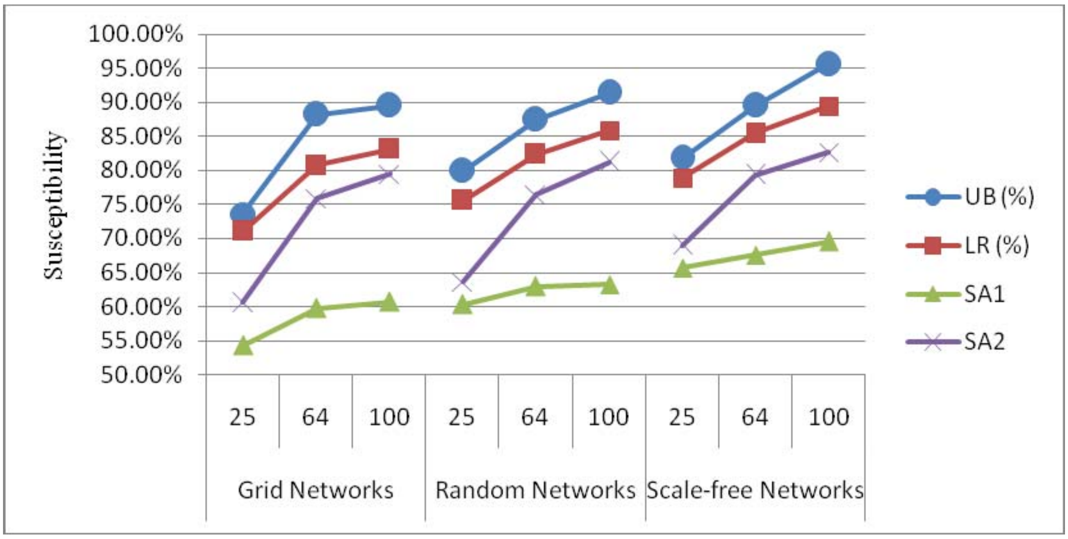


Figure 4-13 Susceptibility of diffrernt Networks and Topologies

4.1.5 Discussion of Results

Figures 4-1 to 4-9 show the susceptibility of the targeted network under concave defense capability function and different topology types, nodes, and ratios of B1 to B2. From these figures, we observe:

- Network with pieces based defense budget allocation strategy are the robust that means it is difficult for the attacker to recover information. Since the attacker's objective is to collect pieces to recover information, he/she will compromise nodes containing more pieces. Therefore, to allocate more defense budget to nodes which contain more pieces will be better for the defender.
- The susceptibilities of different topologies have similar trends under pieces based defense budget allocation strategy with the same number of

nodes.

- Generally, it is the best defense budget allocation strategy when the ratio of B1 to B2 is equal to 0.3. It means that to set B2 twice as much as B1 is a more appropriate strategy.
- Degree based and uniform based budget allocation strategies do not work under information dividing scenario. Because the important nodes (nodes with more pieces on them) are not allocated enough defense budgets to them.
- Generally, it is easier for the attacker to recover information when the great part of defense budget is allocated to decrease nodes' random error probability.

Figure 4-10 to 4-12 show the susceptibility under different defense capability function and topologies, we observe:

- It is easier for the attacker to recover information when the ratio of B1 to B2 is more than 0.5 under convex defense capability function. This is due to the reason that most nodes are allocated to less than one unit defense budget.
- Generally, the susceptibility is increasing progressively under linear and convex defense capability.

- The susceptibility is increasing rapidly when the ratio of B1 to B2 exceeding 0.7.
- Generally, the shape of graph is “U” under concave defense capability.

Figure 4-13 compares the solution quality of LB based algorithm with simple algorithm 1 and 2. It also shows the gap between LRs and UBs. The value of each point on figure 4-13 is the average susceptibility of different budget allocation strategies, ratios of B1 to B3 under same network size and topology. From the figure, we observe:

- The performance of LR based algorithm is better than other algorithms.
- The gaps between LRs and UBs are small, which shows that our proposed approach for the APS model is near optimal.
- The performance of SA2 is better than SA1. In SA2, the attacker can find the best one to compromise, it make the attacker easier recover information.

4.2 Computational Experiment with the DRAID Model

4.2.1 Experiment Environment

We transform the proposed algorithms for solving the DRAID model into codes in Visual C++ and execute them in a PC with an INTEL Pentium 4, 3GHz CPU. The Iteration Counter Limit, Improve Counter Limit, and Switch Counter Limit are set to 100, 10, and 20 respectively.. We initiate the step size scalar, Θ, as 0.5 and halve it if the improvement of the objective function value, Z_D, does not occur during a period of Improve Counter Limit.

In the DRAID Mode, the defender wants to minimize the maximized damage incurred by the attacker. Initially, we use the result from solving the APS model as input of the DRAID model to adjust the defense allocation strategy and pieces allocation strategy. After the adjustment, we solve the APS model again according to the current defense strategy. The interaction is repeated until Iteration Counter Limit is reached. Since damage based defense budget allocation strategy is the best one for the defender in the APS model, we adopt this strategy to solve DRAID model. In addition, we adopt 0.3 ratio of B1 to B2 to solve DRAID model because it is also the best ratio for the defender in the APS model.

We adopt three reallocation strategies for comparison. The first is degree based; the second is pieces based; the third is bonus based. The parameters and scenarios used in our experiments are described in Table 4-7.

Table 4-8 Experiment Parameter Settings for the DRAID Model

Parameters of Adjustment_Procedure

Parameters	Value
Iteration Counter Limit	100
Improve Counter Limit	10
Switch Counter Limit	20
Initial Scalar of Step Size $\boldsymbol{\theta}$	0.5
Test Platform	CPU: INTEL Pentium 4, 3GHz RAM: 1GB OS: Microsoft Windows XP

Parameters of the DRAID Model

Parameters	Value
Testing Topology	Grid network, Random network, Scale-free network
Number of Nodes |*N*|	25, 64, 100
The Total Defense Budget B	Equal to Number of Nodes
The Ratio of B1 to B2	0.3
Total Attack Budget A	Equal to Total Defense Budget
Initial Budget Allocation Strategy	Piece-based allocation(b3)
Budget Reallocation Strategy	Degree-based allocation (b2), Pieces-based allocation (b3), Bonus based (b4)
Defense Capability $\hat{a}_i(b^c{}_i)$	Concave: $\hat{a}_i(b^c{}_i) = \log(10b^c{}_i+1) + \varepsilon$, $b^c{}_i$ is the budget allocated to node i to protect it from being compromised, $\forall i \in N$
Error probability on node *i* P($\boldsymbol{b^e{}_i}$)	$\mathrm{P}(b^e{}_i) = \mathrm{p}^0{}_\mathrm{i} \times e^{-(b^e{}_i \times 0.5)}$, $\mathrm{p}^0{}_\mathrm{i}$ is initial random access error probability on node i; $b^e{}_i$ is the budget allocated to node i to decrease random access error probability on it, $\forall i \in N$

4.2.2 Experiment Results

In the experiments, we use the survivability metric to evaluate the performance of different defense budget reallocation strategies. The Init. Surv. value represents the network survivability under initial defense budget allocation, and the Opt. Surv. value is the best network survivability during executing several times adjustment procedure. The improvement ratio of Opt. Surv. to Init. Surv. is calculated by $\frac{Opt.Surv. - Init.Surv.}{Init.Surv.} \times 100\%$.

Table 4-9 Experiment Results of small networks (|N| = 25)

Network Topology	Init. Surv.	Budget Reallocation	Opt. Surv.	Imp. Ratio of Opt. Surv.
Grid Networks	52.73%	Degree Based	52.73%	0%
		Pieces Based	55.54%	5.32%
		Bonus Based	58.73%	11.37%
Random Networks	35.83%	Degree Based	35.83%	0%
		Pieces Based	40.43%	12.83%
		Bonus Based	42.67%	19.09%
Scale-free Networks	45.08%	Degree Based	45.08%	0%
		Pieces Based	47.34%	5.01%
		Bonus Based	48.32%	7.18%

Table 4-10 Experiment Results of Medium-sized networks (|N| = 64)

Network Topology	Init. Surv.	Budget Reallocation	Opt. Surv.	Imp. Ratio of Opt. Surv.
Grid Networks	57.24%	Degree Based	57.24%	0%
		Pieces Based	63.34%	10.65%
		Bonus Based	65.18%	13.87%

Random Networks	57.52%	Degree Based	58.84%	2.29%
		Pieces Based	61.84%	7.51%
		Bonus Based	61.84%	7.51%
Scale-free Networks	57.84	Degree Based	57.84	0%
		Pieces Based	60.23	4.13%
		Bonus Based	60.23	4.13%

Table 4-11 Experiment Results of Large networks (|N| = 100)

Network Topology	Init. Surv.	Budget Reallocation	Opt. Surv.	Imp. Ratio of Opt. Surv.
Grid Networks	34.83%	Degree Based	41.35%	7.23%
		Pieces Based	41.16%	18.17%
		Bonus Based	47.24%	35.63%
Random Networks	38.54%	Degree Based	40.64%	5.44%
		Pieces Based	42.64%	10.63%
		Bonus Based	42.64%	10.63%
Scale-free Networks	33.13%	Degree Based	37.14	12.10%
		Pieces Based	39.59	19.47%
		Bonus Based	42.09	27.04%

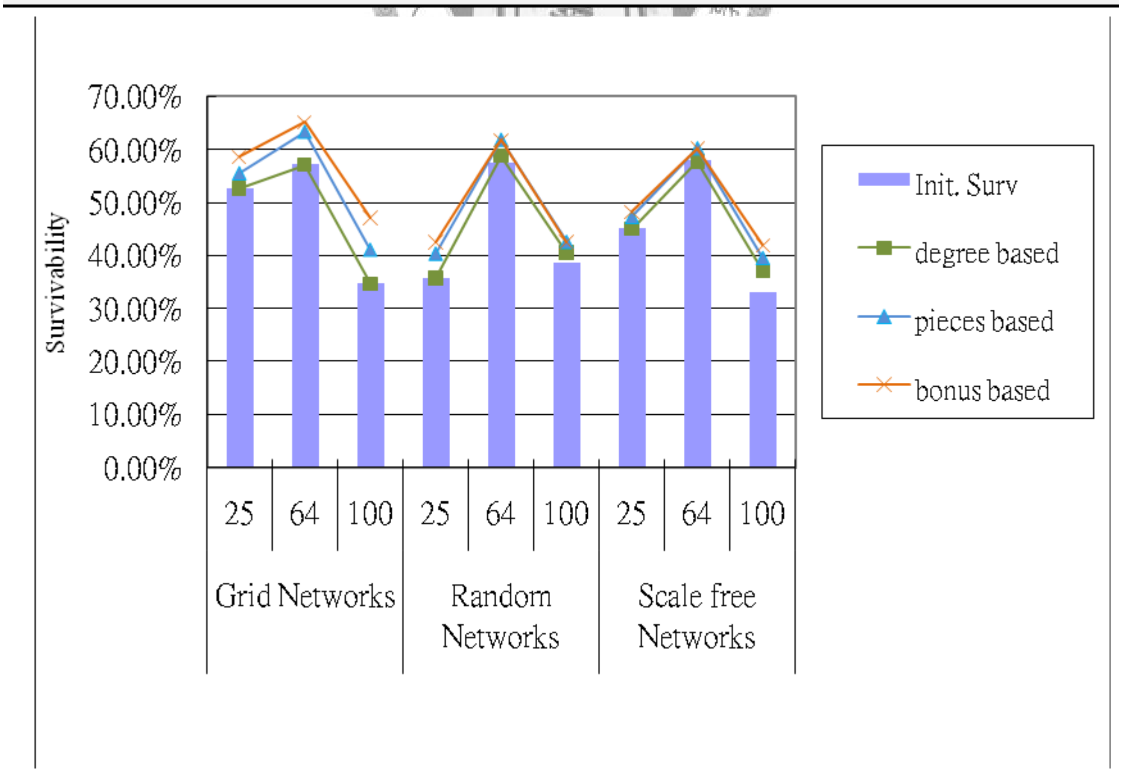


Figure 4-14 Survivability of Different Defense Budget Reallocation Strategy

4.2.3 Discussion of Results

Figure 4-14 show the equilibrium survivability of the targeted networks under different topologies, numbers of nodes, and reallocation strategies. From this figure, we observe:

- The survivability of network can be improved by bonus based or pieces based budget reallocation strategy. It is because more budgets are allocated to the nodes with more pieces.
- The degree based budget reallocation strategy causes little survivability improvement. It is because the nodes with higher degree number may not have higher piece number.
- The best budget allocation strategy is to allocate more defense budget on important nodes, which have more pieces, instead of wasting it on relative valueless nodes (on the attacker’s aspect).

Chapter 5 Conclusion and Future Work

5.1 Conclusion

Internet interconnects the whole world, and makes the world a global village. Individuals, enterprises, and other organizations attend to put information in the network for required people to access. The network is convenient but also vulnerable because of its public characteristic. For some sensitive information, it is only available for some specific groups. Thus, how to protect this information from being stolen by unauthorized user is important.

As a network operator, he/she have to design appropriate defense strategies to protect information from being stolen. In this thesis, we have addressed the attack-defense scenario about information theft, where the attack want to stole information to gain maximal profit while the defender want to decrease damage incurred by information leakage.

The first key contribution of this thesis is we proposed a min-max mathematical model to describe the defense resource allocation strategy problem and the attack path selecting problem. With our efforts, we successfully model the interaction between the attack and the defender in the real world into mathematical models. Moreover, we proposed heuristics to solve the problems.

The second key contribution of this thesis is we not only concern about malicious attack but also random error on nodes. In our study, we evaluate the efficiency under different ratios of budget 1 to budget 2, where budget 1 is allocated to nodes to protect them from being compromised and budget 2 is allocated to nodes to decrease random error on them.

The third key contribution of this thesis is we use the technique of information dividing to develop our defense strategies. We divide information into pieces and allocate these pieces to nodes, and then allocate defense budget to nodes to protect information from being recovered by the attacker.

We have also proposed proper metrics, which are susceptibility and survivability, to evaluate the performance of proposed algorithms. The susceptibility metric represent the proportion of total information value that the attack gain by recovering information; The survivability metric represent the proportion of total information value which is not recovered by the attacker in the target network. According to the metrics, both the attacker and the defender can adjust their strategies to get a better result. In addition, we have studied several different network topologies and observed their robustness against information theft.

5.2 Future Work

In the following, we raise some issues that could be studied further.

- **Design of Honey-Pot**

  The technique of Honey Pot is a kind of fraud skill, which drives an attacker to exhaust his/her energy to launch attack on valueless objects. We can use the technique of honey-pot to waste the attacker's attack budget. For example, we allocate a little information pieces to some nodes as honey-pots. The attack will exhaust his/her attack budget when attacking these honey-pots but gaining little profit.

- **Discussion of critical points**

  In the APS model, an attacker's objective is to collect pieces to recover information. Therefore, the key point to attack is which nodes contain sufficient pieces to recover information. In the network, there exist some critical points, as long as the attacker compromised them, attacker can gain profound profit immediately. Relatively, for the defender, he/she can allocate sufficient defense budget to these critical points to protect information from being recovered. Thus, how to find these critical points is an important issue.

- **Discussion of different pieces allocation patterns**

For an attacker, different pieces allocation pattern may cause different attack behavior. Hence, we can design different defense resource allocation strategy for different pieces allocation pattern. Many pieces allocation patterns can be studied further, such as random based and degree based pieces allocation pattern.

- **<u>Experience of an Attacker</u>**

In the real world, an attacker may obtain some experience after compromising nodes, and this experience will improve the performance of the next attack action. Thus, we can consider this factor in our models to make the models more close to the real world.

References

[1] R. Richardson, "2007 CSI Computer Crime and Security Survey", *Computer Security Institute*, 2007, http://GoCSI.com.

[2] L.A. Gordon, M.P. Loeb, W. Lucyshyn, and R. Richardson, "2006 CSI/FBI Computer Crime and Security Survey", *Computer Security Institute*, 2006, http://GoCSI.com.

[3] P. Tarvainen, "Survey of the Survivability of IT Systems," *The 9th Nordic Workshop on Secure IT-systems*, November 2004.

[4] J.C. Knight and K.J. Sullivan, "On the Definition of Survivability," *Technical Report CS-TR-33-00, Department of Computer Science, University of Virginia*, December 2000.

[5] J.C. Knight, E.A. Strunk, and K.J. Sullivan, "Towards a Rigorous Definition of Information System Survivability," *Proceedings of the DARPA Information Survivability Conference and Exposition (DISCEX 2003)*, Volume 1, pp. 78-89, April 2003.

[6] V.R. Westmark, "A Definition for Information System Survivability," *Proceedings of the 37th IEEE Hawaii International Conference on System Sciences*, Vol. 9, 2004.

[7] S.C. Liew and K.W. Lu, “A Framework for Network Survivability Characterization,” *IEEE Journal on Selected Areas in Communications*, Vol. 12, No. 1, pp. 52-58, January 1994 (ICC, 1992).

[8] J.L Tzeng, “Near Optimal Network Defense Resource Allocation Strategies for the Minimization of Information Leakage”, *Department of Information Management, National Taiwan University*, 2006.

[9] Adi Shamir, “How to Share a Secret”, *Massachusetts Institute of Technology*, 1979.

[10] S.C. Cha, Y.J. Joung, and Y.E. Lue, “Building Universal Profile System over a Peer-to-Peer Network”, *Department of Information Management, National Taiwan University, 2003.*

[11] C.S. Laih, L. Harn, and C.C. Chang, “Contemporary Cryptography and Its Applications”, PP 231-245, 1995.

[12] Andrew S. Tanenbaum, “Computer Networks”, 3rd Edition, 1997.

[13] “INFORMATION SECURITY TAIWAN”, pp. 22-23, No.47, November 2007.

[14] M.L. Fisher, “The Lagrangean Relaxation Method for Solving Integer Programming Problems,” *Management Science*, Vol. 27, No. 1, pp. 1-18, January 1981.

[15] M.L. Fisher, “An Application Oriented Guide to Lagrangean Relaxation,”

Interfaces, Vol. 15, No 2, pp. 10-21, April 1985.

[16] A.M. Geoffrion, “Lagrangean Relaxation and its Use in Integer Programming,” 86 *Mathematical Programming Study*, Vol. 2, pp. 82-114, 1974.

[17] Hakim Weatherspoon and John D. Kubiatowicz “Erasure coding vs. replication: A quantitative comparison”. *Lecture Notes in Computer Science*, 2429:328-339, 2002

[18] S. Rhea, C. Wells, P. Eaton, D. Geels, B. Zhao, H. Weatherspoon, and J. Kubiatowicz. “Maintenance-free global datastorage,” *IEEE Internet Computing*, pages 40–49, 2001.

簡歷

姓名：蘇至浩

出生地：台灣 屏東縣

生日：中華民國七十三年七月二十三日

學歷：九十一年九月至九十五年六月

國立中央大學資訊管理學系學士

九十五年九月至九十七年七月

台灣大學資訊管理研究所碩士