Protecting private data in information systems

Yibing Kong
University of Wollongong
Protecting Private Data in Information Systems

A thesis submitted in fulfillment of the requirements for the award of the degree

Doctor of Philosophy

from

UNIVERSITY OF WOLLONGONG

by

Yibing Kong

School of Information Technology and Computer Science
May 2007
© Copyright 2007

by

Yibing Kong

All Rights Reserved
Dedicated to
Qingzhi Kong, Lijun Han and Yingxu Liu
CERTIFICATION

I, Yibing Kong, declare that this thesis, submitted in fulfilment of the requirements for the award of Doctor of Philosophy, University of Wollongong, is wholly my own work unless otherwise referenced or acknowledged. The document has not been submitted for qualifications at any other academic institution.

_____________________________

Yibing Kong
May 10, 2007
abstract

The use of private data is ubiquitous. On one hand, people submit their private data to obtain services. On the other hand, organizations need private data to carry out their business. Characterized by convenience, efficiency and cost-saving, information systems are useful for private data management. As a result, vast amounts of private data are collected and processed electronically. However, inadequate protection may end up with the abuse of private data. Privacy concerns affect people’s attitude towards providing their private data, which restricts the success of organizations’ business. The importance of privacy control is well recognized today. Privacy control should be regarded as an imperative design criterion for information systems [40].

The common ground between privacy and security allows us to develop privacy protection techniques based on existing security protection techniques. In the past decade, a few studies have been conducted in this area. Nevertheless, they give individuals limited control over their private data. In particular, after an individual submits his/her private data to an organization, he/she almost loses control over it. This thesis considers this lack of control as a potential problem in information systems. Based on existing security protection techniques, three privacy protection approaches are proposed: an access control based approach, a hierarchical encryption based approach and a digital ticket based approach. These approaches are highlighted for their consideration for information donors’ privacy preferences.
First, I would like to thank my supervisors Professor Jennifer Seberry, Dr. Janusz R. Getta, and Dr. Ping Yu without whom this work would not have been possible. 

I also would like to thank Dr. Tianbing Xia, Associate Professor Yi Mu, Ken Finlayson for their kind help on me for the development of my thesis.

My thanks also go to all members of this Crypto group, all staff in the department of computer science, for the help they rendered me.

Finally I am grateful to the support from my family and friends.
Publications From This Thesis


# Contents

Abstract v  
Acknowledgements vi  
Publications From This Thesis vii  

1 Introduction 1  
1.1 Introduction ................................. 1  
1.2 Motivation ................................. 3  
1.3 Proposed Solution ............................ 4  
1.4 Thesis Outline ............................... 5  

2 Literature Review 7  
2.1 Introduction ............................... 7  
2.2 Access Control ............................... 8  
2.2.1 Discretionary Access Control .............. 8  
2.2.2 Mandatory Access Control .................. 8  
2.2.3 Role-Based Access Control .................. 9  
2.2.4 Generalized Role-Based Access Control (GRBAC) .............. 10  
2.2.4.1 GRBAC Policy Rule ....................... 11  
2.2.4.2 GRBAC Hierarchy Semantics ............... 11  
2.2.5 Enterprise Privacy Practices (E-P3P) ................ 13  
2.2.5.1 E-P3P Elements ......................... 13  
2.2.5.2 E-P3P Hierarchy Semantics ............... 14
3 Access Control based Approach

3.1 Introduction .................................................. 26
3.2 A Generalized Policy Support System (GPSS) .................. 27
  3.2.1 Hierarchy ................................................. 27
  3.2.2 A Generalized Policy Support System ................... 30
  3.2.3 GPSS Hierarchy Semantics ............................. 33
  3.2.4 Access Request Evaluation ............................. 34
    3.2.4.1 Matching Rules ................................. 34
    3.2.4.2 Validating $\alpha$ against Hierarchy .......... 35
    3.2.4.3 Result of Access Request Evaluation .......... 39
    3.2.4.4 Scenarios of Using GPSS Hierarchy Semantics .. 39
  3.2.5 Comparison with Other PSSs ........................... 42
    3.2.5.1 Hierarchy Semantics Limitations ............... 42
    3.2.5.2 GPSS’s Generality ............................. 44
  3.2.6 Summary ................................................ 45
3.3 A GPSS based Privacy Protection System ..................... 46
  3.3.1 Privacy Protection System as Integration of Security Systems . 46
    3.3.1.1 Privacy and Security .......................... 46
    3.3.1.2 A Privacy Control Model ...................... 47
### 3.3.2 GPSS based Privacy Protection System

- **3.3.2.1** GPSS, A Suitable Security Model to Use ........................................ 50
- **3.3.2.2** An Instance of GPSS ................................................................. 51
- **3.3.2.3** A Privacy Protection System Equipped with *IGPSS* .................. 53
- **3.3.2.4** An Example .............................................................. 54
- **3.3.2.5** Summary .......................................................... 62

#### 3.4 Chapter Summary .............................................................................. 63

### 4 Hierarchical Encryption based Approach

- **4.1** Introduction ................................................................. 65
- **4.2** A Cryptographic Solution for General Access Control ................. 66
  - **4.2.1** Backgrounds ............................................................. 66
    - **4.2.1.1** Chinese Remainder Theorem ........................................ 67
    - **4.2.1.2** RSA Algorithm .......................................................... 68
  - **4.2.2** A Data Based Solution ................................................. 69
    - **4.2.2.1** Overview ................................................................. 69
    - **4.2.2.2** System Elements ..................................................... 70
    - **4.2.2.3** Cryptographic Access Control ..................................... 70
    - **4.2.2.4** Authorization Alterations ........................................ 71
  - **4.2.3** A Key Based Solution .................................................... 72
  - **4.2.4** Comparison with the *RRN* System ....................................... 74
    - **4.2.4.1** *RRN* System .......................................................... 75
    - **4.2.4.2** Our Solution’s Generality .......................................... 77
    - **4.2.4.3** Performance and Security Analysis ................................ 78
    - **4.2.4.4** Experimental Results ............................................... 79
  - **4.2.5** Summary .......................................................... 80
- **4.3** A Privacy Protection System Based On Hierarchical Encryption ........ 81
  - **4.3.1** System Elements .......................................................... 81
  - **4.3.2** A Customized Hierarchical Encryption Scheme ....................... 82
  - **4.3.3** System Setup ............................................................. 83
  - **4.3.4** Encryption Following Hierarchical Structure ......................... 84
5 Digital Tickets based Approach

5.1 Introduction .................................. 91
5.2 The Advantages of Using Digital Ticket for Privacy Control ............. 92
5.3 A Digital Ticket Privacy Control System .......................... 94
  5.3.1 Participants .................................. 94
  5.3.2 The Structure of the System .......................... 94
  5.3.3 An Encryption Scheme .......................... 96
  5.3.4 Digital Ticket Structure .......................... 97
  5.3.5 The Protocols of Digital Ticket Circulation ..................... 98
5.4 Security Analysis ................................ 100
  5.4.1 Security of Our System .......................... 101
  5.4.2 Security of Ticket Circulation Protocols ..................... 102
5.5 Comparison with the LO System ................................ 102
5.6 Chapter Summary ................................ 103

6 Conclusion .................................... 104

A The Java Source Code for Hierarchical Encryption Experiments ......... 108
  A.1 GenerateTestFiles.java .......................... 109
  A.2 GenerateRSAKeys.java .......................... 110
  A.3 RRNEncrypt.java ................................. 115
  A.4 RRNDecrypt.java ................................. 120
  A.5 DataBasedSolutionEncrypt.java ........................ 125
  A.6 DataBasedSolutionDecrypt.java ........................ 130
  A.7 DataBasedSolutionAdd.java ........................ 136
  A.8 DataBasedSolutionRemove.java ........................ 141
  A.9 KeyBasedSolutionEncrypt.java ........................ 145
  A.10 KeyBasedSolutionDecrypt.java ........................ 149
<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>2.1</td>
<td>XML digital ticket properties and contrast to digital cash</td>
</tr>
<tr>
<td>3.1</td>
<td>Comparison between non-private information system and private information system</td>
</tr>
<tr>
<td>4.1</td>
<td>Performance comparison between $RRN$ and $DBS$</td>
</tr>
<tr>
<td>4.2</td>
<td>Experimental results</td>
</tr>
</tbody>
</table>
List of Figures

2.1 An example of object role hierarchy ........................................ 12
2.2 A general digital ticket model .................................................. 21
3.1 An example of subject hierarchy ................................................ 29
3.2 An example of strict permission inheritance for object hierarchy $TH_1$ 40
3.3 An example of strict permission inheritance for object hierarchy $TH_2$ 40
3.4 An example of lenient permission inheritance for subject hierarchy $GH$ 41
3.5 Hierarchy semantics limitation 1 .................................................. 42
3.6 Hierarchy semantics limitation 2 .................................................. 43
3.7 Hierarchy semantics limitation 3 .................................................. 44
3.8 A privacy control model ........................................................... 48
3.9 Subject hierarchy $GH$ ............................................................... 56
3.10 Object hierarchy $TH$ ............................................................... 56
3.11 Purpose hierarchy $PH$ ............................................................. 57
3.12 Condition hierarchy $CH$ ........................................................ 57
3.13 Obligation hierarchy $OH$ ........................................................ 58
4.1 Three simultaneous congruences sets (SCSs) ................................. 71
4.2 An example of subject hierarchy $GH = (G, \leq_G)$ ......................... 86
5.1 The digital ticket privacy control system ....................................... 95
5.2 Issue protocol ............................................................................. 98
5.3 Cancellation protocol ............................................................... 99
5.4 Transfer protocol ...................................................................... 99
5.5 Consumption protocol ............................................................. 100
Chapter 1

Introduction

1.1 Introduction

Information technology owes its dominance over traditional paper based systems to the ability of information manipulation - to store endlessly, to sort efficiently, to locate effortlessly, and to make decision effectively [88]. Today, information technology is penetrating every aspect of human life. The collection, storage and remote retrieval of vast amounts of personal information have become a routine and inexpensive proposition [24]. Information technology brings people with great convenience. For example, providing credit card information allows shopping online instead of crowding around shopping center. In addition, information technology brings quality to services provided by organizations. For instance, the electronic health records available online make it possible to provide the best treatment to a patient even if he/she is travelling national wide.

However, behind these attractive benefits, the risks of privacy violation are increasing. Easy access to private data increases the temptations for interested parties (individuals, businesses and governments) to intrude upon people’s privacy in unprecedented ways [21]. In recent years, news about privacy breaches continues to hit the headlines. Some of them are cited below.

(1) A computer breach may have exposed more than 32,000 current and former San Diego County employees to financial danger and may have revealed the closely guarded home addresses of some 5,000 law enforcement personnel [29].
(2) The University of San Diego has notified almost 7,800 individuals, including some faculty members, students and vendors, that hackers gained illicit access to computers containing their personal income tax data [20].

(3) Account information of thousands of clients has been compromised after bank employees illegally sold information to a man identified as Orazio Lembo, posing as a collection agent. Police in Hackensack originally estimated the number of affected at 500,000. The number was raised to 676,000 because some people may have more than one account. Hackensack police chief Charles Zisa said the number of breached accounts may top 1 million [36].

(4) 13,000 patients who rented or bought wheelchairs, walkers or other medical equipment from the Olympia Providence Home Services store are at risk for identity theft after the confidential medical records were stolen [75].

Massive privacy breaches have made privacy a central concern in the information age [53]. Numerous surveys (e.g. [1, 113, 114, 115, 116, 117]) conducted around the world have found consistently high levels of concern about privacy [3]. Privacy concerns might seriously hamper user acceptance of information technology [47]. People start considering whether it is worth to risk their privacy to gain benefits offered by information technology. Providing vital personal information, such as credit card information and health records, is often more difficult due to lack of trust between information donors and organizations. This distrust, caused by the privacy concerns of information donors, becomes a serious obstacle to the widespread adoption of information technology. To acquire information donors’ trust, organizations have started to publish privacy policies in order to promise fair personal information practices. Nevertheless, privacy cannot be sufficiently protected solely by legislation. Privacy should also be enforced by information technologies and should become an important design criterion for information and communication systems [40].
1.2 Motivation

This thesis presents three privacy protection approaches based on existing security protection techniques. These approaches are highlighted for their consideration for information donors’ privacy preferences, which are often neglected in previous privacy protection techniques. The motivation of this thesis comes from the following aspects:

(1) *Utilizing information security techniques to protect privacy.*

Protecting information privacy is a sibling of protecting information security, because in both cases the basic problem is how to protect information against unauthorized accesses. This common ground allows us to utilize information security techniques to protect information privacy. The access control techniques, hierarchical encryption techniques and digital tickets techniques are normally considered as security protection techniques. These techniques have been studied extensively for the sake of protecting information security. However, in terms of protecting information privacy, only a few researches have been done by utilizing the above security protection techniques. Utilization of existing security techniques is an optimum way of developing privacy protection techniques. Therefore, the first objective of this thesis is to develop privacy protection mechanisms based on existing security protection techniques.

(2) *Limited control was given to information donors:*

Current privacy protection techniques give individuals limited control over their privacy (This will be further discussed in Chapter 2 of this thesis). This manifests as after an individual submits his/her private data to an organization, he/she almost loses control over it. That is, he/she cannot decide to whom, when and how the data can be accessed and has to rely on the organization to manage it properly. We refer to this problem as *lack of control.* To solve this problem, we need to give individuals control over their private data throughout its lifetime. The lifetime here means the time that starts when the private data is created and ends when the private data is destroyed. Therefore, the second objective of this thesis is to discuss mechanisms to increase individuals’ control over their private data.
No best solution:

Meeting privacy challenge is not trivial. Currently, there is no solution that can solve privacy problems perfectly. Existing privacy protection approaches have their advantages and limitations. Therefore, the third objective of this thesis is to develop several privacy protection approaches that are better than their predecessors.

1.3 Proposed Solution

In this thesis, three privacy protection approaches are proposed and analyzed: an access control based approach, a hierarchical encryption based approach and a digital ticket based approach.

(1) The access control based approach is general and capable of expressing rich data protection policies. In this approach, an individual is given the right to configure his/her own access control system, this will enable him/her to control his/her privacy. One disadvantage of this approach is that the storage of data is assumed to be fully trusted because all private data is stored in plaintext form. This causes a number of problems. First, it does not protect private data against malicious insiders, e.g. a corrupt administrator. Second, it does not protect private data once its storage is compromised.

(2) The hierarchical encryption based approach utilizes a general and efficient hierarchical encryption scheme. In this approach, an individual generates his/her encrypted private data and because of that he/she is capable of control his/her privacy. All private data should be encrypted in a way such that only legitimate users (or groups) can decrypt it. Unauthorized users, including unauthorized administrators, cannot decrypt it. In this approach, the storage of data does not need to be trusted because data is cryptographically protected. However, these protections come at some computational costs.

(3) The digital ticket based approach utilizes digital ticket technique to protect private data. This approach possesses several advantages, e.g. flexibility. In this approach,
the capability of customizing and issuing digital tickets enables an individual to control his/her privacy. The digital ticket technique we use supports many features, such as the number of times a ticket is to be consumed, ticket expiry date and so on. These features make privacy control more elaborate. Furthermore, with digital ticket technique, private data is also protected by encryption. Therefore, the storage of the data does not need to be trusted. However, using of cryptographic mechanisms also makes this approach computationally more intensive.

In summary, our privacy protection approaches utilizes existing security protection techniques. In addition, these approaches increase individuals’ control over their privacy. Individuals are able to specify and maintain their privacy preferences during their private data’s lifetime. That means even after the submission of their private data, individuals are still capable of controlling when, how and by whom the private data is used. Moreover, our approaches have other advantages (e.g. generality, flexibility), which will be detailed in this thesis.

1.4 Thesis Outline

This rest part of this thesis is organized as follows.

Chapter 2 briefly reviews three categories of technologies that are related to protecting private data in information systems: access control, hierarchical encryption and digital tickets. The limitations of previous privacy solutions are also presented.

Chapter 3 presents the privacy protection approach, which is based on access control. The contents of Chapter 3 have been published in Kong, Getta, Yu and Seberry’s papers [66, 67, 71].

(1) First, we propose and analyze a Generalized Policy Support System (GPSS). GPSS is capable of expressing and enforcing rich data protection policies.

(2) Second, we propose and analyze a privacy control model, which consists of a set of security systems. This privacy control model enables us to utilize security mechanisms directly.
(3) Finally, we propose and analyze a GPSS based privacy protection system by integrating GPSS and the privacy control model.

Chapter 4 presents the privacy protection approach, which is based on hierarchical encryption. The contents of Chapter 4 have been published in Kong, Seberry, Getta and Yu’s papers [68, 69].

(1) A new hierarchical encryption scheme is proposed. Compared to most recent hierarchical encryption solutions, our scheme is more general and efficient.

(2) Based on the new hierarchical encryption scheme, we design a privacy control system aimed at protecting private data in information systems.

Chapter 5 presents the privacy protection approach, which is based on digital tickets. The contents of Chapter 5 have been published in Kong, Seberry, Getta and Yu’s paper [70].

(1) We discover the advantages in using digital tickets to manage access to private data.

(2) Inspired by these advantages, we propose a general digital ticket privacy control system based on Fujimura, Nakajima and Sekine’s [43, 44] XML digital ticket technique.

Chapter 6 concludes this thesis.
2.1 Introduction

This chapter reviews the technologies that are related to protecting private data in information systems.

Privacy is a sibling of security. It means that in both cases the basic problem is how to protect information against unwanted accesses. As a result, almost all privacy enhancing technologies are derived from computer security technologies. This is why to review privacy enhancing technologies, we have to also review the relevant computer security technologies. This chapter briefly reviews the following three categories of security/privacy technologies:

(1) Access Control

(2) Hierarchical Encryption

(3) Digital Tickets

For each of the above categories, we introduce its security techniques and its privacy techniques, if any. Besides, we point out their shortcomings related to our problem: protecting private data in information systems.
2.2 Access Control

Access control is the mechanism to ensure the confidentiality, integrity, and availability of information. The fundamental of access control is to protect data against unauthorized access. Whenever a subject tries to access a data object, the access control mechanism checks the rights of the subject and decides to either permit or deny the access. Early research efforts in the area of access control focused on Discretionary Access Control (DAC) and Mandatory Access Control (MAC) models, which are also known as traditional access control models. Within the last decade, Role-Based Access Control (RBAC) has emerged as full-fledged model as mature as traditional MAC and DAC concepts [38]. Recently, Generalized Role-Based Access Control (GRBAC) [89] and Enterprise Privacy Practices (E-P3P) [10] were proposed, which are capable of enforcing more sophisticated data protection policies in an organization.

2.2.1 Discretionary Access Control

Discretionary Access Control (DAC) models govern the access to the information on the basis of the user’s identity and of rules specifying, for each user and each object in the system, the types of access that the user is allowed on the object [25]. The request of a user to access an object is checked against the specified authorizations. The access is granted if there exists an authorization stating that the user can access the object of the specified access type. Otherwise, it is denied. DAC models allow users to grant access rights to other users. For example, the creator of an object is allowed to grant and revoke other users’ access to the object created. Therefore, DAC represents a flexible way to enforce different protection requirements. The access control models proposed in this category are: the Access Matrix model [46, 50, 73], the Take-Grant model [57, 78] and the Action-Entity model [23, 39].

2.2.2 Mandatory Access Control

Mandatory Access Control (MAC) models govern the access of users to the information on the basis of the classification of the objects and subjects of the system [25]. Objects
are passive entities, e.g. data files, records. Subjects are active entities that access the objects, e.g. users, processes. Access classes are associated with every subject and object in the system, and access of a subject to an object is permitted if there exists some relationship, depending on the access type, which is satisfied between the classifications of the subject and the object. The access models proposed in this category are: the Bell and LaPadula model [12, 13, 14, 15], the Biba model [19], the Dion model [35], the Sea View model [33, 34, 79, 80], the Jajodia and Sandhu model [56, 55, 103, 104], and the Smith and Winslett model [106, 107, 118].

2.2.3 Role-Based Access Control

Role-Based Access Control (RBAC) models represent arguably the most important recent innovation in the area of access control [17]. RBAC models are based on the concept of role. A role represents a specific function within an organization and can be seen as a set of actions associated with this function. Under an RBAC model, all authorizations needed to perform a given operation are granted to the role associated with the operation, rather than being granted to users directly. Users are then made members of roles, thereby acquiring the roles’ authorizations. Users’ accesses to objects are mediated by roles. Each user is authorized to play certain roles and, on the basis of roles, he/she can perform accesses to the objects. Because a role groups a number of related authorizations, authorization management is greatly simplified. Whenever a user needs to perform a certain operation, the user only needs to be granted to play a proper role associated with the operation, rather than being directly granted the required authorizations. Also, when a user changes his function within the organization, one only needs to revoke the permission to play the role associated with the function from the user. Complicated authorization revoke operations are no longer needed.

Another important concept of RBAC is a role hierarchy. A role hierarchy is a structure imposed on a set of roles, in accordance with the structure of an organization, which classifies some roles as sub-roles of other roles [101]. Formally, a hierarchy is a partial order defining a seniority relation between roles, whereby senior roles acquire the authorizations of their juniors, and junior roles acquire the user membership of their
seniors [38]. For example, in the healthcare context, a role *Specialist* could contain the roles of *Doctor* and *Intern*. This means that a member of role *Specialist* acquires the permissions associated with the roles *Doctor* and *Intern* without the need to explicitly associate those permissions to the role *Specialist*. Moreover, the roles *Cardiologist* and *Rheumatologist* could each contain the role *Specialist*.

RBAC also supports *Separation of Duty (SoD) constraints* [6, 72]. There are two types of SoD constraints: *Static Separation of Duty* and *Dynamic Separation of Duty*. Static SoD means that a user cannot be authorized for both roles, e.g. role *Teller* and role *Auditor*. Dynamic SoD means that a user cannot act simultaneously in both roles, e.g. role *Teller* and role *Account Holder*.

RBAC models have been widely investigated [37]. A standard has been developed [38] as well as an XML-based encoding of RBAC [18]. Relevant extensions include: the development of administration models [30, 63, 64], the temporal RABC model (TRBAC) [16, 60], the development of security analysis techniques [77], and highly expressive access control models [89]. RBAC models are also supported by commercial DBMSs [95]. Finally, it is worth mentioning that RBAC has been used by some privacy enhancing technologies:

1. RBAC is used as part of products for enterprise privacy management [61].

2. Heard's privacy protection system [51] utilizes RBAC and Access Control Lists (ACLs) to protect patients' privacy in healthcare systems.

### 2.2.4 Generalized Role-Based Access Control (GRBAC)

Generalized Role-Based Access Control (GRBAC), proposed by Moyer and Ahamad [89], is an extension of traditional RBAC. GRBAC is a highly expressive, easy-to-use access control model. It generalizes the classical concept of *role* through the new concepts: *subject role*, *object role* and *environment role*.

1. A subject role is very similar to an RBAC role. It classifies subjects based on any security-relevant characteristics about them. Examples of such characteristics are occupation, position and security clearance level.
(2) An object role classifies objects in a system based on those objects’ security-relevant properties, so that the system can mediate access to the objects based on their properties. Examples of such properties are object type, object size, creation date, and sensitivity level.

(3) An environment role is used to capture security-relevant information about the environment for use in GRBAC policies. Examples of such roles are business hours, Tuesday mornings, CPU load, network load, or even weather conditions.

These concepts are used to structure the subjects (users), objects (data) and environments (conditions). With these new types of roles, GRBAC provides the freedom to structure an access control policy around subjects, objects, environmental conditions, or a combination of all three. In addition, GRBAC is simple because it uses only one grouping primitive: roles. In the following paragraphs, we introduce how GRBAC policy rules are constructed.

2.2.4.1 GRBAC Policy Rule

A GRBAC policy rule is defined as a pair \( \langle \text{transaction}, \text{permission bit} \rangle \), where

(1) A transaction is a tuple of the form \( \langle S, O, E, op \rangle \), where \( S \) specifies a subject role, \( O \) specifies an object role, \( E \) specifies an environment role, and \( op \) specifies an operation. Semantically, it represents an operation in which a subject acting in subject role \( S \) performs operation \( op \) on an object in object role \( O \) under environmental conditions specified by environment role \( E \).

(2) A permission bit is either positive (+) or negative (−). + means its associated transaction is allowed. − means its associated transaction is prohibited.

For example, a policy rule \( \langle \langle \text{All Subjects}, \text{All Objects}, \text{Weekends}, \text{All Ops} \rangle, - \rangle \) means that nobody can access any objects on weekends.

2.2.4.2 GRBAC Hierarchy Semantics

A hierarchy is a partial order on a set of elements that defines the seniority relationship between elements [38]. Hierarchy is not a new concept, it has been extensively studied
in the past. Role hierarchy [59, 86, 87, 101] in role-based access control and hierarchy in Flexible Authorization Framework (FAF) [54] are examples of such studies. **Hierarchy semantics** is an inseparable part of hierarchy that defines rules of authorization propagations.

![Figure 2.1: An example of object role hierarchy](image)

There are three role hierarchies in GRBAC: subject role hierarchy, object role hierarchy, and environment role hierarchy. A role hierarchy is a directed, acyclic graph in which every node represents a role and every edge represents a parent-child relationship between the two connected nodes. Role $R_i$ is a super-role of role $R_j$ if and only if there exists a path originating at node $R_j$ and terminating at node $R_i$. For any role $R$, its **entry set** $ER_R = \{R\} \cup \{R' : R' \text{ is a super-role of } R\}$. A sample object role hierarchy is shown in Figure 2.1. In the figure, an object is a leaf of the hierarchy. For example, **jingle.mp3** is an object that enters two object roles: Classified file and MP3 file.

In GRBAC, **permission inheritance** defines the semantics of role hierarchy. GRBAC provides three basic types of permission inheritance: **standard**, **strict** and **lenient**. To permit a role $R$ to be used in a transaction:

1. **standard** permission inheritance requires that at least one policy rule allows a role in its entry set to participate in the transaction and no policy rules prohibit any role in its entry set from participating in the transaction.
(2) strict permission inheritance requires that at least one policy rule allows each role in its entry set to participate in the transaction and no policy rules prohibit any role in its entry set from participating in the transaction.

(3) lenient permission inheritance requires one policy rule allows a role in its entry set to participate in the transaction and no policy rules prohibit the same role from participating in the transaction.

Three types of permission inheritance give a system flexibility of enforcing access control at different levels of strictness. In summary, GRBAC is capable of creating rich access control policies. Moreover, GRBAC provides an algorithm to enforce the access control policies defined in the model. For details of this algorithm, please refer to Section 4.3 of Moyer and Ahamad’s paper [89].

2.2.5 Enterprise Privacy Practices (E-P3P)

Enterprise Privacy Practices (E-P3P), developed by Ashley, Hada, Karjoth, Powers and Schunter [9, 10], has a well-defined privacy architecture and semantics. It enables an organization to express its privacy policies in E-P3P format and to enforce the policies automatically.

2.2.5.1 E-P3P Elements

An E-P3P privacy policy defines what data users can perform what actions for what purposes on which data categories. An E-P3P privacy policy can be formalized by a pair (Terms, Rules) where Terms defines a terminology used in the policy and Rules defines a set of authorization rules. The terminology Terms consists of six elements (DCH, PH, DUH, A, O, C). The elements DCH, PH, and DUH define the hierarchies of data categories, purposes, and data users. A, O, and C define the sets of actions, obligations, and conditions. Data categories DCH is a pair \((T, \geq_T)\) structured in a tree hierarchy, where \(T\) is the set of data categories and \(\geq_T\) defines a hierarchy relationship between two data categories. \(t \geq_T t'\) means that \(t\) is an ancestor data category of \(t'\) where each parent data category groups its child data categories. Similarly, purpose
and data users are structured in a tree hierarchy $PH = (P, \geq_P)$ and $DUH = (U, \geq_U)$, respectively.

The set Rules is a pair $(Ruleset, dr)$ that contains a set Ruleset of authorization rules that allow or deny an action as well as an element dr that defines the default ruling (permission) of this rule set. Allowed default rulings are $\{+, \emptyset, -, \times\}$ representing “allow”, “don’t care”, “deny”, and “error”. An authorization rule $(i, t, p, u, r, a, \bar{\omega}, \bar{c}) \in Ruleset$ consists of the following eight elements:

- **Precedence** $i \in \mathbb{Z}$ ($\mathbb{Z}$ denotes the set of integers) defines the precedence of the rule. Higher precedence rules unconditionally override lower precedence ones.

- **Data Category** $t \in T$ defines the category of data that may be accessed.

- **Purpose** $p \in P$ defines the purpose for which data may be accessed.

- **Data User** $u \in U$ defines the data user that may use the data.

- **Ruling** $r \in \{+, -\}$ defines the ruling (permission) of the rule. A rule either allows (+) or denies (−) the action.

- **Action** $a \in A$ defines the action that may be performed on the data.

- **Obligations** $\bar{\omega} \in O$ defines the duties that the rule mandates. A rule can contain a set $\bar{\omega}$ of obligations that need to be performed. Examples of obligations are “limited retention,” “log this access”, or “notify data subject”.

- **Conditions** $\bar{c} \in C$ defines the conditions under which the rule is applied. A rule can contain a set of boolean conditions that are combined by logical and. Rules where at least one of the conditions is not satisfied will be ignored.

### 2.2.5.2 E-P3P Hierarchy Semantics

In E-P3P, an access request is processed in the following two steps [10]. The first step creates a set of preliminary authorization rules $PA$. $PA$ is a rule set that contains all policy rules in Ruleset and all policy rules derived from Ruleset by using the hierarchy
2.2. Access Control

semantics defined in this system. The second step assesses the access request according to \( PA \).

The hierarchy semantics of E-P3P is defined as follows:

1. **Down-inheritance:** for each rule \((i, t, p, u, r, a, \bar{o}, \bar{c}) \in PA\), for every \((t', p', u')\) such that \( t' \leq t, p' \leq p, \) and \( u' \leq u \), a tuple \((i, t', p', u', r, a, \bar{o}, \bar{c})\) is added to \( PA \).

2. **Up-inheritance of deny:** for each rule \((i, t, p, u, -, a, \bar{o}, \bar{c}) \in PA\), for every \((t', p', u')\) such that \( t \leq t', p \leq p', \) and \( u \leq u' \), a tuple \((i, t', p', u' -, a, \bar{o}, \bar{c})\) is added to \( PA \).

In this system, if contradicting policy rules coexist, *denies-take-precedence* will be applied to remove contradicting policy rules with lower precedences from \( PA \).

In summary, E-P3P is a fine-grained privacy policy model that enables an organization to define and enforce formalized privacy practices. Therefore, the organization can keep its privacy promises technically and prevent accidental privacy violations. However, to be complete, E-P3P needs a component to enable an individual to control his/her privacy. Suppose a private data owner decides to submit his/her private data to an organization based on the organization's privacy practices and the organization uses E-P3P to protect the private data. After this submission, the owner loses the control over his/her private data and totally relies on the organization to keep its promises. What if the organization changes its privacy practices to another version? What if the owner wants to change his/her privacy preferences over the private data? E-P3P provides no solutions for these questions.

### 2.2.6 Access Control based Privacy Control Techniques

Besides E-P3P, there are many privacy control approaches that utilize access control techniques.

Fischer-Hübner [40, 41] proposed a formal *task-based privacy model* that can be used to technically enforce legal privacy requirements. In this model, a user can access private data if the following three conditions are satisfied. First, it is necessary for the completion of the task. Second, the user is authorized to perform the task. Third, the task’s purpose is accordant to the purpose of the collection of the private data.
2.2. Access Control

Similar to E-P3P, this task-based privacy model lacks the functionality which enables an individual to control his/her privacy.

The Platform for Privacy Preference (P3P) [31, 32, 97] is a protocol designed for privacy protection on the Web. P3P allows web-site owners to express their privacy policies in a standard format. A user can programmatically check against his/her privacy preferences to decide whether to release private data to the web-site. As described above, P3P gives some control to individuals over which private data a web-site should collect. Nevertheless, P3P has two problems. First, there is a lack of privacy enforcement mechanism, i.e. there is no technical mechanism to assure organizations acting according to their stated privacy policies [62, 99]. Once private data is submitted to an organization, its owner can only rely on manual protection by the organization. Second, the support of privacy control for individuals is limited. For example, a private data owner is not able to change his/her privacy preferences over the private data after its submission.

The model of Hippocratic databases [4, 5, 52] takes consumer privacy into account in the way it stores and retrieves information. The model is based on a major tenet of the Hippocratic oath governing doctor-patient relationships. The database negotiates the privacy of information between a user and an organization. The database owners set information storage and retrieval policies which database donors can access or deny [48]. Hippocratic databases enable individuals to control their privacy in the sense that a private data owner has rights to opt-in and opt-out pre-defined privacy preferences options before he/she submits his/her private data. However, it is not very flexible for a private data owner to manage his/her privacy. First, the pre-defined privacy preferences options are limited. Second, the authors of Hippocratic databases have not mentioned how a private data owner could change his/her privacy preferences over the submitted private data.

Heard [51] proposed an access control framework for patient privacy protection based on standardized access roles and an access control list. In his model, a patient has right to grant or deny these standardized access roles. As a consequence, the patient controls accesses to his/her private data. Heard’s patient privacy protection system supports privacy preferences by giving patients rights to grant or deny standardized
access roles to their private data. Compared to above solutions, private data owners
have more flexibility to manage their privacy because at any time they can decide
which group of people is allowed to use which part of their private data. However,
this solution is specially designed for a health care scenario. It is not general enough
as a solution for protecting private data in information systems. Furthermore, the
techniques this solution is based on are traditional RBAC and access control list. As
a result this solution is not as expressive as other privacy control techniques, such as
E-P3P.

2.2.7 Summary

In summary, access control techniques have been naturally applied to those for privacy
control. However, as stated above, no solutions are good enough:

(1) Some solutions do not have the facility to enable an individual to control his/her
privacy, e.g. E-P3P and the task-based privacy model.

(2) Some solutions give limited privacy control rights to individuals, e.g. P3P and
Hippocratic databases.

(3) Some solutions are not general and expressive, e.g. Heard’s patient privacy pro-
tection system.

In this thesis, we propose a general, expressive solution for protecting private data
in information systems. Our solution is based on access control techniques. It enables
an individual to control his/her privacy, even after his/her private data is submitted.
For details of our solution, please refer to Chapter 3.

2.3 Hierarchical Encryption

Hierarchical encryption is first proposed by Akl and Taylor [7, 8] in the early 1980s.
Since then more researches [27, 49, 81, 92, 96, 105, 121] have been dedicated to this
area. In general, hierarchical encryption is based on generating cryptographic keys
for each security class such that the key for a lower level security class depends on
the key for the security class that is higher up in the hierarchy. In other words, data encrypted by a lower level security class’s encryption key can be decrypted by a higher level security class’s decryption key. According to our best knowledge, Ray, Ray and Narasimhamurthi’s system [96] (it is referred to as the *RRN* system throughout the remainder of this thesis) is the most recent development in this area. A brief description of *RRN* system is given below.

### 2.3.1 *RRN* System

In many organizations, hierarchical organization of personnel enforces hierarchical organization of access control policies. The *RRN* system is a cryptographic solution that implements general access control. That is, besides supporting access control policies following the hierarchical structure of an organization, the *RRN* system also supports access control policies that do not follow the hierarchical structure. Furthermore, the *RRN* system is simple and can be easily incorporated in existing systems. *RRN* is superior to the previous models in the following aspects.

1. Most of hierarchical encryption solutions prior to the *RRN* system employ complex cryptographic techniques [7, 8, 27, 81, 92], integrating these into existing systems may not be trivial.

2. Others have undesirable requirements [7, 8, 81]: if a user at a security level wants to access data at lower levels, then all intermediate nodes must be traversed.

3. Furthermore the majority of these works, as expected, do not address the problem of how their mechanisms can be adapted if the access control policy does not adhere to the hierarchical structure.

For technical details of *RRN* system, please refer to Section 4.2.4.1 of this thesis.

### 2.3.2 Summary

The idea of applying hierarchical encryption in privacy control is inspired naturally. Because the intrinsic properties of hierarchical encryption support its application in privacy control.
(1) First, hierarchical encryption is a substitute to RBAC. It is suitable to protect information in a large organization where personnel are organized hierarchically. The most recent development of hierarchical encryption, e.g. RRN system, is more flexible in the sense that it supports general access control.

(2) Second, RBAC does not protect information against insiders, e.g. an administrator. This problem can be solved by using hierarchical encryption.

(3) Third, hierarchical encryption enables an individual to control his/her privacy if the generation of ciphertext is controlled by the individual, i.e. the individual controls who can decrypt the ciphertext.

In this thesis, a new hierarchical encryption scheme is proposed. Compared to RRN system, our scheme is more general and efficient. Based on the new hierarchical encryption scheme, we design a privacy control system aiming at protecting private data in information systems. For details of this system, please refer to Chapter 4.

2.4 Digital Tickets

Analogous to a paper ticket, a digital ticket is a certificate that guarantees certain rights of the ticket owner [43]. There are two primary types of digital tickets techniques:

(1) Kerberos

(2) XML Digital Ticket

The digital tickets technique has been applied in the area of resource access control, e.g. e-commerce and mobile service access control [26, 76, 83, 94, 110]. Moreover, Lategan and Olivier [74] proposed a privacy protection system based on Kerberos system (it is referred to as the LO system throughout the remainder of this thesis).

2.4.1 Kerberos

Kerberos [65, 85, 90, 91, 109] is an authentication service that issues tickets granting an individual access to resources on a network, without the individual having to log on
each time such access is required - a valid ticket is all that is required. Kerberos relies exclusively on conventional encryption, making no use of public-key encryption [108].

The working process of Kerberos is depicted as follows [119]. To obtain services from an application server in a “Kerberized” environment, a client must first obtain a Kerberos ticket from the authentication server. The ticket contains, among other things, a section of data (the name of the server, the name of the client, a time-stamp, a lifetime, and a random session key) encrypted with the server’s secret key. The client presents the ticket to the application server, which can verify the ticket for authenticity. Since the client does not know the server’s secret key, it cannot forge a valid ticket. In addition, the client cannot tamper with the contents of a ticket without being detected, because after decrypted by the server the tampered ticket contents are not readable. The actual procedure for obtaining, storing, and using tickets is divided into two steps:

1. When the user first logs in and enters his password, the client software uses the password to obtain a special ticket known as a Ticket-Granting Ticket (TGT) from the central authentication server.

2. When a user requires access to a Kerberized service, the client software presents the TGT to the Ticket-Granting Server (TGS), which then issues a ticket for that particular service. This service specific ticket is then used to authenticate the actual request for service.

This is done to minimize the number of times the user needs to enter his password. Once a user has a valid TGT, the application software can automatically obtain service-specific tickets without human intervention.

### 2.4.2 XML Digital Ticket

XML digital ticket technique contains a set of mechanisms:

1. A generalized digital ticket definition language [43, 44]

2. A digital ticket circulation control scheme [42]

3. A digital ticket management scheme [83]
A digital ticket is analogous to a paper ticket. It is a certificate that guarantees certain rights of the ticket owner [43]. In order to achieve the same goal of paper ticket, digital ticket technique utilizes a set of cryptographic techniques, such as digital signature and so forth.

The general digital ticket definition given by Fujimura and Nakajima [43] is as follows. Let $I$ be a ticket issuer, $U$ be a user (ticket owner), and $S$ be a service promised to the user. A ticket is defined as $Signed_I(I, S, U)$, where the phrase $Signed_I$ means that the entire block is signed by the issuer’s digital signature.

![Figure 2.2: A general digital ticket model](image)

Figure 2.2 shows a general digital ticket system. Inside this system, there are three participants: ticket issuer, user and service provider. The relationships among them are detailed as follows.

(1) **Ticket acquisition**: to access a service, a user needs to apply for an appropriate ticket from a ticket issuer. The ticket issuer authenticates the user and charges the user for the ticket required. Then the ticket issuer issues a ticket to the user.

(2) **Ticket consumption**: the user presents the ticket to a service provider to acquire services. The service provider checks the ticket. If the ticket is valid, services written on the ticket are delivered to the user.

(3) **Ticket clearance**: after the user consumes the ticket, the ticket issuer settles the payments with the service provider.
### 2.4. Digital Tickets

XML digital ticket has a lot of similarities to digital cash [82, 93, 111]. Table 2.1 lists the properties that XML digital ticket supports as well as a contrast to digital cash. As shown in the table, for digital cash and XML digital ticket, the required levels on (2) anonymity, (4) transferability, (6) divisibility, and (7) persistency are different. In addition, the following three properties in Table 2.1 are important for XML digital ticket, but not required by digital cash.

(11) Machine understandable: the terms and description of the service on a ticket must be objectively understood by both the service provider and consumer, otherwise

<table>
<thead>
<tr>
<th>Properties</th>
<th>Digital Cash</th>
<th>XML Digital Ticket</th>
</tr>
</thead>
<tbody>
<tr>
<td>(1). Secure</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>(2). Anonymous</td>
<td>Yes</td>
<td>(2-1). Untraceable</td>
</tr>
<tr>
<td>(Untraceable)</td>
<td></td>
<td>(2-2). Traceable</td>
</tr>
<tr>
<td>(3). Portable</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>(Physical independence)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>(4). Transferable</td>
<td>Yes</td>
<td>(4-1). Transferable</td>
</tr>
<tr>
<td>(4-1). Transferable</td>
<td></td>
<td>(4-2). Not transferable</td>
</tr>
<tr>
<td>(5). Off-line capable</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>(6). Divisible</td>
<td>Yes</td>
<td>(6-1). Specified times</td>
</tr>
<tr>
<td>(Number of times to be consumed)</td>
<td></td>
<td>(6-2). Only once</td>
</tr>
<tr>
<td>(6-2). Only once</td>
<td></td>
<td>(6-3). Infinite times</td>
</tr>
<tr>
<td>(7). Persistent</td>
<td>Yes</td>
<td>(7-1). Specified period</td>
</tr>
<tr>
<td>(Valid period)</td>
<td></td>
<td>(7-2). Persistent</td>
</tr>
<tr>
<td>(8). Wide acceptability</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>(9). User friendly</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>(10). Monetary freedom</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>(11). Machine understandable</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>(12). State manageable</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>(13). Composable</td>
<td>No</td>
<td>Yes</td>
</tr>
</tbody>
</table>

Table 2.1: XML digital ticket properties and contrast to digital cash
the value of the ticket cannot be determined.

(12) State manageable: some tickets have a payment status, i.e., paid or unpaid, and/or reservation status, e.g., waiting list, reserved, or cancelled. The status may be changed dynamically.

(13) Composable: combining two or more tickets is sometimes required to obtain a service or one ticket may comprise several parts. For example, a travel ticket can comprise an accommodation ticket and a plane ticket.

The technique of XML digital ticket provides a general framework to produce different kinds of digital ticket systems, where all the above mentioned properties are supported.

2.4.3 LO System

LO system utilizes Kerberos tickets to control accesses to private information. The general case of its implementation is stated as follows. For further details, please refer to Lategan and Olivier’s paper [74].

When an individual $i$ wants to send some private data $m$ to an organization $o$, $o$’s privacy policy $P$ is checked by using P3P. $i$ gives $o$ a Ticket-Granting Ticket $T(o, i, P)$ to grant $o$ access to $m$ as described in the intersection between $i$’s own privacy policy and $P$ for a limited time period. The actual private information $m$ is stored in a trusted third party $S$, for round the clock availability. When $o$ needs part of $m$, $o$ presents $T$ and $d$, a description of the required subset of $m$ and optionally $o_2$, another organization who actually requires the access to $m$. $S$ verifies that $P$ is not changed, and sends $p$ a ticket $T_{d,o_2}$, where $o_2 = o$ if $m$ was not supplied to $o_2$ and if allowed by $P$. A ticket cannot be reused, and can only be used by the party it has been issued to. If $o$ has to send private information to another party $o_2$, $o$ has to request another ticket for $o_2$ from $S$ if allowed by $P$. If $o$ has to reuse information, a new ticket can be requested with $T$, unless $T$ has expired.

LO system gives individuals some control over their privacy in the sense that individuals can customize and issue tickets. Nevertheless, LO system has two problems.
First, it requires a trusted third party to store private data. Second, $LO$ system is based on Kerberos, which supports less functionalities than XML digital ticket, e.g., composable, state manageable etc.

### 2.4.4 Summary

Researches show that it is ideal to use digital tickets for rights management in the distributed, distrustful environments [83, 94, 110]. With regard to privacy control, the same environment applies. We discover the following advantages for using digital tickets to manage access to private data.

1. Flexibility
2. Different privacy control strategies
3. Pre-computed access decision
4. Consent/Privacy preference
5. Restricted rights of administrators
6. Proof of compliance
7. Comprehensive control

These advantages will be detailed in Section 5.2 of this thesis. Inspired by these advantages, we propose a general digital ticket privacy control system based on Fujimura et al.’s [43, 44] XML digital ticket technique. In our system, by acquiring the capability of customizing and issuing digital tickets, an individual is able to control his/her privacy. Compared to $LO$ system, our system does not require a trusted third party to store private data. Moreover, instead of using Kerberos tickets, our system utilizes XML digital tickets, which support more functionalities: e.g., compositability, state manageability etc. These features make elaborate privacy control realistic. For details of our digital ticket privacy control system, please refer to Chapter 5.
2.5 Chapter Summary

In this chapter, we briefly review three categories of technologies related to protecting private data in information systems: access control, hierarchical encryption and digital tickets. We also point out the limitations of current solutions.

(1) In access control category, the security technologies that we have reviewed are: Discretionary Access Control (DAC), Mandatory Access Control (MAC), Role-Based Access Control (RBAC) and Generalized Role Based Access Control (GRBAC). The privacy control technologies that we have reviewed are: Enterprise Privacy Practices (E-P3P), Fischer-Hübner’s task based privacy control model, Platform for Privacy Preference (P3P), Hippocratic databases and Heard’s access control framework for patient privacy protection.

(2) In hierarchical encryption category, we reviewed the most recent work in this area: Ray, Ray and Narasimhamurthi’s hierarchical encryption system (RRN system).

(3) In digital ticket category, the security technologies that we have reviewed are: Kerberos and XML digital ticket. The privacy control technology that we have reviewed is: Lategan and Olivier’s privacy information control system (LO system).

In the next chapter, we present our privacy protection approach based on access control mechanisms.
Access Control based Approach

3.1 Introduction

As one of the most popular information safeguarding techniques, access control is widely deployed in information systems. Great efforts have been made in this area over decades. Traditional access control approaches, such as Discretionary Access Control (DAC) and Mandatory Access Control (MAC), have evolved to more flexible and powerful approaches, e.g. Role-Based Access Control (RBAC). Utilizing access control to protect private data in information systems is a natural approach. A few researches have been done in this area: Enterprise Privacy Practices (E-P3P), Fischer-Hübner’s task based privacy control model, Platform for Privacy Preference (P3P), Hippocratic databases and Heard’s access control framework for patient privacy protection are representatives of them (Please refer to Section 2.2 for details).

In this chapter, we present our privacy protection approach, which is based on access control.

(1) First, we propose a Generalized Policy Support System (GPSS) that covers GRBAC and E-P3P. GPSS is capable of expressing and enforcing rich data protection policies.

(2) Second, we propose a privacy control model, which consists of a set of security systems. This privacy control model enables us to utilize security techniques directly.

(3) Finally, we propose a GPSS based privacy protection system by integrating GPSS and the privacy control model.
We advise that this chapter is based on Kong, Getta, Yu and Seberry’s papers [66, 67, 71].

3.2 A Generalized Policy Support System (GPSS)

As the next generation of access control technique, RBAC possesses the following advantages over its predecessors [11, 38, 45, 58, 100, 102]:

(1) As roles represent organizational responsibilities and functions, RBAC directly supports arbitrary, organization-specific security policies.

(2) Security administration is greatly simplified by the use of roles to organize access privileges.

(3) RBAC is “policy-neutral” in the sense that it uses role hierarchies and constraints. A wide range of security policies can be expressed, including traditional DAC and MAC, and user-specific ones.

However, RBAC is still not expressive enough to efficiently and effectively enforce more sophisticated data protection policies in an organization. Recently, more powerful systems have appeared. Generalized Role-Based Access Control (GRBAC) and Enterprize Privacy Practices (E-P3P) are representatives of these systems. GRBAC (see Section 2.2.4) is capable of creating rich access control policies. E-P3P (see Section 2.2.5) enables an enterprise to express its privacy policies in E-P3P format and to enforce them automatically. We shall call these two systems as Policy Support Systems (PSSs) because they are designed for expressing and enforcing data protection policies. In this section, we present a Generalized Policy Support System (GPSS) that covers GRBAC and E-P3P.

3.2.1 Hierarchy

A mathematical structure called hierarchy was defined by Jajodia, Samarati, Sapino, and Subrahmanian [54] as a triple \((X, Y, \leq)\), where \(X\) is the set of primitive entities (e.g. a user, an object), \(Y\) is the set of categories (e.g. a group, an object type), \(\leq\) is
a partial order on \((X \cup Y)\) such that each \(x \in X\) is a \textit{minimal element} of \((X \cup Y)\). An element \(x \in X\) is said to be minimal iff there are no elements below it in the hierarchy, that is iff \(\forall y \in (X \cup Y) : y \leq x \Rightarrow y = x\). This definition is rich enough to capture all hierarchy structures presented in GRBAC and E-P3P. Here, we simplify this definition of hierarchy to a two-entry tuple \((Y, \leq)\).

\textbf{Definition 1:}

A hierarchy \(H\) is a pair \((Y, \leq)\), where:

1. \(Y\) is the set of categories (hierarchy elements), such that a primitive entity is treated as a category of itself, called \textit{primitive category}. A primitive category contains one primitive entity and the name of the category is the same as the name of the primitive entity. For example a primitive entity \textit{James Bond} belongs to a primitive category called \textit{James Bond}.

2. \(\leq\) is a partial order on \(Y\) such that each primitive category in \(Y\) is a minimal element of \(Y\).

\(\square\)

In addition, we define the following two binary relations over \(Y\).

\textbf{Definition 2:}

1. The first binary relation \(<\) describes \textit{descendant-ancestor} relationship between the elements in \(Y\). If \(y_i, y_j \in Y\) and \(y_i < y_j\), \(y_j\) is said to be an ancestor of \(y_i\) and \(y_i\) is said to be a descendant of \(y_j\). The relation \(y_i < y_j\) is interpreted as \(y_i\) is \textit{in the category of} \(y_j\). For an element \(y_i\), a set of all its ancestors is defined as \(A_{set}_{y_i} = \{y_k : y_k \in Y \text{ and } y_i < y_k\}\). And a set of all \(y_i\)'s descendants is defined as \(D_{set}_{y_i} = \{y_k : y_k \in Y \text{ and } y_k < y_i\}\).

2. The second binary relation \(\leq\) describes \textit{descendant-ancestor} or \textit{equivalent} relationships between elements in \(Y\). That is, if \(y_i, y_j \in Y\) and \(y_i \leq y_j\), \(y_j\) is either an ancestor of \(y_i\) or \(y_i\) itself.

\(\square\)
A hierarchy $H = (Y, \leq)$ can be expressed by a directed acyclic graph, in which every node represents an element in $Y$ and every edge represents a parent-child relationship between two elements in $Y$. That is, an edge, which starts from $y_i$ and ends at $y_j$, indicates that $y_j$ is a parent of $y_i$. An example of subject hierarchy is presented in Figure 3.1, where Michael, George and Lucy are primitive categories.

![Figure 3.1: An example of subject hierarchy](image)

Some of the hierarchies used in practice are listed as follows. **Subject hierarchy** $GH = (G, \leq_G)$ where $G$ is a set of user groups (roles), $\leq_G$ defines hierarchy relationships between user groups in $G$. **Object hierarchy** $TH = (T, \leq_T)$ where $T$ is a set of object types, $\leq_T$ defines hierarchy relationships between object types in $T$. **Environment hierarchy** $EH = (E, \leq_E)$ where $E$ is a set of environments, $\leq_E$ defines hierarchy relationships between environments in $E$. **Purpose hierarchy** $PH = (P, \leq_P)$ where $P$ is a set of purposes, $\leq_P$ defines hierarchy relationships between purposes in $P$. 
3.2.2 A Generalized Policy Support System

In this section, we present the formal definition of GPSS.

**Definition 3:**

A Generalized Policy Support System \( GPSS = (H, S, A, R, P) \), where:

1. \( H \) denotes a set of \( n \) hierarchies \( H_1, ..., H_n, n \geq 1 \).

2. \( S \) is a collection of \( m \) sets \( S_1, ..., S_m \), where \( m \geq 0 \). The sets defined in \( S \) provide additional restrictions on PSS, for example the sets of obligations and conditions in E-P3P are instances of such sets. \( S \) is optional and its existence depends on the designer of a PSS, e.g. in GRBAC, \( S \) is absent.

3. \( A \) is a set of actions to be performed on data.

4. \( R \) is a set of authorization types (rulings). \( R = \{+, -, \oplus, \ominus, \otimes\} \), where + means positive authorization, \( - \) means negative authorization, \( \oplus \) means implicit positive authorization, \( \ominus \) means implicit negative authorization, and \( \otimes \) means no authorization. The rulings + and \( - \) are used for explicit authorization assignment through policy rules. The rulings \( \oplus \) and \( \ominus \) are used to indicate derived authorization. The ruling \( \otimes \) is used when none of the above four rulings are applicable.

5. \( P \) is a set of precedences over policy rules. \( P = \mathbb{Z} \), i.e. \( P \) is the set of integers that determines precedence orders over a set of policy rules. The larger number denotes the higher precedence. \( P \) is also optional. The absence of \( P \) means that all policy rules have the same precedence order.

\( \square \)
The elements of the above five parts are used as basic units to form policy rules. The collection of all the policy rules in a GPSS is called policy rule set, denoted as $\Gamma$.

**Definition 4:**

A policy rule $\gamma \in \Gamma$ is a tuple $(\gamma_{H_1}, ..., \gamma_{H_n}, \gamma_{S_1}, ..., \gamma_{S_m}, \gamma_A, \gamma_R, \gamma_P)$ where

1. $\gamma_{H_i} \in H_i$, where $n \geq i \geq 1$.
2. $\gamma_{S_i} \in S_i$, where $m \geq i \geq 0$.
3. $\gamma_A \in A$ is the action entry that specifies the action to be performed.
4. $\gamma_R \in \{+, -\}$ is the ruling entry that specifies either positive or negative authorization.
5. $\gamma_P \in P$ is the precedence of $\gamma$.

□

Giving a policy rule $\gamma = (\gamma_{H_1}, ..., \gamma_{H_n}, \gamma_{S_1}, ..., \gamma_{S_m}, \gamma_A, \gamma_R, \gamma_P)$, we can assign an authorization to the hierarchy elements: $\gamma_{H_1}, ..., \gamma_{H_n}$, respectively. We call an authorization explicitly defined by policy rules *explicit authorization*. This compares with authorizations derived by hierarchy semantics, which is defined as *implicit authorization*.

**Definition 5:**

An authorization of a hierarchy element $y \in Y$ is denoted as $A^i_y$ where $i$ is the index of the authorization. $A^i_y$ is defined as a triple $(ruling, precedence, origin)$, where:

1. $ruling \in R$, this ruling entry can be accessed by $A^i_y.ruling$. 
(2) *precedence* ∈ 𝑃, this precedence entry can be accessed by 𝐴_𝒚_i.precedence. Due to the optional property of 𝑃, the precedence entry of an authorization is also optional.

(3) *origin* ∈ 𝑌, this origin entry can be accessed by 𝐴_𝒚_i.origin. *origin* records the hierarchy element where this authorization comes from. For an implicit authorization, *origin* ≠ 𝒚. For an explicit authorization, *origin* = 𝒚.

□

If an explicit authorization 𝐴_𝒚_i of 𝒚ᵢ ∈ 𝐻 propagates to 𝒚ⱼ ∈ 𝐻, then 𝐴_𝒚_i is converted to an implicit authorization 𝐴_𝒚ⱼ (𝑘 and ℓ are some positive integers) by the following processes:

1. If 𝐴_𝒚_i.ruling = +, then 𝐴_𝒚ⱼ.ruling = ⊕.
2. If 𝐴_𝒚_i.ruling = −, then 𝐴_𝒚ⱼ.ruling = ⊖.
3. 𝐴_𝒚ⱼ.precedence = 𝐴_𝒚_i.precedence.
4. 𝐴_𝒚ⱼ.origin = 𝒚ᵢ.

According to the definition of a policy rule, an *access request* can be defined as follows.

**Definition 6:**

An *access request* 𝛼 is a tuple 𝛼 = (𝛼_{𝑯_1}, ..., 𝛼_{𝑯_𝒏}, 𝛼_{𝑺_₁}, ..., 𝛼_{𝑺_𝐦}, 𝛼_{𝑨}), where:

1. 𝛼_{𝑯ᵢ} ∈ 𝑯ᵢ, where 𝑛 ≥ 𝑖 ≥ 1.
2. 𝛼_{𝑺ᵢ} ∈ 𝑺ᵢ, where 𝑚 ≥ 𝑖 ≥ 0.
3. 𝛼_{𝑨} ∈ 𝑨 is the action entry that specifies the action to be performed.

□
3.2.3 GPSS Hierarchy Semantics

Our interpretation of the concept of hierarchy is such that the relationship between a descendant element and its ancestor element is in the category of. In GPSS, there are two types of authorization propagations (⊕, ⊖, and ⊗ authorizations do not propagate, the term authorization in this section denotes + and − authorizations):

1. Down propagation of authorizations: the common rationale is that when an authorization is applied on an ancestor element (superior category), this authorization may also be applied to its descendant elements (inferior categories) implicitly. As a consequence, authorizations propagate downwards.

2. Up propagation of negative authorizations: the common rationale is that when a negative authorization is applied on a descendant element (inferior category), this authorization may also be applied to its ascendant elements (superior categories). As a result, negative authorizations propagate upwards.

Moreover in order to implement standard, strict and lenient permission inheritances (see Section 2.2.4.2), for a hierarchy $H = (Y, \leq)$ in GPSS, there are three types of hierarchy elements (a hierarchy element is an element in $Y$): standard element, strict element and lenient element. System Security Officer (SSO) has the ability to categorize hierarchy elements according to practical needs. We formally define the above concepts as follows:

**Definition 7:**

For a hierarchy $H = (Y, \leq)$, SSO categorizes hierarchy elements in $Y$ into three sets: the standard element set ($STAN_H$), the strict element set ($STRI_H$), and the lenient element set ($LENI_H$) such that the following properties are satisfied.

1. Exclusiveness: $STAN_H \cap STRI_H = \emptyset$, $STRI_H \cap LENI_H = \emptyset$, and $STAN_H \cap LENI_H = \emptyset$.

2. Completeness: $Y = STAN_H + STRI_H + LENI_H$. 

□
Therefore, every hierarchy element in \( Y \) is of one, and only one type. The details for implementing standard, strict and lenient permission inheritance are described in the step 4 of Section 3.2.4.2.

### 3.2.4 Access Request Evaluation

This section presents the process for GPSS to evaluate an access request \( \alpha = (\alpha_{H_1}, ..., \alpha_{H_n}, \alpha_{S_1}, ..., \alpha_{S_m}, \alpha_A) \), which contains the following steps:

1. Finding *matching rules* for \( \alpha \).
2. Validate \( \alpha \) against \( \alpha_{H_i}, n \geq i \geq 1 \), respectively.

We will depict these steps in the following sections.

#### 3.2.4.1 Matching Rules

Not all policy rules can be used to validate an access request. Many of them are not relevant to the access request, i.e. they contain elements that have no relationship (e.g. equivalent, superior, inferior) with corresponding elements of the access request. For example, given an access request that contains an element \( \alpha_{H_i} = \text{Primary Care Physician} \) (\( H_i \) is shown in figure 3.1), a policy rule that contains the corresponding element \( \gamma_{H_i} = \text{Charge Nurse} \) cannot be used to validate the access request. Therefore, to evaluate an access request, we need to find all relevant policy rules of the access request. In this thesis, we call these policy rules *matching rules* of the access request. The concept of *matching rule* is defined as follows.

**Definition 8:**

For an access request \( \alpha = (\alpha_{H_1}, ..., \alpha_{H_n}, \alpha_{S_1}, ..., \alpha_{S_m}, \alpha_A) \), a policy rule \( \gamma \in \Gamma \) is called a *matching rule* of \( \alpha \) if one of the following property sets is satisfied:

1. Property set 1:
   
   (a) \( \alpha_{H_i} \leq \gamma_{H_i} \), where \( n \geq i \geq 1 \). That is, all hierarchy elements of the policy rules are either superior or equivalent to their corresponding hierarchy elements of the access request.
(b) \( \gamma_{S_i} = \alpha_{S_i} \), where \( m \geq i \geq 0 \). That is, all set elements of the policy rules are equivalent to their corresponding set elements of the access request.

(c) \( \gamma_A = \alpha_A \). The actions of the policy rule and access request are equivalent.

(2) Property set 2:

(a) There exists at least one \( \alpha_{H_i} \) such that \( \gamma_{H_i} < \alpha_{H_i} \). And for all other \( \alpha_{H_j} \), \( \alpha_{H_j} \leq \gamma_{H_j} \) holds, where \( n \geq i, j \geq 1 \) and \( i \neq j \). That is, at least one hierarchy element of the policy rule is inferior to its corresponding hierarchy element of the access request and all the rest hierarchy elements are either superior or equivalent to their corresponding hierarchy elements of the access request.

(b) \( \gamma_{S_i} = \alpha_{S_i} \), where \( m \geq i \geq 0 \). That is, all set elements of the policy rules are equivalent to their corresponding set elements of the access request.

(c) \( \gamma_A = \alpha_A \). The actions of the policy rule and access request are equivalent.

(d) \( \gamma_R = - \). The ruling of the policy rule is -.

\[ \square \]

The above two property sets reflect the GPSS hierarchy semantics: down propagation of authorizations corresponds to property set 1 and up propagation of negative authorizations corresponds to property set 2. To evaluate \( \alpha \), the matching rule set \( \Gamma_\alpha \) is used, where \( \Gamma_\alpha = \{ \gamma | \gamma \in \Gamma \text{ and } \gamma \text{ is a matching rule of } \alpha \} \). Therefore, the first step to evaluate \( \alpha \) is to find all matching rules of \( \alpha \) and form the matching rule set \( \Gamma_\alpha \).

### 3.2.4.2 Validating \( \alpha \) against Hierarchy

The next step is to validate hierarchical elements in \( \alpha \): \( \alpha_{H_1}, \alpha_{H_2}, ..., \alpha_{H_n} \). That is, \( \forall \alpha_{H_i} \in \alpha \) where \( n \geq i \geq 1 \), \( \alpha_{H_i} \) needs to be validated according to the matching rule set \( \Gamma_\alpha \) whether \( \alpha_{H_i} \) is authorized for \( \alpha_A \). If and only if all of these hierarchy elements are authorized for \( \alpha_A \), \( \alpha \) is granted.

To validate \( \alpha_{H_i} \) against the hierarchy \( H_i \), the following processes are carried out:

(1) According to \( \Gamma_\alpha \), explicitly assign authorizations to the hierarchy elements in \( H_i \).
(2) For those hierarchy elements that have inequable authorizations (see Definition 9), resolve the conflicts. After conflict resolution, there should be at most one authorization for each hierarchy element in $H_i$ (such authorization is called effective authorization).

(3) Propagate effective authorizations to $\alpha_{H_i}$.

(4) Resolve conflicts and derive the final authorization for $\alpha_{H_i}$.

In the following paragraphs, we will depict the above processes in detail.

**Step 1. Explicit Assignment of Authorizations**

For each rule $\gamma \in \Gamma_\alpha$, where $\gamma = (\gamma_{H_1}, ..., \gamma_{H_n}, \gamma_{S_1}, ..., \gamma_{S_m}, \gamma_A, \gamma_R, \gamma_P)$, we assign an authorization $(\gamma_R, \gamma_P, \gamma_{H_i})$ explicitly to $\gamma_{H_i}$, $n \geq i \geq 1$.

Here is an example of assigning explicit authorizations, assume $\Gamma_\alpha = \{\gamma_1, \gamma_2\}$, where $\gamma_1 = (..., \gamma_{1H_i}, ..., \text{read}, +, 1)$, $\gamma_2 = (..., \gamma_{2H_i}, ..., \text{read}, -, 2)$, and $\gamma_{1H_i} = \gamma_{2H_i} = y \in H_i$. Then $A^1_y = (+, 1, y), A^2_y = (-, 2, y)$.

**Step 2. Working-out Effective Authorizations**

**Definition 9:**

For a hierarchy element $y$, two authorizations $A^j_y, A^k_y$ are equal, denoted by $A^j_y = A^k_y$, if all properties below are satisfied:

1. $A^j_y.ruling = A^k_y.ruling$
2. $A^j_y.precedence = A^k_y.precedence$

For a hierarchy element $y$, two authorizations $A^j_y, A^k_y$ are called inequable authorizations, if $A^j_y = A^k_y$ does not hold.

When a hierarchy element has inequable authorizations, conflicts arise. Our system provides methods of conflicts resolution, called authorization precedence policy.
Definition 10:

The *authorization precedence policy* is composed of the following rules:

1. An authorization with higher precedence overrides authorizations with lower precedences.

2. If the precedences are the same, *denies-take-precedence* will apply. The priority order is: $-, \ominus, +, \oplus$. An authorization with higher priority order overrides authorizations with lower priority orders.

After explicit assignment of authorizations, we need to work out effective authorizations before authorization propagation. For each hierarchy element $y \in H_i$ that has at least one explicit authorization, $y$’s effective authorization is derived as follows.

1. If $y$ has only one authorization $A^1_y$, then the effective authorization for $y$ is:
   $$A^e_y = A^1_y.$$  

2. If $y$ has more than one authorization $A^1_y, A^2_y, \ldots, A^k_y$ and they are all equal, then the effective authorization for $y$ is: $A^e_y = A^1_y$.

3. If $y$ has more than one authorization $A^1_y, A^2_y, \ldots, A^k_y$ and they are not equal, then authorization precedence policy is used to resolve the conflicts. Assume $A^j_y$ is the result of the conflict resolution (i.e., $A^j_y$ is the authorization with the highest precedence and priority order), $y$’s effective authorization is: $A^e_y = A^j_y$.

Step 3. Authorization Propagation

In this step, all effective authorizations in $H_i$, excluding the effective authorization of $\alpha_{H_i}, A^e_{\alpha_{H_i}}$ (if it exists), propagate to $\alpha_{H_i}$.

An effective authorization $A^e_y$ propagates to $\alpha_{H_i}$ and becomes an implicit authorization $A^j_{\alpha_{H_i}}$, via the following processes:

1. if $A^e_y.ruling = +$, then $A^j_{\alpha_{H_i}}.ruling = \oplus$.  

(2) if \( A_y^e,\text{ruling} = - \), then \( A_{\alpha_{H_i}}^i,\text{ruling} = \ominus \).

(3) \( A_{\alpha_{H_i}}^i,\text{precedence} = A_y^e,\text{precedence} \).

(4) \( A_{\alpha_{H_i}}^i,\text{origin} = y \).

**Step 4. Deriving the Final Authorization for \( \alpha_{H_i} \)**

After authorization propagation, the last step is to evaluate the final authorization for \( \alpha_{H_i} \): \( A_{\alpha_{H_i}}^f \).

As stated in Section 3.2.3, the hierarchy elements in the hierarchy \( H_i \) are classified into three exclusive sets: \( \text{STAN}_{H_i} \), \( \text{STRI}_{H_i} \) and \( \text{LENI}_{H_i} \). \( A_{\alpha_{H_i}}^f \) is evaluated via three steps: strict permission check, lenient permission check and standard permission check. The strict permission check makes sure that all strict hierarchy elements in \( \text{Aset}_{\alpha_{H_i}} \cup \{\alpha_{H_i}\} \cup \text{Dset}_{\alpha_{H_i}} \) are explicitly authorized for the access. The lenient permission check is performed if and only if there exists more than one lenient hierarchy element in \( \text{Aset}_{\alpha_{H_i}} \cup \{\alpha_{H_i}\} \cup \text{Dset}_{\alpha_{H_i}} \). The lenient permission check removes all negative authorizations whose origins are lenient hierarchy elements. The standard permission check uses authorization precedence policy to resolve conflicts. This process is depicted as follows.

(1) **Strict permission check:**

(a) If \( \forall y \in (\text{Aset}_{\alpha_{H_i}} \cup \{\alpha_{H_i}\} \cup \text{Dset}_{\alpha_{H_i}}) \cap \text{STRI}_{H_i} \), there exists an authorization \( A_{\alpha_{H_i}}^j \) such that \( A_{\alpha_{H_i}}^j,\text{origin} = y \) and \( A_{\alpha_{H_i}}^j,\text{ruling} \in \{+\} \), then continue with the next check: lenient permission check.

(b) Else \( A_{\alpha_{H_i}}^j = (-, \infty, \alpha_{H_i}) \), skip the other two checks (the infinite symbol \( \infty \) denotes the highest precedence).
(2) Lenient permission check:

(a) If \( \exists A^j_{\alpha_{H_i}}.origin, A^k_{\alpha_{H_i}}.origin \in \text{LENI}_{H_i} \) and \( A^j_{\alpha_{H_i}}.origin \neq A^k_{\alpha_{H_i}}.origin \), remove all authorizations \( A^\ell_{\alpha_{H_i}} \), where \( A^\ell_{\alpha_{H_i}}.origin \in \text{LENI}_{H_i} \) and \( A^\ell_{\alpha_{H_i}}.ruling \in \{-, \oplus\} \).

(b) Continue with the next check: standard permission check.

(3) Standard permission check:

(a) For all remaining authorizations of \( \alpha_{H_i} \), use the authorization precedence policy to resolve conflicts. Assume \( A^j_{\alpha_{H_i}} \) is the result of conflict resolution (i.e., \( A^j_{\alpha_{H_i}} \) is the authorization with the highest precedence and priority order), \( A^f_{\alpha_{H_i}} = A^j_{\alpha_{H_i}} \).

### 3.2.4.3 Result of Access Request Evaluation

As mentioned, the validation of \( \alpha \) in \( \Gamma_\alpha \) consists of \( n \) sub-validations from \( \alpha_{H_1} \) to \( \alpha_{H_n} \). That is, for \( \forall \alpha_{H_i} \in \alpha \) where \( n \geq i \geq 1 \), \( \alpha_{H_i} \) needs to be validated according to the matching rule set \( \Gamma_\alpha \) whether \( \alpha_{H_i} \) is authorized for \( \alpha_A \). The evaluation result of \( \alpha \) is decided as follows.

1. If \( \forall A^f_{\alpha_{H_i}}.ruling \in \{+, \oplus\} \), where \( n \geq i \geq 1 \), then \( \alpha \) is approved.
2. Else \( \alpha \) is denied.

### 3.2.4.4 Scenarios of Using GPSS Hierarchy Semantics

This section shows some examples of using the GPSS hierarchy semantics. In these examples, we will show how strict permission inheritance restricts access and how lenient permission inheritance increases access.

Figure 3.2 shows an example of strict permission inheritance. In this example, SSO wants to be strict with regard to accesses to elements entering category \( y_4 \) because \( y_4 \) is \textit{Classified file}. SSO defines \( STRI_{TH_1} = \{y_4\} \). Then to be able to access an element \( y \in Dset_{y_4} \cup \{y_4\} \), one prerequisite is that \( y_4 \) must be explicitly granted access. Assume there is an access request \( \alpha \) that requires to access \( y_8 \). The category \( y_8 \) is a descendant
3.2. A Generalized Policy Support System (GPSS)

Figure 3.2: An example of strict permission inheritance for object hierarchy $TH_1$ of $y_4$. According to $\Gamma_\alpha$, only $y_4$ has an explicit authorization $(+, 1, y_4)$ (as shown in Figure 3.2). This authorization propagates downwards to $y_8$. After permission check and conflict resolution, $A^f_{y_8} = (\oplus, 1, y_4)$. In this case, $A^f_{y_8}.ruling = \oplus$.

Figure 3.3: An example of strict permission inheritance for object hierarchy $TH_2$

Figure 3.3 shows another example of strict permission inheritance. In this example, SSO wants to be strict to accesses to elements entering category $y_3$ and $y_4$ because $y_3$ is Sensitive information and $y_4$ is Private information. SSO defines $STRI_{TH_2} = \{y_3, y_4\}$. 
3.2. A Generalized Policy Support System (GPSS)

Then to be able to access an element \( y \in A_{y_3} \cup \{y_3\} \cup D_{y_3} \cup A_{y_4} \cup \{y_4\} \cup D_{y_4} \), one prerequisite is that \( y_3, y_4 \) or both \( y_3 \) and \( y_4 \) must be explicitly granted access. Assume there is an access request \( \alpha \) that requires to access \( y_8 \). \( y_8 \) is a descendant of \( y_3 \) and \( y_4 \). According to \( \Gamma_\alpha \), \( y_3 \) has an explicit authorization \((+1,y_3)\) and \( y_4 \) has an explicit authorization \((+2,y_4)\) (as shown in Figure 3.3). These two authorizations propagate downwards to \( y_8 \). After permission check and conflict resolution, \( A_{y_8}^f = (\oplus,2,y_4) \). In this case, \( A_{y_8}^f.ruling = \oplus \).

![Figure 3.4: An example of lenient permission inheritance for subject hierarchy GH](image)

An example of lenient permission inheritance is shown in Figure 3.4. A primitive category \( y_8 \) (George) enters both \( y_4 \) (Primary Care Physician) and \( y_5 \) (Specialist Physician). In order to let George work for both roles, SSO defines \( LENI_{GH} = \{y_4,y_5\} \). Because to be able to work for both roles, George must get maximum rights from the two roles. For example, a right allowed by \( y_4 \) and prohibited by \( y_5 \) should be allowed to George. Assume there is an access request \( \alpha \) that requires to access \( y_8 \). \( y_8 \) is a descendant of \( y_4 \) and \( y_5 \). According to \( \Gamma_\alpha \), \( y_4 \) has an explicit authorization \((+1,y_4)\) and \( y_5 \) has an explicit authorization \((-1,y_5)\) (as shown in Figure 3.4). These two authorizations propagate downwards to \( y_8 \). After permission check and conflict resolution, \( A_{y_8}^f = (\oplus,1,y_4) \). In this case, \( A_{y_8}^f.ruling = \oplus \).
3.2.5 Comparison with Other PSSs

PSSs are capable of expressing and enforcing rich data protection policies. GRBAC and E-P3P are representatives of such systems. Here, we present a comparison between our GPSS and the above two PSSs. The comparison results reveal the superiority of GPSS.

3.2.5.1 Hierarchy Semantics Limitations

We have introduced the hierarchy semantics of GRBAC (see Section 2.2.4.2) and E-P3P (see Section 2.2.5.2). In this subsection, we analyze the limitations of their hierarchy semantics.

![Hierarchy semantics limitation 1](image)

Figure 3.5: Hierarchy semantics limitation 1

(1) The rule of GRBAC’s strict permission inheritance is very restrictive. The original intention of GRBAC’s strict permission inheritance is to restrict accesses to the elements in the category of sensitive/vulnerable categories defined by the System Security Officer (SSO). However it is too strict to be practical. Consider the object hierarchy $TH$ shown in Figure 3.5. Assume there is a request $\alpha_1$ that requires to access $y_8$. According to $\Gamma_{\alpha_1}$, $y_8$’s parents $y_4$ and $y_7$ have explicit authorizations. Following the rule of GRBAC’s strict permission inheritance, $y_8$ is prohibited from being accessed by $\alpha_1$. This is because GRBAC’s strict permission inheritance
requires the requested element \( (y_8) \) and all its ancestors \( (y_4, y_7, y_1, y_5 \text{ and } y_2) \) to be explicitly authorized for the access. That is, GRBAC treats all hierarchy elements in \( TH \) as sensitive/vulnerable categories. Obviously this is too strict. If a system deploys GRBAC’s strict permission inheritance, few access requests can be granted. In contrast with GRBAC, GPSS has the flexibility of defining sensitive/vulnerable categories in a hierarchy. In the example above, GPSS only defines the category \textit{Classified file} \((y_4)\) as a sensitive/vulnerable category. Then according to \( \Gamma_{\alpha_1} \), \( y_8 \) is allowed to be accessed by \( \alpha_1 \).

(2) GRBAC does not consider descendants’ negative authorizations. When evaluating an element \( y_i \) in a hierarchy \( H = (Y, \leq) \), GRBAC only evaluates authorizations in \( A_{y_i} \cup \{y_i\} \). The neglect of authorizations in \( D_{y_i} \) results in incorrect authorization evaluations when there exist negative authorizations in \( D_{y_i} \). The rationale behind is that if a part of \( y_i \) is prohibited, \( y_i \) is prohibited as well. Consider the object hierarchy \( TH \) shown in Figure 3.6. Assume there is a request \( \alpha_2 \) that requires to access \( y_2 \). According to \( \Gamma_{\alpha_2} \), \( y_2 \) has an explicit positive authorization \((+, 1, y_2)\) and \( y_5 \) has an explicit negative authorization \((-1, y_5)\). Following GRBAC’s standard permission inheritance, \( y_2 \) is allowed to be accessed by \( \alpha_2 \) because \( y_5 \)’s negative authorization is neglected. As a comparison, to decide the final authorization of \( y_i \), GPSS considers authorizations in \( A_{y_i} \cup \{y_i\} \) and negative
3.2. A Generalized Policy Support System (GPSS)

authorizations in \( Dset_y \). In the example above, according to \( \Gamma_{\alpha_2} \), \( y_2 \) is prohibited from being accessed by \( \alpha_2 \).

![Figure 3.7: Hierarchy semantics limitation 3](image)

(3) E-P3P only supports one type of permission inheritance: standard permission inheritance. As a consequence, E-P3P is not flexible in practice. Consider the object hierarchy \( TH \) shown in Figure 3.7. Assume there is a request \( \alpha_3 \) that requires to access \( y_8 \). According to \( \Gamma_{\alpha_3} \), only \( y_7 \) has an explicit positive authorization \((+, 1, y_2)\). In this case, E-P3P permits \( y_8 \) being accessed by \( \alpha_3 \). E-P3P has no ability to identify any sensitive/vulnerable categories, such as \textit{Classified file}. In contrast with E-P3P, GPSS supports all three types of permission inheritances. In the example above, GPSS defines the category \textit{Classified file} \((y_4)\) as a sensitive/vulnerable category. Then according to \( \Gamma_{\alpha_3} \), \( y_8 \) is prohibited from being accessed by \( \alpha_3 \).

3.2.5.2 GPSS’s Generality

GPSS is a general model that “includes” GRBAC and E-P3P. This has been proven by the following facts. First, GPSS’s definition is general. Second, GPSS’s hierarchy semantics is general.

The GPSS’s definition (see Definition 3 of this chapter) covers the definitions of GRBAC and E-P3P. A GRBAC system is a triple \((\mathcal{H}, A, R)\), where \( \mathcal{H} = \{GH, TH, EH\} \) \((GH \) is subject hierarchy, \( TH \) is object hierarchy and \( EH \) is environment hierarchy).
A GRBAC policy rule is a tuple \((S, O, E, op, \text{permission bit})\), where \(S \in GH, O \in TH, E \in EH, op \in A\) and permission bit \(\in \{+, -\}\). There is no precedence over GRBAC policy rules, hence the set \(P\) is absent. An E-P3P system is a tuple \((H, S, A, R, P)\), where \(H = \{GH, TH, PH\}\) (\(GH\) is subject hierarchy, \(TH\) is object hierarchy and \(PH\) is purpose hierarchy), \(S = \{O, C\}\) (\(O\) is the set of obligations and \(C\) is the set of conditions). An E-P3P policy rule is a tuple \((i, t, p, u, r, a, o, c)\), inside which \(i \in P, t \in TH, p \in PH, u \in GH, r \in \{+, -\}, a \in A, o \in O\) and \(c \in C\).

As discussed in the Section 3.2.5.1, GPSS solves the hierarchy semantics limitations encountered in GRBAC and E-P3P. In addition, GPSS’s hierarchy semantics generalizes the hierarchy semantics of GRBAC and E-P3P. For a hierarchy \(H = (Y, \leq)\),

1. GRBAC’s standard permission inheritance is a special case of GPSS’s where \(STAN_H = Y\),
2. GRBAC’s strict permission inheritance is a special case of GPSS’s where \(STRI_H = Y\),
3. GRBAC’s lenient permission inheritance is a special case of GPSS’s where \(LENI_H = Y\),
4. E-P3P’s hierarchy semantics is a special case of GPSS’s where \(STAN_H = Y\).

### 3.2.6 Summary

We have presented a Generalized Policy Support System (GPSS). Our GPSS is composed by the following elements: hierarchies, sets, actions, rulings, and precedences. The hierarchy semantics of GPSS is general and flexible. GPSS is capable of expressing and enforcing rich data protection policies. In the last part of this section, a comparison with previous PSSs (GRBAC and E-P3P) has been provided. The comparison results illustrate the superiority of GPSS. GPSS forms a solid base for developing an access control based privacy protection system. In the next section, we propose a privacy control model that combines a set of security systems. Then a set of GPSSs are used to compose a GPSS based privacy protection system.
3.3 A GPSS based Privacy Protection System

Our strong intuition is that privacy is a sibling of security. It means that in both cases the basic problem is how to protect information against unauthorized accesses. Large numbers of security techniques have already been developed and implemented. An interesting problem is how to reuse these techniques to protect privacy. A sound solution to this problem would allow for a quick and relatively inexpensive implementation of a privacy protection system. In this section, we investigate a method to develop a privacy protection system by directly exploiting existing security techniques. Analysis of security and privacy requirements leads to a new privacy control model. The model derives a privacy protection system from a collection of security systems. Based on the privacy control model, we propose a privacy protection system that contains a set of GPSS security systems.

3.3.1 Privacy Protection System as Integration of Security Systems

By comparing privacy with security, we will propose a new privacy control model that consists of a set of security systems.

3.3.1.1 Privacy and Security

Security is regarded as a balance between confidentiality, integrity and availability. Westin [112] defined privacy as the right for individuals to determine for themselves when, how and to which extent information about them can be exposed to others.

Privacy is a sibling of security. It means that in both cases, the basic problem is how to protect information against unwanted accesses. The concepts of privacy and security are intricately linked together. It is not easy to clearly distinguish them from each other. We utilize a set of contrasts between privacy and security to present our understanding about privacy.
Table 3.1: Comparison between non-private information system and private information system

<table>
<thead>
<tr>
<th>System Element</th>
<th>Non-private information system</th>
<th>Private information system</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data ownership</td>
<td>Owned and deposited by organization</td>
<td>Owned by information donor, deposited by organization</td>
</tr>
<tr>
<td>The beneficiary of stored data</td>
<td>Organization</td>
<td>Organization and information donor</td>
</tr>
<tr>
<td>Person (party) to be impacted by unauthorized leakage of data</td>
<td>Organization</td>
<td>Organization and information donor</td>
</tr>
<tr>
<td>Access control decision maker</td>
<td>Organization</td>
<td>Decided by information donor and organization</td>
</tr>
</tbody>
</table>

Privacy concerns arise in an information system when it contains personal information (private data). Personal information is information related to an individual whose identity is apparent or can reasonably be ascertained from the information [2]. We call such information system \textit{private information system}. An information system that contains no private data does not bear privacy problems. Such system is called \textit{non-private information system}. Table 3.1 provides a comparison between a non-private information system and a private information system. It illustrates five aspects of differences lying under security and privacy. The most essential one is that in a private information system, an access control decision should be made by information donor and organization, in order to conform to both information donor’s privacy preferences and organization’s privacy control policies.

Obviously, to control privacy for a private information system, more issues need to be taken into account than to control security for a non-private information system.

\subsection{A Privacy Control Model}

If we treat every information donor as a sub-organization, then we can treat an organization’s private information system (IS) as a collection of such sub-organization’s
3.3. A GPSS based Privacy Protection System

Figure 3.8: A privacy control model

information systems. Following this idea, we obtain a logic structure of a private information system (see Figure 3.8). By virtually dividing a private information system into pieces based on data ownership, we have a clear view of who is responsible for which part of the information system. The privacy protection system is a combination of an organization’s security system and a set of information donors’ security systems (see Figure 3.8).

Therefore, our privacy control model is formally defined as follows.

**Definition 11:**

Given a private information system held by an organization that contains private data contributed by \( n \) information donors: \( \{d_1, d_2, ..., d_n\} \). The privacy control model is a collection of the following security systems:

1. \( SEC_{org} \): the security system specified by the organization, which is aimed at protecting the whole information system. This security system reflects the organization’s privacy control policies.
(2) A set $\text{DONORSEC} = \{SEC_{d_1}, SEC_{d_2}, ..., SEC_{d_n}\}$ which contains $n$ security systems, each is specified by an information donor and is aimed at protecting the information donor’s private data. These security systems reflect privacy preferences of information donors.

How to define these $n + 1$ security systems remains an interesting issue. The privacy protection system presented here uses $n + 1$ homogeneous security systems: each of them is a GPSS system. However, they could be different systems, e.g. $SEC_{org}$ uses Role-Based Access Control (RBAC), while $SEC_d$ uses Mandatory Access Control (MAC). Heard’s privacy control system for electronic health records [51] is an instance of this privacy control model, where $SEC_{org}$ uses RBAC and $SEC_d$ uses Access Control Lists (ACLs), for more details, please refer to Section 2.2.6.

The advantages of this privacy control model are as follows:

(1) Increased privacy control for individuals: the model provides platforms for information donors to manage their privacy. For an information donor $d$, to manage his/her privacy, $d$ needs to configure the corresponding security system $SEC_d$.

(2) Applicability and flexibility:

(a) For any information system storing private data, this model is applicable.

(b) This privacy model is very flexible because various security techniques may be used.

(3) Linking security and privacy together seamlessly.

(4) Making privacy protection system easily implementable by directly utilizing current security techniques, because:

(a) Security techniques have evolved which are much more mature than those for privacy.

(b) The similarities between security and privacy support this utilization.
3.3. A GPSS based Privacy Protection System

Here is a scenario that shows how this privacy control model works. If a user wants to access a piece of private data that belongs to an information donor $d_i$, then this access needs to be validated by two security systems: $SEC_{org}$ and $SEC_{d_i}$. The user can access the private data if and only if both $SEC_{org}$ and $SEC_{d_i}$ permit this access.

We have presented a privacy control model based on security. By doing so, we can utilize security techniques directly to solve privacy problems. Now we will discuss a privacy control model that equips Generalized Policy Support System (GPSS) - a GPSS based privacy protection system.

### 3.3.2 GPSS based Privacy Protection System

The privacy control model proposed in Section 3.3.1.2 is just a framework for a privacy protection system. In this section, we enrich our privacy control model by exploiting current security techniques. We choose a suitable security technique to equip the privacy control model, and then make this model practical and ready to solve privacy problems.

#### 3.3.2.1 GPSS, A Suitable Security Model to Use

As introduced in Section 3.2, GPSS is general and capable of expressing and enforcing rich data protection policies. We use GPSS for its expressiveness and simplicity. From privacy point of view, an information donor needs to decide his/her privacy preferences, which contain:

1. Who can access?
2. Which private data can be accessed?
3. For what purposes can access be permitted?
4. What conditions must be satisfied before access?
5. What obligations must be carried out after access?
6. How to access?
We find that the formation of privacy preferences can be easily expressed by a GPSS policy rule. Therefore, GPSS can be used to express both organizations’ privacy policies and information donors’ privacy preferences. By customizing GPSS, we derive an instance of GPSS as the security system, which is then used to construct the privacy protection system.

### 3.3.2.2 An Instance of GPSS

Here, we define an instance of GPSS: $IGPSS$, which will be used as a component in our privacy protection system.

**Definition 12:**

$$IGPSS = (H, A, R),$$

where:

1. $H = \{GH, TH, PH, CH, OH\}$
   - (a) $GH$ is subject hierarchy. $GH = (G, \leq_G)$
   - (b) $TH$ is object hierarchy. $TH = (T, \leq_T)$
   - (c) $PH$ is purpose hierarchy. $PH = (P, \leq_P)$
   - (d) $CH$ is condition hierarchy. $CH = (C, \leq_C)$
   - (e) $OH$ is obligation hierarchy. $OH = (O, \leq_O)$

2. $A$ is the set of actions to be performed on data, such as read, write etc.

3. $R$ is the set of authorization types (rulings): $R = \{+, -, \oplus, \ominus\}$. 

The set of all policy rules in an IGPSS is denoted by Γ. A policy rule \( \gamma \in \Gamma \) is defined as follows.

**Definition 13:**

\[ \gamma = (g, t, p, c, o, a, r) \]

(1) \( g \in G \) defines a subject.

(2) \( t \in T \) defines an object.

(3) \( p \in P \) defines a purpose.

(4) \( c \in C \) defines a condition.

(5) \( o \in O \) defines an obligation.

(6) \( a \in A \) defines an action.

(7) \( r \in R \) defines a ruling.

According to the definition of a policy rule, an access request \( \alpha \) in IGPSS is defined as follows.

**Definition 14:**

\[ \alpha = (g, t, p, c, o, a) \]

(1) \( g \in G \) defines a subject.

(2) \( t \in T \) defines an object.

(3) \( p \in P \) defines a purpose.

(4) \( c \in C \) defines a condition.
(5) \( o \in O \) defines an obligation.

(6) \( a \in A \) defines an action.

\[ \square \]

The above paragraphs describe an instance of GPSS: \( IGPSS \). \( IGPSS \) is specially designed for expressing privacy control policies and privacy preferences. Of course, according to actual demands, \( IGPSS \) could be customized to another form different from the above.

3.3.2.3 A Privacy Protection System Equipped with \( IGPSS \)

As mentioned in Section 3.3.1.2, our privacy protection system contains \( n + 1 \) security systems: \( SEC_{d_1} \), \( SEC_{d_2}, \ldots, SEC_{d_n} \), and \( SEC_{org} \). If we equip \( IGPSS \) into these security systems, then every security system is an \( IGPSS \) policy rule set.

The access control decision-making procedure is discussed as follows. An access request is approved if and only if it is approved by both the organization’s security system and all relevant information donor’s security systems. For simplicity, we assume that every access requests only one information donor’s private data.

Consider the following scenario. A user \( u \) requests access to information donor \( d_i \)’s private data. The access control procedure is that the system checks if it is approved by \( SEC_{org} \) and \( SEC_{d_i} \), respectively. If both of the security systems approve this access request, it is approved. Otherwise, it is denied. We hereby develop an access control decision-making algorithm. After receiving an access request \( \alpha \), \( \alpha \) is first validated in \( SEC_{org} \) then validated in \( SEC_{d_i} \). If \( \alpha \) is approved by both \( SEC_{org} \) and \( SEC_{d_i} \), \( \alpha \) is approved. Otherwise, \( \alpha \) is denied. This algorithm is formally presented as follows.
3.3. A GPSS based Privacy Protection System

3.3.2.4 An Example

In this section, a realistic example will be presented to support our idea of privacy control. This example is cited from Agrawal et al.’s use scenario [4]. The solution to this example is using our proposed privacy protection system.

```
BEGIN
  Receive an access request α
  Decide that SEC\textsubscript{org} and SEC\textsubscript{di} are needed for decision-making
  /* Validate α in SEC\textsubscript{org} */
  FIRST VALIDATION:
    Search all matching rules for α in SEC\textsubscript{org}’s Γ, form matching rule set Γ\textsubscript{α}
    Evaluate α by using Γ\textsubscript{α}
    IF α is approved by SEC\textsubscript{org}
      GOTO SECOND VALIDATION
    ELSE
      Deny access
      GOTO END
   ENDIF
  /* Validate α in SEC\textsubscript{di} */
  SECOND VALIDATION:
    Search all matching rules for α in SEC\textsubscript{di}’s Γ, form matching rule set Γ’\textsubscript{α}
    Evaluate α by using Γ’\textsubscript{α}
    IF α is approved by SEC\textsubscript{di}
      Approve access
    ELSE
      Deny access
    ENDIF
END

ACCESS CONTROL DECISION-MAKING ALGORITHM
```
The use scenario is described as follows. Mississippi is an on-line bookseller who needs to obtain certain minimum personal information to complete a purchase transaction. This information includes name, shipping address, and credit card information. Mississippi also needs an email address to notify the customer of the status of the order. Mississippi uses the purchase history of customers to offer book recommendations on its site. It also publishes information about books popular in the various regions of the country (purchase circles).

In the following paragraphs, we will show how our privacy protection system supports customers’ privacy wishes.

System elements for this on-line bookseller system are identified as:

1. **Organization**: On-line bookseller Mississippi (org).

2. **Information donor**: Alice \((d_1)\) and Bob \((d_2)\).

3. **User**: system security officer (SSO) (for managing private data), shipping, charge, customer-service, registration, delivery-company, credit-card-company, olap (for purchase circles), mining (for book recommendations).

4. **Private data**: name, shipping-address, email, credit-card-info, purchase-history.

The privacy protection system for Mississippi contains three security systems, they are:

1. **SEC_{org}**: Mississippi’s security system

2. **SEC_{d_1}**: Alice’s security system

3. **SEC_{d_2}**: Bob’s security system

From the problem description, the policy rule domain is specified. Here we use one prefix: \(all\) to denote a special hierarchy element. For example \(all\text{-}subjects\) is a hierarchy element in subject hierarchy which is located on the top of the hierarchy. That is, \(\forall g \in GH\), where \(g \neq all\text{-}subjects\), \(g\) is a descendant of \(all\text{-}subjects\). Similarly \(all\text{-}objects\), \(all\text{-}purposes\), \(all\text{-}conditions\), and \(all\text{-}obligations\) are used.
3.3. A GPSS based Privacy Protection System

The policy rule domain is as follows.

(1) Subject hierarchy $GH = (G, \leq_G)$: $G = \{ \text{all-subjects, shipping, charge, customer-service, registration, SSO, mining, olap, delivery-company, credit-card-company} \}$, $\leq_G$ is expressed in the subject hierarchy graph (see Figure 3.9).

![Figure 3.9: Subject hierarchy $GH$](image)

(2) Object hierarchy $TH = (T, \leq_T)$: $T = \{ \text{all-objects, name, shipping-address, email, credit-card-info, purchase-history, } d_1 \text{'s name, } d_2 \text{'s name, } d_1 \text{'s shipping-address, } d_2 \text{'s shipping-address, } \ldots \}$, $\leq_T$ is expressed in the object hierarchy graph (see Figure 3.10).

![Figure 3.10: Object hierarchy $TH$](image)
(3) Purpose hierarchy $PH = (P, \leq_P)$: $P = \{ \text{all-purposes, purchase, registration, recommendations, purchase-circles} \}$, $\leq_P$ is expressed in the purpose hierarchy graph (see Figure 3.11).

![Figure 3.11: Purpose hierarchy $PH$](image)

(4) Condition hierarchy $CH = (C, \leq_C)$: $C = \{ \text{all-conditions, during-weekday, during-Monday, during-Tuesday, during-Wednesday, during-Thursday, during-Friday, on-workstation-1~6, on-workstation-1, on-workstation-2, on-workstation-3, on-workstation-4, on-workstation-5, on-workstation-6} \}$, $\leq_C$ is expressed in the condition hierarchy graph (see Figure 3.12).

![Figure 3.12: Condition hierarchy $CH$](image)
(5) Obligation hierarchy \( OH = (O, \leq_O) \): \( O = \{ \text{all-obligations, notify-donor, log-access, notify-donor-and-log-access, 10-year-retention, 3-year-retention, 1-year-retention, 1-month-retention, delete-after-use} \} \), \( \leq_O \) is expressed in the obligation hierarchy graph (see Figure 3.13).

![Obligation hierarchy](image)

Figure 3.13: Obligation hierarchy \( OH \)

(6) Action set: \( A = \{ \text{create, read, delete} \} \).

(7) Ruling set: \( R = \{ +, -, \oplus, \ominus, \otimes \} \).

Organization and information donor can define their policy rules on top of this policy rule domain. We assume there exist some easy-to-use tools to make policy rules for information donors and the system security officer (SSO) of the organization. Moreover, we define the following method for referring a policy rule in our privacy protection system. As mentioned, the privacy protection system for Mississippi contains three security systems: \( SEC_{org}, SEC_{d_1}, \) and \( SEC_{d_2} \). Each of these security systems has a policy rule set, e.g. \( SEC_{org} \) has \( \Gamma_{org} \). For a policy rule \( \gamma_i \in \Gamma_j \), it is referred to as \( \Gamma_{j \cdot \gamma_i} \).
In \( SEC_{org} \), the following are examples of policy rules which may be specified:

1. \( \gamma_1 = ( \text{SSO, all-objects, all-purposes, all-conditions, all-obligations, delete, +} ) \)
2. \( \gamma_2 = ( \text{SSO, name, purchase, all-conditions, 1-month-retention, create, +} ) \)
3. \( \gamma_3 = ( \text{SSO, name, registration, all-conditions, 3-year-retention, create, +} ) \)
4. \( \gamma_4 = ( \text{SSO, purchase-history, purchase, all-conditions, 1-month-retention, create, +} ) \)
5. \( \gamma_5 = ( \text{SSO, purchase-history, recommendations, all-conditions, 10-year-retention, create, +} ) \)
6. \( \gamma_6 = ( \text{SSO, purchase-history, purchase-circles, all-conditions, 1-year-retention, create, +} ) \)
7. \( \gamma_7 = ( \text{delivery-company, name, purchase, during-week-days, notify-donor, read, +} ) \)
8. \( \gamma_8 = ( \text{delivery-company, shipping-address, purchase, during-week-days, notify-donor, read, +} ) \)
9. \( \gamma_9 = ( \text{credit-card-company, name, purchase, during-week-days, notify-donor, read, +} ) \)
10. \( \gamma_{10} = ( \text{credit-card-company, credit-card-info, purchase, during-week-days, notify-donor, read, +} ) \)
11. \( \gamma_{11} = ( \text{customer-service, name, purchase, during-week-days, log-access, read, +} ) \)
12. \( \gamma_{12} = ( \text{customer-service, email, purchase, during-week-days, log-access, read, +} ) \)
13. \( \gamma_{13} = ( \text{customer-service, name, registration, during-week-days, log-access, read, +} ) \)
14. \( \gamma_{14} = ( \text{customer-service, name, registration, during-week-days, log-access, read, +} ) \)
15. \( \ldots \)
For some of the above policy rules, explanations are given as follows:

1. $\Gamma_{org}.\gamma_1$ defines that SSO has rights to delete data.

2. $\Gamma_{org}.\gamma_2$ defines that SSO can create data belonging to the name category for the purchase purpose. The created data can be kept for 1 month, after which it should be deleted by the SSO.

3. $\Gamma_{org}.\gamma_7$ defines that delivery-company can read data belonging to the name category for the purchase purpose when the condition during-week-days is satisfied. After every access, delivery-company needs to notify the information donor of the data about the access.

Now an information donor $d$ can specify policy rules as his/her privacy preferences. These rules are stored in $SEC_d$. Three situations can be identified:

1. If $d$ specifies no policy rules in $SEC_d$, i.e. $\Gamma_d = \emptyset$, then his/her private data is not accessible to Mississippi.

2. If there are three policy rules: $\gamma_1 = (\text{all-subjects, all-objects, all-purposes, all-conditions, all-obligations, delete, +})$, $\gamma_2 = (\text{all-subjects, all-objects, all-purposes, all-conditions, all-obligations, create, +})$, and $\gamma_3 = (\text{all-subjects, all-objects, all-purposes, all-conditions, all-obligations, read, +})$ in $SEC_d$, i.e. $\Gamma_d = \{\gamma_1, \gamma_2, \gamma_3\}$, then he/she accepts every policy rule specified by Mississippi. Accesses to his/her private data are regulated only by $SEC_{org}$.

3. Between these two extremes, $d$ could flexibly specify policy rules in order to protect his/her privacy.

Back to our example, Alice does not want Mississippi to use her information once her purchase transaction is complete. In her security system, she specifies the following policy rules (i.e. $\Gamma_{d_1} = \{\gamma_1, \gamma_2, \gamma_3\}$):

1. $\gamma_1 = (\text{SSO, all-objects, purchase, all-conditions, all-obligations, delete, +})$

2. $\gamma_2 = (\text{SSO, all-objects, purchase, all-conditions, delete-after-use, create, +})$
(3) $\gamma_3 = (\text{all-subjects, all-objects, purchase, during-week-days, notify-donor-and-log-access, read, +})$

The meaning of the above policy rules is explained as follows.

(1) $\Gamma_{d_1} \cdot \gamma_1$ defines that SSO has rights to delete data for the purchase purpose.

(2) $\Gamma_{d_1} \cdot \gamma_2$ defines that SSO can create data for the purchase purpose. Once the purchase is finished, the created data should be deleted by the SSO.

(3) $\Gamma_{d_1} \cdot \gamma_3$ defines that all-subjects can read data for the purchase purpose when the condition during-week-days is satisfied. After every access, the access subject needs to notify the information donor and log the access.

By defining these policy rules, Alice’s privacy preferences are expressed and enforced. Under the protection of $SEC_{org}$ and $SEC_{d_1}$, Mississippi can only use her personal information for the purchase purpose and once the purchase transaction is complete, this personal information is deleted.

Bob, on the other hand, likes the convenience of providing his email and shipping address only once by registering at Mississippi. He also likes Mississippi’s recommendations and does not mind using his purchase history to suggest new recommendations. However, he does not want Mississippi to use his purchase history for purchase circles. In his security system, he specifies the following policy rules (i.e. $\Gamma_{d_2} = \{\gamma_1, \gamma_2, \gamma_3, \gamma_4\}$):

(1) $\gamma_1 = (\text{SSO, all-objects, all-purposes, all-conditions, all-obligations, delete, +})$

(2) $\gamma_2 = (\text{SSO, all-objects, all-purposes, all-conditions, all-obligations, create, +})$

(3) $\gamma_3 = (\text{all-subjects, all-objects, all-purpose, all-conditions, all-obligations, read, +})$

(4) $\gamma_4 = (\text{all-subjects, all-objects, purchase-circles, all-conditions, all-obligations, read, -})$
The meaning of the above policy rules is explained as follows.

1. $\Gamma_{d_2, \gamma_1}$ defines that $SSO$ has rights to delete data.

2. $\Gamma_{d_2, \gamma_2}$ defines that $SSO$ has rights to create data.

3. $\Gamma_{d_2, \gamma_3}$ defines that $all-subjects$ can read data.

4. $\Gamma_{d_2, \gamma_4}$ defines that $all-subjects$ cannot read data for the $purchase-circles$ purpose.

By defining these policy rules, Bob’s privacy preferences are expressed and enforced. Under the protection of $SEC_{org}$ and $SEC_{d_2}$, Mississippi cannot use his personal information for the $purchase-circles$ purpose and Bob can enjoy Mississippi’s other quality services.

This example demonstrates how to use our technique to solve the practical privacy problem. In our privacy protection system, information donors have ability to express and enforce their privacy preferences. Our technique ensures that on the organization side, the access control will abide by information donors’ privacy preferences. As shown in the example, information donors can flexibly opt-in or opt-out of any services provided by Mississippi. And by defining policy rules (subjects, objects, purposes, conditions and obligations), information donors can protect their personal information in an elaborate manner.

### 3.3.2.5 Summary

By comparing privacy with security, we have proposed a privacy control model which is a collection of security systems. The benefits are increased privacy control for individuals, applicability and flexibility, linking security and privacy together seamlessly and making privacy protection system easily implementable. In order to solve practical privacy problems, we have incorporated our privacy control model with a customized GPSS: $IGPSS$. With the supporting example, we have shown that our privacy protection system works very well.
3.4 Chapter Summary

In this chapter, a privacy protection system based on access control has been presented and discussed.

The access control technique we use is Generalized Policy Support System (GPSS). GPSS is a generalized solution to access control because a GPSS can be customized to accommodate itself to different access control needs, e.g. GPSS covers two recent advances in access control: GRBAC and E-P3P. Moreover, GPSS is capable of expressing rich data protection policies and individuals’ privacy preferences. It is ideal to be used in our privacy protection system.

By comparing privacy with security, a new privacy control model has been developed. The model integrates a set of security systems, one for the organization ($SEC_{org}$) and one for each information donor ($SEC_d$). $SEC_{org}$ is configured by the organization and reflects the organization’s privacy control policies. $SEC_d$ is configured by the information donor $d$ and reflects $d$’s privacy preferences. An access request is evaluated against $SEC_{org}$ and related $SEC_d$ and is granted only if it is permitted by both $SEC_{org}$ and $SEC_d$. In such a way, our privacy control model enforces individuals’ privacy preferences in a simple and effective manner. Another main advantage of this approach is that it reuses current access control techniques and allows for a quick and relatively inexpensive implementation of a privacy protection system.

Based on GPSS and the privacy control model, the privacy protection system based on access control has been proposed. This system integrates a set of homogeneous security systems, each of which is an instance of GPSS: $IGPSS$. $IGPSS$ is designed specially for expressing and enforcing privacy control policies and privacy preferences. It includes five hierarchies (subject hierarchy, object hierarchy, purpose hierarchy, condition hierarchy and obligation hierarchy), the action set and the ruling set. The resulting privacy protection system is simple and expressive. An access request is evaluated in a way such that both organization’s privacy control policies and information donors’ privacy preferences are respected. Finally, an illustrating example is used to show how our privacy protection system solves practical privacy problems.
In the next chapter, we present our privacy protection approach based on hierarchical encryption techniques.
Chapter 4

Hierarchical Encryption based Approach

4.1 Introduction

As introduced in Chapter 2, access control is widely deployed as safeguards for information systems. However, access control based systems fail to protect information in the following situations:

(1) When a system administrator is corrupted.

(2) When system storage is compromised by an intruder.

Information is threatened because it is stored as plaintext without encryption. A strategy to prevent above attacks is to encrypt information properly and only store its ciphertext in information systems.

Hierarchical encryption is developed as an alternate approach to access control. By using hierarchical encryption, all information in an information system is encrypted in such a way that data encrypted by a lower level security class can be decrypted by a higher level security class. We have presented a review over hierarchical encryption in Section 2.3.

In this chapter, we propose our private data protection system based on hierarchical encryption.

(1) First, we propose a cryptographic solution aiming at implementing general access control, which is general and efficient.

(2) Second, we present our private data protection system based on hierarchical encryption, which is the proposed cryptographic solution for general access control.
In this system, all private data is encrypted such that for any piece of encrypted private data, only its legitimate users (groups) and the information donor can decrypt it. Unauthorized users, even the system administrator, cannot decrypt it.

We advise that this chapter is based on Kong, Seberry, Getta and Yu’s papers [68, 69].

4.2 A Cryptographic Solution for General Access Control

In this section, we propose a cryptographic solution aiming at general access control. Our solution is based on Chinese Remainder Theorem (CRT) and has two categories: data based solution and key based solution. Assume a data item is to be shared with $k$ participants. In data based solution, this data item is first encrypted by the $k$ participants’ public keys, respectively. Then these $k$ individual ciphertexts are combined by CRT. As a consequence, the final share ciphertext is $k$ times larger than the data item. In key based solution, the data item is first encrypted by a symmetric key to produce a data ciphertext. Next, this symmetric is encrypted by $k$ participants’ public keys, respectively. Finally, these $k$ individual ciphertexts are combined by CRT to produce a symmetric key share ciphertext. The data ciphertext and the symmetric key share ciphertext are concatenated and shared with those $k$ participants. The performance and security analysis shows that our solution is more efficient and secure than the RRN system. Moreover, in our solution, authorization alterations are efficiently supported. Finally, we present a set of experimental results to prove that our solution performs much better than RRN.

4.2.1 Backgrounds

In this section, we will introduce the background knowledge on which our solution is based. For more Number Theory basic knowledge, please refer to the book [120].
4.2.1.1 Chinese Remainder Theorem

The Chinese Remainder Theorem (CRT) is regarded as one of the most useful elements of number theory. In essence, the CRT says it is possible to reconstruct integers in a certain range from their residues modulo a set of pairwise relatively prime moduli [108]. This theorem is formally presented as follows.

**Theorem 1:**

**Chinese Remainder Theorem:**

If the integers $n_1, n_2, ..., n_k$ are pairwise relatively prime, then the system of simultaneous congruences

\[
\begin{align*}
    x &\equiv a_1 \mod n_1 \\
    x &\equiv a_2 \mod n_2 \\
    &\quad\quad\quad\quad\quad\quad\quad.. \\
    x &\equiv a_k \mod n_k
\end{align*}
\]

has a unique solution $x$, such that $0 \leq x < n = n_1n_2...n_k$.

\[\square\]

We call $n_1, n_2, ..., n_k$ the *CRT moduli* and $x$ the *CRT solution*. The proof of CRT is available in most number theory books, e.g. [120]. *Garner’s algorithm* is an efficient method for determining CRT solutions. This algorithm is listed as follows (For further details, please refer to Chapter 14.5 of [84]).
4.2. A Cryptographic Solution for General Access Control

**Algorithm:** Garner’s algorithm for CRT

**INPUT:** a positive integer \( n = \prod_{i=1}^{k} n_i > 1 \), with \( \gcd(n_i, n_j) = 1 \) for all \( i \neq j \), and a modular representation \( a(x) = (a_1, a_2, ..., a_k) \) of \( x \) for the \( n_i \).

**OUTPUT:** the integer \( x \) in radix \( b \) representation.

1. For \( i \) from 2 to \( k \) do the following:
   1.1 \( C_i \leftarrow 1 \).
   1.2 For \( j \) from 1 to \((i-1)\) do the following:
      \( u \leftarrow n_j^{-1} \mod n_i \).
      \( C_i \leftarrow u \cdot C_i \mod n_i \).
2. \( u \leftarrow a_1, x \leftarrow u \).
3. For \( i \) from 2 to \( k \) do the following:
   \( u \leftarrow (a_i - x) \cdot C_i \mod n_i, x \leftarrow x + u \cdot \prod_{j=1}^{i-1} n_j \).
4. Return\((x)\).

### 4.2.1.2 RSA Algorithm

The RSA algorithm [98] contains three parts: key generation, encryption and decryption. Key generation works as follows: find a modulus \( n \) (\( n \) is a product of two large primes) and choose a number \( e \) (\( e \) is a number less than \( n \) and relatively prime to \( \phi(n) \), where \( \phi(n) \) is the Euler’s totient function). Find another number \( d \) such that \( ed \equiv 1 \mod \phi(n) \). The value \( e \) and \( d \) are called the public and private exponents, respectively. The public key \( K \) is the pair \((e, n)\). The private key \( K^{-1} \) is the pair \((d, n)\). The encryption of a message \( m \) with the public key \( K = (e, n) \), denoted by \( E_K(m) \), is defined as:

\[
c = E_K(m) = m^e \mod n .
\]

where \( c \) is the ciphertext produced by the encryption algorithm \( E \). The decryption of a ciphertext \( c \) with the private key \( K^{-1} = (d, n) \), denoted by \( D_{K^{-1}}(c) \), is defined as:

\[
m = D_{K^{-1}}(c) = c^d \mod n .
\]

where \( m \) is the plaintext recovered by the decryption algorithm \( D \).
4.2. A Cryptographic Solution for General Access Control

4.2.2 A Data Based Solution

4.2.2.1 Overview

One popular way of enforcing access control is by the means of Access Control Lists (ACLs). Each piece of data is associated with an ACL, on which its authorized users/groups and corresponding access modes are listed. By looking at an ACL, it is easy to determine who is allowed to do what on the data associated with it. ACL covers the general cases of access control. For example, it supports hierarchical access control. If we generate ACLs according to the hierarchical structure of an organization, then hierarchical access control can be enforced. That is, a data owner and all his/her ancestors are listed on his/her data’s ACLs.

From cryptographic perspective, to enforce general access control, each piece of data must be encrypted such that only subjects on its ACL have the ability to decrypt the data. One straightforward approach exists to solve this problem [96]. Assume each subject is assigned with a pair of keys: a public key and a private key. To share a message \( m \) with \( k \) subjects: \( s_1, s_2, ..., s_k \), for each subject \( s_i \in \{s_1, s_2, ..., s_k\} \), \( m \) is encrypted by \( s_i \)'s public key. Together with a ciphertext for its owner, \( m \) is encrypted \( k + 1 \) times. The system keeps these \( k + 1 \) ciphertexts for sharing a single message \( m \). One negative aspect of this approach has been identified, i.e. storing multiple copies of encrypted data (individual ciphertexts) can be a source of inconsistency [96]. Therefore, to share a single message \( m \), sharing one share ciphertext is preferred rather than sharing multiple individual ciphertexts.

Based on the above straightforward approach, if there exists an efficient method that converts multiple individual ciphertexts to one share ciphertext, then a new approach of enforcing general access control can be established with the advantages of both efficiency and consistency. We have discovered such a method: Chinese Remainder Theorem (CRT). CRT provides a way of mapping a number \( x \in \mathbb{Z}_n \) (\( \mathbb{Z}_n \) is the set of nonnegative integers less than \( n \)) to a series of \( k \) numbers \( a_i \in \mathbb{Z}_{n_i} \), where \( 1 \leq i \leq k, \, n = n_1n_2...n_k \) and \( n_1, n_2, ..., n_k \) are pairwise relatively prime. The mapping is a one-to-one correspondence (called a bijection) between \( \mathbb{Z}_n \) and the Cartesian product \( \mathbb{Z}_{n_1} \times \mathbb{Z}_{n_2} \times ... \times \mathbb{Z}_{n_k} [108] \). This property allows us to construct a share ciphertext
from a series of individual ciphertexts.

### 4.2.2.2 System Elements

Our data based solution consists of the following elements:

1. A set of subjects $S = \{s_1, s_2, ..., s_\ell\}$, where a subject is either a user or a group.

2. A *public key cryptosystem* that consists of three functions:
   
   a. A *Key Generation* function $KG$: $\forall s_i \in S$, $KG$ generates a pair of keys: a *public key* $K_{s_i}$ and its corresponding *private key* $K_{s_i}^{-1}$.
   
   b. An *Encryption* function $E$: $c = E_K(m)$, where $c$ means ciphertext, $m$ means *message* and $K$ means *public key* (*encryption key*).
   
   c. A *Decryption* function $D$: $m = D_{K^{-1}}(c)$, where $K^{-1}$ means *private key* (*decription key*).

3. A *Modulus Generator* $MG$: $\forall s_i \in S$, $MG$ generates a modulus $n_{s_i}$, such that $n_{s_1}, n_{s_2}, ..., n_{s_\ell}$ are pairwise relatively prime. Please note, these moduli are publicly known and will be used as the CRT moduli.

4. A *Shared Information System* (or file system) $SIS$ that stores shared data.

### 4.2.2.3 Cryptographic Access Control

Our data based solution is depicted by a scenario as follows. Assume that a subject $s_i$ wants to share a message $m$ with $k$ subjects $s_{i_1}, s_{i_2}, ..., s_{i_k} \in S$, $s_i$ performs the following operations (For simplicity, we assume that $m < n_{s_1}, n_{s_2}, ..., n_{s_\ell}$. For a longer message, encryption can be performed block by block):

A1. First, $s_i$ computes $k$ individual ciphertexts, i.e. $\forall s_j \in \{s_{i_1}, s_{i_2}, ..., s_{i_k}\}$, $E_{K_{s_j}}(m)$ is calculated.

A2. Second, $s_i$ uses Garner’s algorithm (see Section 4.2.1) to calculate the CRT solution $x$, $0 \leq x < n_{s_1}n_{s_2}...n_{s_k}$, such that $x$ satisfies the following $k$ simultaneous congruences:
4.2. A Cryptographic Solution for General Access Control

(1). \( x \equiv E_{K_{s_{i_1}}}(m) \mod n_{s_{i_1}}. \)

(2). \( x \equiv E_{K_{s_{i_2}}}(m) \mod n_{s_{i_2}}. \)

... 

(k). \( x \equiv E_{K_{s_{i_k}}}(m) \mod n_{s_{i_k}}. \)

A3. Third, \( s_i \) stores \( x \) in \( SIS. \)

For a subject \( s_j \in \{s_{i_1}, s_{i_2}, ..., s_{i_k}\} \), to access \( m \), \( s_j \) needs to compute \( E_{K_{s_j}}(m) = x \mod n_{s_j}. \) Then, \( s_j \) uses the private key \( K_{s_j}^{-1} \) to recover \( m \), i.e. \( m = D_{K_{s_j}^{-1}}(E_{K_{s_j}}(m)). \)

### 4.2.2.4 Authorization Alterations

Alteration of a data item’s authorizations, e.g. a subject is granted/revoked access to a data item, is a frequent event in information systems.

Our data based solution handles authorization alterations according to the status of the affected data item. If the data item is dynamic (i.e. the data item changes at the time of authorization alteration), all operations from A1 to A3 (see Section 4.2.2.3) are re-performed based on the new group of authorized subjects. If the data item is static (i.e. the data item remains the same at the time of authorization alteration), an efficient method is used to process authorization alterations.

**Figure 4.1: Three simultaneous congruences sets (SCSs)**

The method is based on the following property of CRT. Consider the 3 Simultaneous Congruences Sets (SCSs) as shown in Figure 4.1. SCS\(_1\) contains \( k \) simultaneous
congruences, and its CRT solution is $x$. SCS$_2$ is created by adding one congruence to SCS$_1$, and its CRT solution is $x'$. SCS$_3$ is created by removing one congruence from SCS$_1$, and its CRT solution is $x''$. Assume the value of $x$ has already been calculated. To get the value of $x'$, we only need to find the CRT solution for the two congruences: $x' \equiv x \mod n_1n_2...n_k$ and $x' \equiv a_{k+1} \mod n_{k+1}$. To get the value of $x''$, we only need one modular operation: $x'' = x \mod n_1n_2...n_{k-1}$. In a word, the values of $x'$ and $x''$ can be easily derived from $x$.

In our data based solution, granting a subject access to a static data item is equivalent to the transformation from SCS$_1$ to SCS$_2$. The new share ciphertext $x'$ can be derived from the old share ciphertext $x$ efficiently. Revoking a subject from accessing a static data item is equivalent to the transformation from SCS$_1$ to SCS$_3$. The new share ciphertext $x''$ can be derived from the old share ciphertext $x$ simply by a modular operation.

Let us analyze the security of the proposed method for static data item. First, let us consider a special situation: when $x < n_1n_2...n_{k-1}$, $x'' = x \mod n_1n_2...n_{k-1} = x$. In this case, the above revocation method becomes useless because the revoked subject is still capable of decrypting $x''$ (which is equal to $x$). This problem is trivial because the probability of this situation is very low. In Section 4.2.2.1, we have mentioned that CRT’s mapping is a one-to-one correspondence between $\mathbb{Z}_n$ and the Cartesian product $\mathbb{Z}_{n_1} \times \mathbb{Z}_{n_2} \times ... \times \mathbb{Z}_{n_k}$ [108]. The data range $[0, n_1n_2...n_{k-1}]$ is only $\frac{1}{n_k}$ of $[0, n_1n_2...n_k]$. If we choose 1024-bit numbers for CRT moduli, then the probability of $x < n_1n_2...n_{k-1}$ is approximately $2^{-1024}$. However, if $x < n_1n_2...n_{k-1}$, we must re-perform all operations from A1 to A3 to revoke a subject. Finally, it may be argued that it is impossible to revoke a subject from accessing a static data item because the subject can simply store the item before the revoking. Here, we assume some trusted workstations are used for subjects to access encrypted data items, on which saving a data item is disabled.

### 4.2.3 A Key Based Solution

As mentioned in Section 4.2, our cryptographic solution of general access control has two categories: data based solution and key based solution. In data based solution,
to share a message \( m \) with \( k \) participants, the size of the share ciphertext is \( k \) times larger than that of \( m \). As a consequence, data based solution is not preferable if \( m \) or \( k \) is large. Moreover, the data based solution is based on public key cryptosystem. This is because, to share a data item, its owner must know all participants’ encryption keys. In order to protect the confidentiality of decryption keys, we can only use a public key cryptosystem. Public key cryptosystems are typically substantially slower than symmetric key cryptosystems [84]. Therefore, our data based solution is not so efficient, especially when \( m \) or \( k \) is large. In this section, we propose a key based solution, which solves the above problems. Our idea of key based solution is derived from our data based solution: instead of sharing a message, we share its encryption key. The technique used in our key based solution has been used by some secure broadcasting systems, e.g. [28, 122]. In contrast to those secure broadcasting systems, our key based solution applies to a different area: general access control.

In addition to the system elements listed for our data based solution (see Section 4.2.2.2), key based solution requires a symmetric key cryptosystem. Here we denote its encryption function as \( SE \) and its decryption function as \( SD \). This symmetric key cryptosystem is used to encrypt data items and the encryption keys are shared by a public key cryptosystem and CRT.

The key based solution is depicted by the following scenario. If a subject \( s_i \) wants to share a message \( m \) with \( k \) subjects \( s_{i_1}, s_{i_2}, \ldots, s_{i_k} \in S \), \( s_i \) performs the following operations:

**B1.** randomly choose a symmetric key \( K_R \).

**B2.** use \( K_R \) to encrypt \( m \): \( c = SE_{K_R}(m) \).

**B3.** \( \forall s_j \in \{ s_{i_1}, s_{i_2}, \ldots, s_{i_k} \} \), calculate \( E_{K_{s_j}}(K_R) \).

**B4.** find the CRT solution \( x \) to the following \( k \) simultaneous congruences:

\[
\begin{align*}
(1) & \quad x \equiv E_{K_{s_{i_1}}}(K_R) \mod n_{s_{i_1}}. \\
(2) & \quad x \equiv E_{K_{s_{i_2}}}(K_R) \mod n_{s_{i_2}}. \\
& \quad \vdots
\end{align*}
\]
(k). $x \equiv E_{K_{s_{ik}}}(K_R) \mod n_{s_{ik}}$.

B5. store $x || c$ in SIS, where the symbol $||$ means concatenation.

For a subject $s_j \in \{s_{i_1}, s_{i_2}, ..., s_{i_k}\}$, to access $m$, first $s_j$ computes $E_{K_{s_j}}(K_R) = x \mod n_{s_j}$. Then $s_j$ uses private key $K^{-1}_{s_j}$ to retrieve the symmetric key $K_R$, i.e. $K_R = D_{K^{-1}_{s_j}}(E_{K_{s_j}}(K_R))$. Finally $s_j$ uses $K_R$ to recover $m$, i.e. $m = SD_{K_R}(c)$.

In our key based solution, authorization alterations are processed in the following way. For a dynamic data item, whenever its authorization changes, all operations from B1 to B5 are re-performed based on the new group of authorized subjects. For a static data item, if a subject is revoked from accessing the data item, to prevent the subject from using the old symmetric key to retrieve the data item, all operations from B1 to B5 are re-performed based on the new group of authorized subjects. On the other hand, if a subject is granted access to the data item, the re-encryption of data item is not needed because the old symmetric key can still be used. Thus the transformation from SCS1 to SCS2 (see Section 4.2.2.4) can be used to generate a new share ciphertext for the old symmetric key such that the newly authorized subject can retrieve the old symmetric key to decrypt the data item.

In contrast to the data based solution, multiple public key encryptions are performed on a symmetric key and one symmetric key encryption is performed on a data item. Because the size of the symmetric key is usually much smaller than that of the data item, the public key encryptions are more efficient than those of the data based solution. Due to the same reason, the size of the share ciphertext is much smaller than that of the data based solution. In summary, a key based solution is preferable when a data item or the number of participants is large.

4.2.4 Comparison with the RRN System

As discussed in Section 2.3, the RRN system is the most developed hierarchical encryption system and has several advantages over its predecessors. Here, a comparison between our hierarchical encryption system and the RRN system is presented. The results of this comparison demonstrates that our hierarchical encryption system is more
general, secure and efficient. First we depict the technique details of the \textit{RRN} system and point out its disadvantages.

\subsection{RRN System}

The \textit{RRN} system is an RSA based cryptosystem, which can be used not only for access control in a hierarchy but also for general cases. That is, besides supporting access control policies following the hierarchical structure of an organization, the \textit{RRN} system also supports access control policies that do not follow the hierarchical structure. For a brief review of the \textit{RRN} system, please refer to Section 2.3.1. The \textit{RRN} system is based on the following principles \cite{96}.

\textbf{Definition 15:}

Two RSA encryption keys $K_1 = (e_1, n_1)$ and $K_2 = (e_2, n_2)$ are said to be \textit{compatible} if $e_1 = e_2$ and $n_1$ and $n_2$ are relatively prime.

\textbf{Definition 16:}

For two compatible keys $K_1 = (e, n_1)$ and $K_2 = (e, n_2)$, their \textit{product key}, $K_1 \times K_2$, is defined as $(e, n_1n_2)$. $K_1$ and $K_2$ are called \textit{factor keys} of the product key $K_1 \times K_2$.

\textbf{Theorem 2:}

For any two messages $m$ and $\hat{m}$, such that $m, \hat{m} < n_1, n_2$,

\begin{align*}
E_{K_1 \times K_2}(m) &\equiv E_{K_1}(\hat{m}) \mod n_1, \text{ if and only if } m = \hat{m} \\
E_{K_1 \times K_2}(m) &\equiv E_{K_2}(\hat{m}) \mod n_2, \text{ if and only if } m = \hat{m}
\end{align*}

where $K_1 = (e, n_1)$, $K_2 = (e, n_2)$ and $K_1 \times K_2 = (e, n_1n_2)$.

We shall call the ciphertext generated by a factor key ($K_1$ or $K_2$) \textit{individual ciphertext} and the ciphertext generated by their product key ($K_1 \times K_2$) \textit{share ciphertext}. Theorem 2 states that an individual ciphertext can be easily derived from its share ciphertext. Therefore, a message encrypted by a product key can be recovered by any
4.2. A Cryptographic Solution for General Access Control

of its factor keys’ corresponding private keys. We will omit the proof of Theorem 2. For details, please refer to Section 4 of Ray, Ray and Narasimhamurthi’s paper [96].

In an RRN system, the personnel in an organization are organized in a hierarchical structure, which can be represented as a partially ordered set (poset), \((L, <)\). \(L\) is the set of levels of the organization and \(<\) is the dominance relation between the levels. For each level \(L_i \in L\), there is a pair of RSA keys assigned: \(K_{L_i} = (e, n_{L_i})\), \(K_{L_i}^{-1} = (d_{L_i}, n_{L_i})\) such that all RSA public keys in the system are compatible. Moreover, in order to enforce the access control in this hierarchy, a pair of default keys is used.

The default encryption key for \(L_i\) is the product key of all its ancestors’ public keys and its public key \(K_{L_i}\). The default decryption key of \(L_i\) is its private key \(K_{L_i}^{-1}\). In such a way, a message encrypted by \(L_i\)’s default encryption key can be decrypted by \(L_i\) and its ancestors. The RRN system also supports general cases of access control where customized encryption keys are used. The advantages of the RRN system can be summarized as: supporting for general cases of access control, easily incorporated in existing systems, mutual access awareness and protecting for data consistency [96]. However, many problems remain unsolved.

(1) The RRN system is strictly based on RSA cryptosystem, which restricts its application in a wide range of systems.

(2) The RRN system is inefficient.

(a) Generally, the modulus of a product key is a huge number (product of many moduli). It is time-consuming to perform RSA encryption on it.

(b) For a message \(m\), whenever its group of authorized users changes (e.g. a new user is granted to access \(m\)), RRN must re-encrypt \(m\) by using a newly generated encryption key.

(3) The sharing of the RSA public exponent \(e\) opens a potential security hole to attackers.

(4) Share ciphertext size increases proportionally as the number of participants increases. Although this fact has been neglected [96], it is of great importance if original data size is big or numerous participants are involved.
4.2.4.2 Our Solution’s Generality

Our hierarchical encryption solution is general. Our data based solution can be customized equivalent to an RRN system, as follows. Choose RSA as the public key cryptosystem of our data based solution. At the system initialization stage, assign each subject $s_i \in S$ a pair of RSA keys: a public key $K_{s_i} = (e, n_{s_i})$ and a private key $K_{s_i}^{-1} = (d_{s_i}, n_{s_i})$ such that all RSA moduli $n_{s_1}, n_{s_2}, ..., n_{s_t}$ are pairwise relatively prime.

Note, that all subjects share a public exponent $e$. There is no need to use the modulus generator $MG$ here, because we use the RSA moduli as the CRT moduli. To share a message $m$ with $k$ subjects $s_{i_1}, s_{i_2}, ..., s_{i_k} \in S$, the customized system and the RRN system generate two share ciphertexts $x$ and $x'$, respectively. To verify the equivalence of the above customized system and the RRN system, we need to prove that the share ciphertexts generated by the two systems are equal, i.e. $x = x'$.

**Theorem 3:**

In the two systems above, the share ciphertexts $x = x'$.

**Proof.**

To prove $x = x'$, we first demonstrate that $x$ and $x'$ are both the CRT solutions of the same set of simultaneous congruences.

$$x' \mod n_{s_{i_1}} = (m^e \mod n_{s_{i_1}} n_{s_{i_2}} ... n_{s_{i_k}}) \mod n_{s_{i_1}}$$

$$= (m^e - qn_{s_{i_1}} n_{s_{i_2}} ... n_{s_{i_k}}) \mod n_{s_{i_1}}$$

$$= m^e \mod n_{s_{i_1}}$$

$$= E_{K_{s_{i_1}}} (m) .$$

where $m^e = qn_{s_{i_1}} n_{s_{i_2}} ... n_{s_{i_k}} + r$ for some integers $q$ and $r$ ($r < n_{s_{i_1}} n_{s_{i_2}} ... n_{s_{i_k}}$).

Hence $x' \equiv E_{K_{s_{i_1}}} (m) \mod n_{s_{i_1}}$. Similarly, the other $k-1$ congruences $x' \equiv E_{K_{s_{i_2}}} (m) \mod n_{s_{i_1}}, ..., x' \equiv E_{K_{s_{i_k}}} (m) \mod n_{s_{i_k}}$ can be proven.

Thus, $x' < n_{s_{i_1}} n_{s_{i_2}} ... n_{s_{i_k}}$ is a solution to the above $k$ simultaneous congruences. We know that $x$ is also a solution to these $k$ simultaneous congruences. From the Chinese Remainder Theorem, we know that the solution for the $k$ simultaneous congruences is unique in the range $[0, n_{s_{i_1}} n_{s_{i_2}} ... n_{s_{i_k}})$. Therefore $x = x'$ holds.

$\square$
4.2. A Cryptographic Solution for General Access Control

Table 4.1: Performance comparison between RRN and DBS

<table>
<thead>
<tr>
<th>Systems</th>
<th>Encryption</th>
<th>Decryption</th>
<th>Granting access to a subject</th>
<th>Revoking access from a subject</th>
</tr>
</thead>
<tbody>
<tr>
<td>RRN</td>
<td>$O(k^2\ell_e\ell_n^2)$</td>
<td>$O(\ell_d\ell_n^2)$</td>
<td>$O(k^2\ell_e\ell_n^2)$</td>
<td>$O(k^2\ell_e\ell_n^2)$</td>
</tr>
<tr>
<td>DBS</td>
<td>$O(k\ell_e\ell_n^2)$</td>
<td>$O(\ell_d\ell_n^2)$</td>
<td>$O(\ell_e\ell_n^2)$</td>
<td>$O(k\ell_n^2)$</td>
</tr>
</tbody>
</table>

The theorem above indicates that the RRN system is covered as a special case by our data based solution.

4.2.4.3 Performance and Security Analysis

This section compares the performance between an RRN system and a Data Based System (DBS), which is configured as an RRN equivalent (see Section 4.2.2.3). There are two algorithms used in these two systems: fast modular exponentiation algorithm and Garner’s algorithm, whose complexity is detailed in [84, 120].

In both the RRN and the DBS systems, a message $m$ is to be shared with $k$ subjects. $m$ is of $\ell_m$-bit in length. The RSA/CRT moduli of all subjects are of the same bit length: $\ell_n$. The shared public exponent $e$ is $\ell_e$-bit and the private exponents are $\ell_d$-bit (please note, $\ell_d$ is only an approximate value).

The RRN system is purely based on RSA cryptosystem. Assume the fast modular exponentiation algorithm is used. RRN encryption is calculated by $c = m^e \mod n$, where $n$ is the product of the $k$ moduli and of $k\ell_n$-bit in length. Therefore the RRN encryption complexity is $O(\ell_e(k\ell_n)^2) = O(k^2\ell_e\ell_n^2)$. The RRN decryption complexity is $O(\ell_d\ell_n^2)$. DBS system is based on RSA cryptosystem and CRT. Fast modular exponentiation algorithm and Garner’s algorithm are used. DBS encryption consists of $k$ RSA encryptions and one CRT computation, its complexity is $kO(\ell_e\ell_n^2) + O(k\ell_n^2) \approx O(k\ell_e\ell_n^2)$.

The DBS decryption complexity is the same as that of RRN: $O(\ell_d\ell_n^2)$. We next analyze the complexity of authorization alterations. In the RRN system, granting access to a subject (or revoking access from a subject) requires re-encrypting the affected data item. The complexity of this re-encryption is approximately $O(k^2\ell_e\ell_n^2)$. In our data based solution, granting a subject access to a static data item, we only need to generate a new individual ciphertext for the subject and then derive the new share ciphertext.
4.2. A Cryptographic Solution for General Access Control

Table 4.2: Experimental results

<table>
<thead>
<tr>
<th>Systems</th>
<th>Encryption</th>
<th>Decryption</th>
<th>Granting access to a subject</th>
<th>Revoking access from a subject</th>
<th>Share ciphertext size</th>
</tr>
</thead>
<tbody>
<tr>
<td>RRN</td>
<td>71,132 ms</td>
<td>11,707 ms</td>
<td>86,124 ms</td>
<td>57,683 ms</td>
<td>1,008,641 bytes</td>
</tr>
<tr>
<td>DBS</td>
<td>14,320 ms</td>
<td>11,476 ms</td>
<td>1,703 ms</td>
<td>401 ms</td>
<td>1,008,641 bytes</td>
</tr>
<tr>
<td>KBS</td>
<td>581 ms</td>
<td>491 ms</td>
<td>10 ms</td>
<td>561 ms</td>
<td>101,297 bytes</td>
</tr>
</tbody>
</table>

from the old one. The complexity of this process is: $O(\ell e \ell^2 n) + 2O(\ell^2 n) \approx O(\ell e \ell^2 n)$. Revoking a subject from accessing a static data item only needs one modular operation. The complexity of this process is: $O(k\ell^2 n)$. Here we only illustrate authorization alterations for static data items. For dynamic data items, the efficiency of authorization alterations is the same as that of encryption. The performance comparison between an RRN and a DBS is summarized in Table 4.1, which shows that besides decryption, the DBS system is more efficient than the RRN system. Furthermore, our system has the flexibility of choosing an alternative public key cryptosystem which may results in more efficient system than the DBS system.

As we know, the RRN system requires the RSA public exponent $e$ to be shared. This opens a potential security hole to attackers. The claim of [96] that “having multiple copies of the same data encrypted with different keys does not arise” is not true because with the knowledge of the RSA moduli and the participants of a data item, an attacker can create those multiple copies by modular operations. In comparison with the RRN system, if our data based solution uses the RSA cryptosystem, sharing the same RSA public exponent $e$ is not required, i.e. different RSA public exponents can be used. Moreover, our data based solution has the flexibility of choosing an alternative public key cryptosystem which may results in more secure system.

4.2.4.4 Experimental Results

We have written Java programs (see Appendix A) to implement the following three systems: an RRN system, a DBS system and a Key Based System (KBS) using RSA and the Advanced Encryption Standard (AES). Our programs are running on Java 2 Standard Edition (J2SE) 1.4.2 and Windows XP Home Edition. The test machine is
a Pentium M 1.60GHz laptop with 512M memories. In our experiments, we share a 100,000-byte file with 10 participants. The RSA public exponent is 16-bit. The RSA private exponents are approximately 1020-bit. The RSA/CRT moduli are 1024-bit and the AES keys are 128-bit. We have run four tests for each system: encryption, decryption, granting access to a subject and revoking access from a subject (Here we only measure authorization alterations for static data items. For dynamic data items, the efficiency of authorization alterations is the same as that of encryption). The experimental results are shown in Table 4.2, where times are measured in milliseconds (ms) and sizes are measured in bytes. The experimental results demonstrate the following facts, which conform to our earlier discussions.

1. Our data based solution is more efficient than the \textit{RRN} system. The share ciphertext size grows proportionally as the number of participants increases.

2. Key based solution is more efficient than data based solution. And the share ciphertext size does not grow a lot when the number of participants increases.

3. Our authorization alteration mechanism is more efficient than that of the \textit{RRN} system.

\subsection*{4.2.5 Summary}

We have proposed a cryptographic solution for general access control. Our solution is based on the Chinese Remainder Theorem (CRT) and has two categories: data based solution and key based solution. In contrast to the \textit{RRN} system, our data/key based solution is more efficient and flexible. The technique used in our key based solution has been used by some secure broadcasting systems. However, our key based solution applies to a different area: general access control. We have proposed a mechanism for authorization alterations. This mechanism consists of very simple operations, which make it very efficient. Moreover, by using our solution, a system designer has the flexibility of choosing appropriate cryptosystems which may result in more efficient and secure systems. We have utilized a set of experiments to verify our system. The experimental results provide evidences that support our research.
4.3 A Privacy Protection System Based On Hierarchical Encryption

Based on our cryptographic solution for general access control, in this section, we propose a privacy protection system. Two private data encryption algorithms are presented: one deals with encryption following hierarchical structure and the other deals with encryption following information donors’ preferences. In our system, all private data is encrypted such that for any piece of private data only its legitimate users (or groups) and the information owner can decrypt it. Unauthorized users, even the system administrator, cannot decrypt it.

4.3.1 System Elements

In an organization the personnel are grouped in a hierarchical structure where senior groups have more privileges than junior groups. An example of such hierarchy (subject hierarchy $GH$) has been shown in Section 3.2.1. The private data collected in the private information system is either provided by information donors or by users/groups in the organization. Here, we model such a system as follows.

(1) An organization owns a private information system $P$ that stores private data.

(2) There are $n$ users/groups in the organization. The personnel are represented by the subject hierarchy $GH = (G, \leq_G)$, where $G = \{g_1, g_2, ..., g_n\}$ and $\leq_G$ is the partial order on $G$ that defines the hierarchy relationship of elements in $G$. Please note, an individual user is a primary category in $GH$, that is, the user is treated as a group of himself/herself.

(3) There is a set of $\ell$ information donors in the organization: $Donor = \{d_1, d_2, ..., d_\ell\}$. A set of all private data owned by an information donor $d$ in $P$ is denoted by $p(d)$. The private information system $P$ is the union of all information donors’ private data, i.e. $P = \cup_{k=1}^{\ell} p(d_k)$. The $i$th atomic element of the private data owned by an information donor $d$ and submitted to $P$ by a creator $cr$ is denoted as $p^r_i(d)$, where $d$ is the person to whom this private data is related and $cr$ is either a subject
in $G$ or the information donor $d$ (i.e. $cr \in (G \cup \{d\})$). For an information donor $d$, his/her private data set $p(d) = \{p_{x_1}^{x_1}(d), p_{x_2}^{x_2}(d), ...\}$, where $x_i, x_j \in G \cup \{d\}$.

**4.3.2 A Customized Hierarchical Encryption Scheme**

The proposed cryptographic solution for general access control (see Section 4.2) is a general solution in the sense that it is capable of utilizing different cryptosystems to constitute a customized hierarchical encryption scheme. Here, we present a customized hierarchical encryption scheme, which is used in our privacy protection systems. This customized hierarchical encryption scheme is an instance of our key based solution (see Section 4.2.3), where we use RSA as the public key cryptosystem and AES as the symmetric key cryptosystem. This scheme is formally defined as follows.

**Definition 17:**

$KBS$ is a customized hierarchical encryption scheme that consists of the following cryptosystems and algorithms:

1. RSA public key cryptosystem, which contains the following algorithms:
   
   (a) A key generation algorithm $KG$: $KG$ generates an RSA public key $K$ and its corresponding private key $K^{-1}$.
   
   (b) An encryption algorithm $E$: $c = E_K(m)$, where $K$ is a public key, $m$ is a plaintext and $c$ is the corresponding ciphertext generated by $E$.
   
   (c) A decryption algorithm $D$: $m = D_{K^{-1}}(c)$, where $K^{-1}$ is a private key, $c$ is a ciphertext and $m$ is the corresponding plaintext generated by $D$.

2. AES symmetric key cryptosystem, which contains the following algorithms:
   
   (a) A key generation algorithm $SKG$: $SKG$ generates a symmetric key $SK$.
   
   (b) An encryption algorithm $SE$: $c = SE_{SK}(m)$, where $SK$ is a symmetric key, $m$ is a plaintext and $c$ is the corresponding ciphertext generated by $SE$. 
(c) A decryption algorithm $SD$ (in AES, $SE$ and $SD$ are same): $m = SD_{SK}(c)$, where $SK$ is a symmetric key, $c$ is a ciphertext and $m$ is the corresponding plaintext generated by $SD$.

(3) A modulus generation algorithm $MG$: this algorithm is used to generate a set of pairwise relatively prime moduli, which are used as the CRT moduli.

(4) A CRT solution algorithm $CRTS$: this algorithm is used to calculate CRT solutions. We use Garner’s algorithm as $CRTS$ because of its efficiency (see Section 4.2.1.1).

\[ \square \]

### 4.3.3 System Setup

The system needs to be initialized before it is able to work. In the initialization stage, the private information system $P$ is empty. The system sets up parameters for all elements in $(G \cup Donor)$. The setup process is depicted as follows.

(1) Public key generation and assignment: $\forall x \in (G \cup Donor)$, a pair of RSA keys generated by $KG$ is assigned to $x$, which includes a public key $K_x$ and a private key $K_x^{-1}$.

(2) CRT moduli generation and assignment: $\forall x \in (G \cup Donor)$, a CRT modulus $n_x$ generated by $MG$ is assigned to $x$.

After these parameters are set up, the system is initialized. That is, the private information system $P$ is ready to accept private data. Whenever a piece of private data is delivered to $P$, it should be in encrypted form (however its meta data is in plaintext form to enable data retrieval).
The creator $cr$ of a piece of private data $p_i^{cr}(d)$ is responsible for encrypting $p_i^{cr}(d)$ before it is submitted to $P$. First, $cr$ chooses appropriate encryption algorithm (see Section 4.3.4 and Section 4.3.5). Then the ciphertext of $p_i^{cr}(d)$, denoted by $c_{p_i^{cr}(d)}$, is produced by the chosen encryption algorithm. Finally, $c_{p_i^{cr}(d)}$ is padded with meta data and submitted to $P$.

### 4.3.4 Encryption Following Hierarchical Structure

In this section, we propose the first encryption algorithm. In this algorithm, we assume that our encryption policy follows the hierarchical structure of the organization. That is, users of a senior group are capable of decrypting information that could be decrypted by users of a junior group. The encryption result depends on the creator of a piece of private data, as follows (a more flexible scheme will be elaborated in Section 4.3.5, where an information donor can explicitly specify who should have access to data).

1. If the information creator is an information donor, he/she should provide an authorization list that lists a set of authorized subjects. These subjects, their senior subjects and the information donor are capable of decrypting the information.

2. If the information creator is a subject in the organization, then the subject, its senior subjects and the information donor to whom the information is related are capable of decrypting the information. Please note, in this situation, there is no need to provide any authorization lists.

After all authorized subjects have been ascertained, it is ready to encrypt private data such that all authorized subjects have ability to decrypt it. The encryption process has been described in Section 4.2.3. Here we formally model the process of encryption following hierarchical structure in the Encryption Algorithm 1.
BEGIN

To encrypt $p_i^{cr}(d)$, where $cr \in (G \cup \{d\}), d \in Donor$

IF $cr = d$

Retrieve the Authorization List $AL \subseteq G$ from $d$

Add all senior subjects of subjects in $AL$ to $AL$

ELSE IF $cr \neq d$

Authorization List $AL = \{cr\}$

Add all senior subjects of $cr$ to $AL$

END IF

Final Authorization List $FAL = AL \cup \{d\} = \{z_1, ..., z_n\}$

Generate a random AES key $SK$ by using $SKG$

Calculate $c = SE_{SK}(p_i^{cr}(d))$

$count = 1$

WHILE $count \leq n$

Calculate $a_{count} = E_{K_{zcount}}(SK)$

$count = count + 1$

END WHILE

Calculate the CRT solution $x$ by using $CRTS$, where:

$$x \equiv a_1 \mod n_{z_1}$$

$$x \equiv a_2 \mod n_{z_2}$$

...  

$$x \equiv a_n \mod n_{z_n}$$

Concatenate $x$ with $c$: $c_{p_i^{cr}(d)} = x||c$

RETURN($c_{p_i^{cr}(d)}$)

END

∀$z \in FAL$, to access $p_i^{cr}(d)$, first $z$ computes $E_{K_z}(SK) = x \mod n_z$. Then $z$ uses private key $K_z^{-1}$ to retrieve the symmetric key $SK$, i.e. $SK = D_{K_z^{-1}}(E_{K_z}(SK))$.

Finally $z$ uses $SK$ to recover $p_i^{cr}(d)$, i.e. $p_i^{cr}(d) = SD_{SK}(c)$. 

The Encryption Algorithm 1
4.3. A Privacy Protection System Based On Hierarchical Encryption

To clearly illustrate this type of encryption, an example is utilized. Assume we have:

(1) An organization which owns a private information system \( P \).

(2) A set of subjects in the organization: \( G = \{g_1, g_2, g_3, g_4, g_5\} \). The hierarchy of \( G \): \( GH = (G, \leq_G) \) is given in Figure 4.2.

(3) A set of information donors in the organization: \( Donor = \{d_1, d_2\} \).

![Figure 4.2: An example of subject hierarchy \( GH = (G, \leq_G) \)](image)

At the initialization stage, the following parameters are assigned to the elements in \((G \cup Donor)\).

(1) \( g_1 \): RSA Key pair \( K_{g_1}, K_{g_1}^{-1} \) and CRT modulus \( n_{g_1} \).

(2) \( g_2 \): RSA Key pair \( K_{g_2}, K_{g_2}^{-1} \) and CRT modulus \( n_{g_2} \).

(3) \( g_3 \): RSA Key pair \( K_{g_3}, K_{g_3}^{-1} \) and CRT modulus \( n_{g_3} \).

(4) \( g_4 \): RSA Key pair \( K_{g_4}, K_{g_4}^{-1} \) and CRT modulus \( n_{g_4} \).

(5) \( g_5 \): RSA Key pair \( K_{g_5}, K_{g_5}^{-1} \) and CRT modulus \( n_{g_5} \).

(6) \( d_1 \): RSA Key pair \( K_{d_1}, K_{d_1}^{-1} \) and CRT modulus \( n_{d_1} \).

(7) \( d_2 \): RSA Key pair \( K_{d_2}, K_{d_2}^{-1} \) and CRT modulus \( n_{d_2} \).
4.3. A Privacy Protection System Based On Hierarchical Encryption

An information donor $d_1$ wants to submit his/her private data $p^{d_1}_{1}(d_1)$ to $P$. The donor $d_1$ provides the authorization list for $p^{d_1}_{1}(d_1)$ as $\{g_2, g_3\}$. Then $d_1$ uses the encryption algorithm 1 to generate the encrypted private data $c_{p^{d_1}_{1}(d_1)}$. Because $g_1$ is the senior group of $g_2$ and $g_3$ (see Figure 4.2), the final authorization list is $\{g_1, g_2, g_3, d_1\}$. The encrypted private data $c_{p^{d_1}_{1}(d_1)}$ returned by the encryption algorithm 1 could be decrypted by using one of the RSA private keys of $g_1, g_2, g_3$ and $d_1$.

A user in group $g_4$ creates a piece of private data related to $d_2$: $p^{g_4}_{1}(d_2)$. The user uses the encryption algorithm 1 to generate the encrypted private data $c_{p^{g_4}_{1}(d_2)}$. Because $g_1$ and $g_2$ are senior groups of $g_4$ (see Figure 4.2), the final authorization list is $\{g_1, g_2, g_4, d_2\}$. The encrypted private data $c_{p^{g_4}_{1}(d_2)}$ returned by the encryption algorithm 1 could be decrypted by using one of the RSA private keys of $g_1, g_2, g_4$ and $d_2$.

The example above shows that the proposed encryption scheme follows the hierarchical structure of the organization. However, there are cases when this scheme is not utilizable. In next subsection, we will introduce another encryption scheme as a complement.

4.3.5 Encryption Following Information Donors’ Preferences

Sometimes information donors may have special requirements of sharing their private data such that the encryption policy does not follow the hierarchical structure of the organization. For example, assume in a pharmacy, the sales group is a junior group to the manager group. Patient $A$ wants his prescription to be viewed only by the sales group (we assume that the prescription is in electronic form), not by any other groups including the manager group. This requirement cannot be enforced by the encryption algorithm 1. As a result, in this section, we propose a new encryption algorithm.

In this new algorithm, whenever a piece of private data is created, regardless of its creator, the information donor (information owner) should provide an authorization list. Based on this authorization list, the new algorithm returns the ciphertext corresponding to the private data. The process of encrypted private data generation is depicted below by the Encryption Algorithm 2.
BEGIN
To encrypt \( p^c_r(d) \), where \( cr \in (G \cup \{d\}), d \in Donor \)
Retriece the Authorization List \( AL \subseteq G \) from \( d \)
Final Authorization List \( FAL = AL \cup \{d\} = \{z_1, ..., z_n\} \)
Generate a random AES key \( SK \) by using \( SKG \)
Calculate \( c = SE_{SK}(p^c_r(d)) \)
\( count = 1 \)
WHILE \( count \leq n \)
\( \quad \text{Calculate } a_{count} = E_{K_z^{count}}(SK) \)
\( \quad \text{count} = \text{count} + 1 \)
END WHILE
Calculate the CRT solution \( x \) by using \( CRTS \), where:
\( x \equiv a_1 \mod n_{z_1} \)
\( x \equiv a_2 \mod n_{z_2} \)
\( \quad \text{...} \)
\( x \equiv a_n \mod n_{z_n} \)
Concatenate \( x \) with \( c \): \( c_{p^c_r(d)} = x || c \)
RETURN \( (c_{p^c_r(d)}) \)
END

The encryption algorithm 2

\( \forall z \in FAL, \) to access \( p^c_r(d) \), first \( z \) computes \( E_{K_z}(SK) = x \mod n_z \). Then \( z \) uses private key \( K_z^{-1} \) to retrieve the symmetric key \( SK \), i.e. \( SK = D_{K_z^{-1}}(E_{K_z}(SK)) \) Finally \( z \) uses \( SK \) to recover \( p^c_r(d) \), i.e. \( p^c_r(d) = SD_{SK}(c) \).

To encrypt a piece of personal information \( p^c_r(d) \), \( cr \) must use a suitable encryption algorithm to generate the corresponding ciphertext. The decision of whether to use the encryption algorithm 1 or the encryption algorithm 2 is given by the information donor \( d \). To clearly illustrate this type of encryption, we will utilize the example used in Section 4.3.4.

An information donor \( d_1 \) wants to submit another piece of private data \( p^c_{12}(d_1) \) to \( P \). \( d_1 \) provides the authorization list for \( p^c_{12}(d_1) \) as \( \{g_2, g_3\} \) and chooses the encryption
algorithm 2. The final authorization list is \( \{g_2, g_3, d_1\} \). The encrypted private data \( c_{p_2^d(d_1)} \) returned by the encryption algorithm 2 could be decrypted by using one of the RSA private keys of \( g_2, g_3 \) and \( d_1 \).

A user in group \( g_5 \) creates a piece of private data related to \( d_1 \): \( p_{g_5}^d(d_1) \). After the creation of \( p_{g_5}^d(d_1) \), the user asks \( d_1 \) for his/her choice of the encryption algorithm. Assume that \( d_1 \) replies with the encryption algorithm 2 and an authorization list \( \{g_3\} \). The final authorization list is \( \{g_5, d_1\} \). The encrypted private data \( c_{p_3^g(d_1)} \) returned by the encryption algorithm 2 could be decrypted by using one of the RSA private keys of \( g_5 \) and \( d_1 \).

4.3.6 Summary

Inspired by our cryptographic solution for general access control, we have proposed a novel way of privacy protection in information systems. Based on hierarchical encryption, this system encrypts all private data before it is collected by an organization. By using this technique, the private data is cryptographically secured. According to information donors’ preferences, one of the two private data encryption algorithms can be used. The encrypted private data is then stored and used by the organization. The privacy is protected by encryption because only authorized users/groups are capable of decryption. We have utilized an illustrating example to demonstrate the usage of the system.

4.4 Chapter Summary

In this chapter, a privacy protection system based on hierarchical encryption has been presented and discussed.

Hierarchical encryption is developed as an alternate solution to implement access control in a hierarchy. Assume there is a subject hierarchy \( GH = (G, \leq_G) \), then hierarchical encryption ensures that: \( \forall x \in G \), a message \( m \) encrypted by \( x \) can be decrypted by \( y \) if and only if \( y \in Aset_x \) holds. In such a way, hierarchical encryption achieves the same effects provided by access control in a hierarchy. Moreover, encryption protects information against corrupted insiders (e.g. an administrator) and successful intruders,
4.4. Chapter Summary

which cannot be achieved by access control. Recently, the RRN system was proposed as a cryptographic solution for general access control. That means besides implementing access control in a hierarchy, RRN also implements arbitrary access control needs. In this chapter, we have proposed a new cryptographic solution for general access control. Our solution is based on the Chinese Reminder Theorem (CRT). Compared to the RRN system, our solution has the following prominent advantages:

(1) Our solution is more flexible in the sense that it is capable of utilizing arbitrary encryption/decryption algorithms.

(2) Our solution is much more efficient.

(3) Our solution is able to generate smaller ciphertexts.

Based on our cryptographic solution for general access control, we have proposed a novel way of privacy protection in information systems. First, we have customized a hierarchical encryption scheme that is based on RSA and AES. Second, we have proposed two private data encryption algorithms. In our private data protection system, private data is encrypted based on information donors' privacy preferences such that for any piece of encrypted private data, only its legitimate users (groups) and information donor can decrypt it. Unauthorized users, even system administrator, cannot decrypt it.

In the next chapter, we will present our privacy protection approach based on digital tickets.
5.1 Introduction

In previous chapters, we have depicted privacy control that is based on access control and hierarchical encryption technologies. Here, we will introduce our privacy protection approach, which is based on digital tickets.

In this chapter, we first enumerate the advantages of using digital tickets to protect privacy. Then we propose a digital ticket privacy control system, which utilizes XML digital ticket technique (for details, see Section 2.4.2). In the rest parts of this chapter, we simply use the term digital ticket to refer to XML digital ticket. In our system, a piece of private data is encrypted by an appropriate encryption key before it is submitted to an organization. A user can access a piece of private data only if he/she has a valid ticket and conditions listed on the ticket are satisfied. During this process, private data is decrypted by authorized users at the last possible stage. Later on, obligations of every access must be “paid” and consistent with restrictions written on tickets. Otherwise, users’ credibility will be decreased. At the end of this chapter, a comprehensive security analysis is provided.

We advise that this chapter is based on Kong, Seberry, Getta and Yu’s paper [70].
5.2 The Advantages of Using Digital Ticket for Privacy Control

The study of digital ticket techniques shows that it is ideal for rights management under a distributed, distrustful environment. The same environment applies to the problem of privacy control, where organizations collect private data of information donors and the information is used by either internal or external users of organizations. The characteristics of the above environment are depicted as follows. Usually, organizations are not trusted by information donors. Because there are a huge number of organizations, many of which are unknown to an information donor, trust relationship is hard to be established. The same situation happens between users and information donors. As we know, in general, organizations, users and information donors are distributed in a wide area. As a result, it is a distributed, distrustful environment. In this environment, using digital ticket to manage accesses to private data has the following advantages:

(1) **Flexibility**: a user can easily get access by requesting a ticket and an information donor can easily grant access by asking a ticket issuer to issue a ticket. They do not have to enter into long-term contractual relationships with organizations.

(2) **Different privacy control strategies**: for different private data, information donors may choose different control strategies. Information donors may either directly participate in ticket issuing decision-making or delegate the decision rights to ticket issuers, where information donors’ privacy preferences are given according to which ticket issuers make ticket issuing decision. This thesis covers the most rigid case of privacy control, where information donors participate in every ticket issuing decision-making. Other cases of privacy control are beyond the scope of this thesis.
5.2. The Advantages of Using Digital Ticket for Privacy Control

(3) **Pre-computed access decision**: a ticket contains all the necessary information to make access decision. Complicated access validation (e.g. the comparison of organization privacy policies and information donor’s privacy preferences) is computed before the ticket is issued. At the point of access, only some simple conditions (e.g. valid period of a ticket) need to be checked. It increases efficiency at the point of access.

(4) **Centralized trust**: an information donor does not need to set up multiple accounts in multiple organizations. He/She only needs to choose and register with a ticket issuer who is responsible for managing his/her privacy according to his/her privacy preferences. In this way, information donors receive professional privacy care at great convenience. Moreover, it is easier to set up trust relationship with one ticket issuer than multiple organizations.

(5) **Consent**: consent is represented by digital tickets. That is because an access can only be accomplished with a valid digital ticket and a digital ticket cannot be issued without an information donor’s approval. Therefore, every access to private data has information donor’s consent, i.e. a digital ticket.

(6) **Restricted rights of administrators**: in our digital ticket based system, private data is encrypted. The keys to decrypt private data are controlled by information donors. Administrators do not have rights to grant access because they cannot forge digital tickets.

(7) **Comprehensive control**: the properties of digital tickets, such as valid period, reusability, ticket transferability and ticket cancellation, provide comprehensive control on private data. For example, temporal control is supported. In addition, in our ticket based system, an information donor is capable of controlling every access to his/her private data by issuing appropriate tickets.

(8) **Proof of compliance**: it is easy to monitor all activities because digital tickets contain all information about access. That is, digital tickets are evidences of access. Compliance with privacy acts and privacy preferences could be easily verified.
5.3 A Digital Ticket Privacy Control System

In this section, we will propose a digital ticket privacy control system, which utilizes the digital ticket technique to protect personal privacy.

5.3.1 Participants

Before we introduce our digital ticket system, the participants of the system are clarified as follows.

(1) A Ticket Circulation Manager $TCM$: $TCM$ keeps information of all issued digital tickets and ticket circulation certificates. For example, $TCM$ utilizes consumption certificates to prevent unauthorized re-consumption of a digital ticket.

(2) A set of organizations: they receive digital tickets, check digital tickets’ validity with the ticket circulation manager and provide ticket owners access to private data.

(3) A set of users: they request digital tickets for accessing private data.

(4) A set of information donors: they submit private data to organizations and make decisions of digital ticket issuing.

(5) A set of ticket issuers: they are responsible for ticket issuing and information donors’ privacy management.

5.3.2 The Structure of the System

Figure 5.1 shows the structure of our digital ticket privacy control system. Their relationships are as follows.

(1) Private data submission: information donors submit their private data to organizations.

(2) Ticket issuing decision: whenever a ticket issuer $I$ receives a ticket request, the related information donor is notified about this request and participates in the
Figure 5.1: The digital ticket privacy control system

decision-making of the ticket issuing. If an information donor delegates the decision rights to $I$, then $I$ must act according to the information donor’s privacy preferences. Information donors can monitor $I$’s compliance by querying $TCM$.

(3) **Ticket acquisition**: to access a piece of private data, a user needs to apply for an appropriate ticket from $I$. $I$ authenticates the user and makes the ticket issuing decision with the related information donor. With the donor’s permission, $I$ issues a ticket to the user.

(4) **Ticket consumption**: a user presents a ticket to an organization to acquire private data. The organization validates the ticket with $TCM$. If the ticket is valid, private data requested on the ticket is provided to the user.

(5) **Ticket query**: ticket query includes verification of ticket validity, submission of digital ticket information, submission of ticket circulation certificates and so forth. For example, an information donor $d$ may ask $TCM$ about the status of a ticket issued with $d$’s permission. $d$ may also ask $TCM$ to cancel a ticket before it is used.
5.3. A Digital Ticket Privacy Control System

5.3.3 An Encryption Scheme

We propose an encryption scheme to ensure the confidentiality of private data. Our scheme is based on the homomorphic property of the RSA cryptosystem [98] (for details of the RSA cryptosystem, please refer to the Section 4.2.1.2). The homomorphic property is described as follows: let \( m_1 \) and \( m_2 \) be two plaintext messages and let \( c_1 \) and \( c_2 \) be their respective RSA ciphertexts. The ciphertext corresponds to the plaintext \( m = m_1 m_2 \mod n \) is \( c = c_1 c_2 \mod n \), where \( n \) is the RSA modulus.

We next depict our encryption scheme. For every participant \( x \) in our system, there is a pair of RSA keys assigned: a public key \( K_x = (e_x, n_x) \) and a private key \( K_x^{-1} = (d_x, n_x) \). For instance,

(1) for the Ticket Circulation Manager TCM, a public key \( K_{TCM} = (e_{TCM}, n_{TCM}) \) and a private key \( K_{TCM}^{-1} = (d_{TCM}, n_{TCM}) \) are assigned.

(2) for an information donor \( d \), a public key \( K_d = (e_d, n_d) \) and a private key \( K_d^{-1} = (d_d, n_d) \) are assigned.

(3) for a user \( u \), a public key \( K_u = (e_u, n_u) \) and a private key \( K_u^{-1} = (d_u, n_u) \) are assigned.

(4) for an organization \( O \), a public key \( K_O = (e_O, n_O) \) and a private key \( K_O^{-1} = (d_O, n_O) \) are assigned.

To submit a piece of private data \( p_d \) to \( O \), \( d \) chooses a random number \( m_1 < n_d, n_u, n_O \) and submits \( m_1 \) to \( O \). In such a way, \( p_d \) remains secret from \( O \). To access \( p_d \), \( u \) asks \( d \) for a ticket \( T \). \( T \) consists of a clue \( c'_2 \) to recover \( p_d \):

\[
c'_2 = E_{K_{TCM}}(E_{K_u}(m_2)), \text{ where } p_d = m_1 m_2 \mod n_u
\]

To access \( p_d \), \( u \) presents \( T \) to \( O \). \( O \) performs the following operations to give \( u \) access.

(1) \( O \) checks \( T \)'s validity with \( TCM \). If \( T \) is valid, \( TCM \) opens the clue by decrypting \( c'_2 : c_2 = D_{K_{TCM}^{-1}}(c'_2) \) and appends \( c_2 \) to \( T \). Otherwise, \( T \) remains unchanged. \( TCM \) sends the validation results and \( T \) back to \( O \). If \( T \) is not valid, \( O \) refuses \( u \)'s request. Otherwise \( O \) continues the next step.
(2) \( O \) encrypts \( m_1 \) by using \( u \)'s public key \( K_u \). Ciphertext \( c_1 = E_{K_u}(m_1) \) is produced.

(3) \( O \) computes \( c = c_1c_2 \).

(4) \( O \) gives \( u \) access to \( c \). Note that \( c = c_1c_2 \equiv (m_1m_2)^{e_u} \equiv (p_d)^{e_u} \mod n_u \), thus \( u \) is capable of recovering \( p_d \) by using \( K_u^{-1} \).

We now analyze the security of this scheme. First, \( p_d \) is only accessible to the authorized user \( u \) at the last possible stage. In the process of ticket circulation, \( p_d \) is kept secret from \( O \) and \( u \). Only after the conditions on the ticket are validated, \( u \) has access to \( E_{K_u}(p_d) \). Second, this scheme protects \( p_d \) against a collusion attack, where \( u \) and \( O \) collude to bypass \( T \)'s validation by \( TCM \). The clue \( c'_2 \) is protected by \( TCM \)'s public key \( K_{TCM} \). Only after \( TCM \) validates \( T \) and \( T \) is valid, \( c'_2 \) can be opened. Therefore, without \( TCM \)'s effort, \( p_d \) is not accessible.

### 5.3.4 Digital Ticket Structure

A digital ticket used in our system includes, but is not limited to, the following fields:

(1) **Ticket identifier**: the identity of this ticket.

(2) **Ticket owner’s identifier**: the identity of the owner of this ticket.

(3) **Information donor’s identifier**: the identity of the information donor related to this ticket.

(4) **Ticket owner’s rights**: it contains a set of atomic private data identifiers.

(5) **Clues for private data access**: by using these clues, the ticket owner is capable of recovering requested private data.

(6) **Conditions of the ticket**:

   (a) **Temporal constraints**: it contains a start time and an end time that specify the valid period of this ticket.

   (b) **Reusability counter**: it contains a counter that stores the number of times this ticket is allowed to be consumed.
5.3. A Digital Ticket Privacy Control System

(7) *Timestamp*: it contains a timestamp that indicates the time when this ticket is issued.

(8) *Issuer’s digital signature*: this ticket is digitally signed by its issuer.

The digital ticket described above can be implemented by using Fijimura, Nakajima and Sekine’s XML ticket language [44]. However, the actual implementation of digital tickets and ticket circulation certificates is beyond the scope of this thesis.

5.3.5 The Protocols of Digital Ticket Circulation

In our system, there are five types of transactions: issue, cancellation, transfer, consumption and obligation. In the following parts of this chapter, we will use the symbol $||$ to denote concatenation.

![Ticket Circulation Manager TCM](image)

**Figure 5.2: Issue protocol**

**Issue protocol**: as shown in Figure 5.2, in an issue transaction, a user $u$ sends a ticket request $R$ to ticket issuer $I$. Then $I$ forwards $R$ to the relevant information donor $d$ and gives $d$ information about $u$’s credit information to help $d$ with the issuing decision. $d$ makes ticket issuing decision and sends this decision to $I$. If $d$ permits to issue a ticket, $I$ prepares a digital ticket $T$ and registers $T$ with the ticket circulation manager $TCM$. After this registration is confirmed, $I$ sends $T$ to $u$. Please note, if $d$ delegates the issuing decision rights to $I$, $d$ does not need to be involved in every ticket issuing transaction. However this issue is beyond the scope of this thesis.
5.3. A Digital Ticket Privacy Control System

Cancellation protocol: as shown in Figure 5.3, in a cancellation transaction, to cancel a ticket $T$, its information donor $d$ prepares a cancellation certificate $CAC_T$ and sends $CAC_T$ to $TCM$. $TCM$ checks if the ticket is cancellable (i.e. the ticket is not expired and is not completely consumed). If $T$ is cancellable, $TCM$ marks this ticket as a cancelled ticket. Otherwise, $TCM$ makes no change to the ticket $T$. $TCM$ attaches $CAC_T$ to $T$ and sends $d$ the cancellation result (e.g. whether the cancellation is successful).

Transfer protocol: as shown in Figure 5.4, in a transfer transaction, a user $u_j$ sends a transfer request to a user $u_i$ that requires transferring $u_i$’s digital ticket $T$. Then $u_i$ prepares a transfer certificate $TC_T$ and sends $T||TC_T$ to $TCM$. $TCM$ checks whether $T$ is allowed to transfer (Please note, this transfer transaction may need information donor $d$’s consent, however, this issue is beyond the scope of this thesis). After receiving $TCM$’s confirmation, $u_i$ sends the new ticket $T = T||TC_T$ to $u_j$. 
5.4 Security Analysis

In this section, we analyze the security aspects of our digital ticket privacy control system.

**Consumption protocol:** as shown in Figure 5.5, in a consumption transaction, a user \( u \) presents ticket \( T \) together with \( T \)'s consumption certificate \( COC_T \) to \( O \). Then \( O \) forwards \( T||COC_T \) to \( TCM \) to validate \( T \). If \( T \)'s validity is confirmed by \( TCM \), \( O \) sends the requested private data to \( u \).

**Obligation protocol:** this is a step after the consumption of a ticket, its consumer needs to complete the obligations defined on the ticket. As shown in Figure 5.6, in an obligation transaction, a user \( u \) prepares \( T \)'s obligation certificate \( OC_T \) and sends \( T||OC_T \) to \( TCM \). \( TCM \) checks the compliance of the fulfillment of those obligations and sends the obligation verification result to \( u \).
5.4.1 Security of Our System

Unlike other privacy control system, our system does not require a trusted party that has full control over the information donor’s private data. In our system, organizations, ticket issuers and the TCM are partially trusted. They cannot disclose private data because the clues to recover the private data are controlled by its information donor. Still there is a possibility that some participants of our system could collude to violate privacy. Here, we enumerate some possible collusion attacks and their countermeasures.

(1) The first possible collusion attack is the collusion between organizations and users. This type of attack does not compromise our system because the cooperation of an organization $O$ and a user $u$ does not disclose any private data other than what is already disclosed to $u$. This has been discussed in Section 5.3.3.

(2) In addition to the above collusion attack, there exist other three types of collusion attacks: organization and ticket issuer collusion, user and ticket issuer collusion and organization, user and ticket issuer collusion. One common characteristic of these three attacks is that a ticket issuer is included in the collusion. A secret sharing scheme could be used to reduce the risk of these collusion attacks. Suppose $k$ ticket issuers $I_1, I_2, ..., I_k$ are in charge of issuing digital tickets for the information donor $d$. Now a ticket $T$ is approved by $d$ to be issued to user $u$. Then the clue in $T$ for $u$ to recover $d$’s private data $c_2 = E_{K_{TCM}}(E_{K_u}(m_2))$ is regarded as the secret to share. To share $c_2$, $d$ randomly chooses $k - 1$ numbers $x_1, x_2, ..., x_{k-1}$ and computes the $k^{th}$ number $x_k$ such that the product of these $k$ numbers is equal to $c_2$, i.e. $c_2 = x_1x_2...x_k$. Then $d$ sends $x_1, x_2, ..., x_k$ to $I_1, I_2, ..., I_k$, respectively. To issue a valid ticket, these $k$ ticket issuers need to cooperate. Otherwise the issued ticket is useless because of incomplete clues. In this way, we minimize the threat of these collusion attacks. It is also possible to reduce the trust on TCM by adding more TCMs to monitor digital tickets circulation. For example, ticket query results must be signed by all TCMs. Otherwise, the results are treated as invalid.
5.4.2 Security of Ticket Circulation Protocols

We next analyze the security of our ticket circulation protocols. The cryptographic properties we want to preserve in our protocols are: confidentiality, data integrity, data origin authentication and non-repudiation [22]. Confidentiality is protected by encryption. That is, if a message transmitted in our protocol is confidential, then it should be encrypted by using its recipient’s public key. Data integrity and data origin authentication are protected by digital signatures. In our protocols, all messages that need these two properties are digitally signed. For example, digital tickets and ticket circulation certificates are digitally signed by appropriate participants of the system. Therefore it is impossible to forge them. Non-repudiation is also protected by digital signatures. For example, if a ticket $T$ is transferred from user $u_i$ to user $u_j$, then a transfer certificate $TC_T$ signed by $u_i$ is registered with $TCM$. After this transfer transaction, $u_i$ cannot deny having permitted this transfer because of the genuineness of $TC_T$. Other ticket circulation certificates such as cancellation certificates, consumption certificates and obligation certificates are also used for the sake of non-repudiation. Furthermore, timestamps should be added to messages in our protocol to prevent a replay attack, in which an adversary records information seen in a protocol and then sends it to the same or a different principal, possibly during a later protocol run [22]. For example, assume a ticket $T$ is allowed to be consumed several times and $T$’s owner is user $u$. Every time $u$ consumes $T$, a consumption certificate signed by $u$ is registered with $TCM$. If no timestamp is included in consumption certificates, it is possible for an adversary to replay the consumption protocol and register consumption certificates legitimately. Although the adversary cannot gain access to any private data, $u$’s legitimate rights for private data access are affected.

5.5 Comparison with the LO System

Using digital tickets to protect privacy is not a new idea. Lategan and Olivier [74] proposed a system (LO system) to limit access to private information, which is based on Kerberos tickets (see Section 2.4.3 for more details). The approach of the LO system
is similar to ours in the sense that it utilizes digital ticket to protect privacy. That is, in a LO system, every access to private data is controlled by a Kerberos ticket. However, LO requires a trusted third party to store private data. It is obvious that the unrestricted rights of the trusted third party put great threats to privacy. As a contrast to the LO system, our system does not require a trusted third party to store private data. Moreover, instead of using Kerberos tickets, our system utilizes the idea of generalized digital ticket framework (XML digital tickets) [42, 43, 44, 83] (for details, see Section 2.4.2) that supports more features, e.g. composability, state manageability etc. These features make elaborate privacy control realistic. In the rest parts of this chapter, we simply use the term digital ticket to refer to XML digital ticket.

5.6 Chapter Summary

In this chapter, we have proposed a privacy protection system based on digital tickets. First we enumerate the advantages of using digital tickets for privacy control. In particular, we have illustrated that these inherent properties of digital tickets make them fit for privacy management in a distributed, distrustful environment. Inspired by these advantages, we propose a digital ticket privacy control system. In our system, information collection organizations cannot read the content of private data because private data is encrypted before its submission. Every access to private data is controlled by a digital ticket. Private data is only disclosed to its authorized users at the last possible stage after all restrictions are satisfied. Later on, obligations of every access must be “paid” and consistent with restriction written on tickets. Otherwise, users’ credibility will be decreased. At the end of this chapter, a comprehensive security analysis has been provided.

In the next chapter, the conclusion of this thesis is presented.
Chapter 6

Conclusion

In this thesis, we have discussed several novel privacy protection approaches. These approaches are based on access control techniques, hierarchical encryption techniques and digital tickets techniques, respectively. And these approaches are highlighted for their consideration for information donors’ privacy preferences, which are often neglected in previous privacy protection approaches.

The access control techniques, hierarchical encryption techniques and digital tickets techniques are normally considered as security protection techniques. These techniques have been studied extensively for the sake of protecting information security. In terms of protecting information privacy, only a few researches have been done by utilizing the above security protection techniques. Moreover, our literature review (See Chapter 2) reveals that these privacy protection solutions provide individuals limited control over their privacy. For example, in some privacy protection solutions, a private data owner is not able to change his/her privacy preferences over the private data after it is submitted to an organization. This lack of control problem makes a negative impact on the success of the information systems that contain private data. As a comparison, the privacy protection approaches proposed in this thesis increase individuals’ control over their privacy. They also claim some other advantages, such as generality, flexibility and efficiency.

We have proposed a privacy protection approach based on access control. First, we developed an access control technique: Generalized Policy Support System (GPSS). GPSS is a generalized solution to access control because a GPSS can be customized to accommodate itself to different access control needs, e.g. GPSS covers two recent
advances in access control: GRBAC and E-P3P. Moreover, GPSS is capable of expressing rich data protection policies and individuals’ privacy preferences. It is ideal to be used in our privacy protection system. We have compared our GPSS with GRBAC and E-P3P to show its superiority. Second, we proposed a new privacy control model that integrates a set of security systems: one for an organization and one for each information donor who submits private data to the organization. This model provides a platform for us to utilize existing security mechanisms to protect privacy. In addition, it is a platform for expressing and enforcing both individuals’ privacy preferences and organization’s privacy policies. Finally, based on the privacy control model, we utilize GPSS to construct a privacy protection system. This system integrates a set of homogeneous security systems, each of which is an instance of GPSS: IGPSS. IGPSS is designed specially for expressing and enforcing privacy control policies and privacy preferences. An access request is evaluated in a way such that both organization’s privacy control policies and information donors’ privacy preferences are respected. We also provide an illustrating example that shows how our privacy protection system solves practical privacy problems.

The second privacy protection approach we proposed is based on hierarchical encryption. First, we proposed a new hierarchical encryption solution for general access control. That means besides implementing access control in a hierarchy, our solution also implements arbitrary access control needs. Our solution is based on the Chinese Reminder Theorem (CRT) and has two categories: data based solution and key based solution. By comparing with a recent study in hierarchical encryption (the RRN system), we have shown that our solution has the following prominent advantages: flexible, efficient and capable of generating smaller ciphertexts. Second, we have developed a novel way to protect privacy based on our hierarchical encryption solution. We have customized a hierarchical encryption scheme that is based on RSA and AES. Then, we have proposed two private data encryption algorithms. In our private data protection system, it is an individual’s responsibility to generate his/her encrypted private data. Therefore the individual is capable of control his/her privacy. Each piece of private data is encrypted in a way such that only legitimate users (or groups) can decrypt it. Unauthorized users, including unauthorized administrators, cannot decrypt it.
The third privacy protection approach we proposed is based on digital tickets. We identified the advantages of using digital tickets for privacy control. In particular, we have illustrated that these inherent properties of digital tickets make them fit for privacy management in a distributed, distrustful environment. Inspired by these advantages, we proposed a digital ticket privacy control system. In our system, information collection organizations cannot read the content of private data because private data is encrypted before its submission. Every access to private data is controlled by a digital ticket. Private data is only disclosed to its authorized users at the last possible stage after all restrictions are satisfied. Later on, obligations of every access must be “paid” and consistent with restriction written on tickets. Otherwise, users’ credibility will be decreased. In this approach, the capability of customizing and issuing digital tickets enables an individual to control his/her privacy. The digital ticket technique we use supports many features, such as number of times to be consumed, ticket expiry etc., which make elaborate privacy control realistic. In comparison with a recent advance of privacy control system that utilizes Kerberos tickets (the LO system), our system does not require a trusted third party to store private data. Moreover, instead of using Kerberos tickets, our system utilizes the idea of generalized digital ticket framework that supports more features.

In conclusion, in our privacy protection approaches information donors’ privacy preferences are greatly respected. Even after the submission of their private data, individuals are capable of control when, how and by whom the private data is accessed. Moreover, our approaches have some other advantages (e.g. generality, flexibility), which have been discussed in this thesis. Finally it is worth mentioning that there does not exist one super solution that can solve the privacy problems perfectly. All privacy protection approaches have their limitations.

(1) In access control based privacy protection approach, because all private data is stored in plaintext form, the storage of the data must be fully trusted. This raises some problems. First, it does not protect private data against malicious insiders, e.g. a corrupt administrator. Second, it does not protect private data once its storage is compromised.
(2) In hierarchical encryption based privacy protection approach, the storage of the data does not need to be trusted because data is cryptographically protected. However, these protections come at some computational costs.

(3) In digital ticket based privacy protection approach, private data is also protected by encryption. Therefore, the storage of the data does not need to be trusted. However, using of cryptographic mechanisms also makes this approach computationally more intensive.

In the future, we would imagine some hybrid privacy protection approaches, which combine different privacy protection mechanisms to achieve better privacy protection at reasonable costs.


