

國立交通大學

資訊科學系

碩士論文

具偽裝效果及驗證功能之文字型文件資訊分享的研究



**A Study on Information Sharing of Text-type Documents with
Steganography and Authentication Capabilities**

研究生：黃貴笠

指導教授：蔡文祥 教授

中華民國九十三年六月

具偽裝效果及驗證功能之文字型文件資訊分享的研究

A Study on Information Sharing of Text-type Documents with
Steganography and Authentication Capabilities

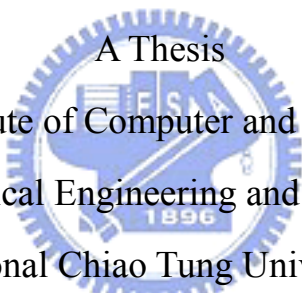
研究生：黃貴笠

Student: Kuei-Li Huang

指導教授：蔡文祥

Advisor: Wen-Hsiang Tsai

國立交通大學
資訊科學研究所
碩士論文



A Thesis
Submitted to Institute of Computer and Information Science
College of Electrical Engineering and Computer Science
National Chiao Tung University
In Partial Fulfillment of the Requirements
For the Degree of
Master
In
Computer and Information Science
June 2004
Hsinchu, Taiwan, Republic of China

中華民國九十三年六月

具偽裝效果及驗證功能之文字型文件資訊分享的研究

研究生：黃貴笠

指導教授：蔡文祥 教授

國立交通大學電機資訊學院
資訊科學研究所

摘 要

秘密分享是一種資訊隱藏的技術，能將一份秘密資料轉換成多份分享並將之分發給參與者保管。之後，蒐集一定份數以上的分享可將原始秘密資料回復。偽裝學是另一種資訊隱藏技術，能將一份資料編譯或轉換成某種格式的檔案或檔案的一部分，達到保護一份資料的目的。而驗證則是保證資料完整性和真確性的資訊隱藏技術。本研究提出了一些具偽裝效果和驗證功能的文字型文件的資訊分享方法。首先提出的是，具偽裝效果和驗證功能的純文字文件之資訊分享方法，利用邏輯運算對秘密文件做秘密分享，並將分享轉換成利用有意義句子形成的英文文件，達到偽裝的目的。並且將驗證訊號藏入這些英文文件內，達到驗證的目的。接著提出一具偽裝效果的 HTML 文件秘密分享方法，將分享偽裝成一份 HTML 文件，並提出一個用來驗證這些偽裝成 HTML 的文件的新方法。這兩個方法合起來利用合作式分享運算對一份秘密 HTML 文件中的內容做分享，將分享偽裝成跟秘密文件具有相同格調的 HTML 文件，而內容部分則用另一份藏入分享資料和驗證訊號的內容來取代。最後針對電子郵件格式的文件，提出一具偽裝效果的階層式秘密分享的方法和一用來驗證電子郵件形式分享的方法。這些方法對一份秘密電子郵件文件中每一構成要素做秘密分享、偽裝和驗證，而保留了原秘密文件中構成要素的架構。這些方法根據不同格式內容的構成要素，將含有分享資料和驗證訊號藏在裡面的構成要素用來取代原來的構成要素，形成一份份的分享。實驗結果證明了這些方法的可行性。

A Study on Information Sharing of Text-type Documents with Steganography and Authentication Capabilities

Student: Kuei-Li Huang

Advisor: Dr. Wen-Hsiang Tsai

Department of Computer and Information Science
National Chiao Tung University

ABSTRACT

Secret sharing is a kind of data hiding technique that transforms secret data into shares, which can be distributed to participants to keep and collected to recover the original secret. Steganography is another kind of data hiding technique that translates digital data into certain formats with difference appearances for protection of original data. And authentication is a technique for assuring the fidelity and integrity of protected digital data. In this study, secret sharing methods for text-type documents with steganography and authentication capabilities are proposed. A secret sharing method for pure texts with steganography and authentication capabilities is first proposed. The method shares a secret text by exclusive-OR operations. For steganographic effects, the shares are translated into meaningful sentences in which certain authentication signals using spaces are hidden for authentication. Next, a method for sharing HTML documents with steganographic effects is proposed, and a new authentication method for verifying HTML shares is described. The two methods together share the contents of a secret HTML by cooperative sharing operations, and retain the style of the secret HTML in the shares by replacing the contents of the secret HTML with fake contents imperceptibly containing share data and authentication signals. Finally, a hierarchical secret sharing method for e-mails and an authentication method for e-mail shares are proposed. The methods are applied to the components of a secret e-mail, and make the framework of the e-mail shares identical to that of the secret e-mail. The methods substitute fake components with share data

for the original e-mail components to form shares. And depending on the content of each component, corresponding authentication signals are embedded and an appropriate authentication process conducted. Experimental results are also included, which show feasibility of all the proposed methods.



ACKNOWLEDGEMENTS

The author is in hearty appreciation of the continuous guidance, discussions, support, and encouragement received from his advisor, Dr. Wen-Hsiang Tsai, not only in the development of this thesis, but also in every aspect of his personal growth.

Thanks are due to Mr. Chih-Hsuan Tzeng, Mr. Chang-Chou Lin, Mr. Chih-Jen Wu, Mr. Tsung-Yuan Liu, Mr. Cheng-Jyun Lai, Mr. Yen-Chung Chiu, Miss Yen-Lin Chen, Mr. Wei-Liang Lin, Mr. Yi-Chieh Chen and Mr. Kuei-Li Huang for their valuable discussions, suggestions, and encouragement. Appreciation is also given to the colleagues of the Computer Vision Laboratory in the Department of Computer and Information Science at National Chiao Tung University for their suggestions and help during his thesis study.

Finally, the author also extends his profound thanks to his family for their lasting love, care, and encouragement. He dedicates this dissertation to his parents.

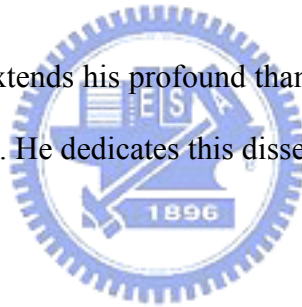


TABLE of CONTENTS

ABSTRACT(in Chinese)	i
ABSTRACT(in English)	ii
ACKNOWLEDGEMENTS	iii
TABLE of CONTENTS	v
LIST OF FIGURES	viii

Chapter 1 Introduction	1
1.1 Motivation.....	1
1.2 Review of Related Works	2
1.2.1 Review of Secret Sharing Methods.....	2
1.2.2 Review of Steganographic Methods for Text-Type Documents	3
1.3 Overview of Proposed Methods.....	4
1.3.1 Definitions of Terms	4
1.3.2 Brief Descriptions of Proposed Methods.....	5
1.4 Thesis Organization	9
Chapter 2 Proposed Techniques of Information Sharing of Text-Type Documents with Steganographic Effects and Authentication Capabilities	10
2.1 Introduction.....	10
2.2 Review of Adopted Techniques	10
2.2.1 Secret Sharing Technique by Exclusive-OR Operations	11
2.2.2 Secret Sharing Technique by Cooperative Sharing Operations.....	11
2.2.3 Hierarchical Secret Sharing Technique.....	13
2.2.4 Data Hiding for Text Documents	15
2.3 Proposed Techniques and Contributions.....	16
2.3.1 Data Magnitude Control by Modulus Adjustment.....	16
2.3.2 Steganographic Technique for Share Data by Use of Simple Sentences.....	19
2.3.3 Steganographic Technique for Text Components in HTML Documents	22
2.3.4 Steganographic Techniques for Non-text Components in HTML Documents	24
2.3.5 Steganographic and Authentication Techniques for Header Components in E-Mails	27
2.3.6 Authentication Technique for Verifying Share Data.....	31
Chapter 3 Secret Sharing Method with Steganographic Effects and	

Authentication Capability for Pure Text Documents	34
3.1 Introduction.....	34
3.2 Overview of Proposed Method	35
3.2.1 Secret Pure Text Sharing Process.....	35
3.2.2 Secret Pure Text Recovery Process.....	36
3.3 Proposed Detailed Processes for Sharing Secret Pure Texts.....	37
3.3.1 Use of Exclusive-OR Operations for Sharing.....	38
3.3.2 Steganographic Technique for Last Piece of Share Data.....	39
3.3.3 Steganographic Technique for Authentication of Text Shares	40
3.4 Experimental Results	42
3.5 Discussions and Summary	42
Chapter 4 Secret Sharing Method with Steganographic Effects for HTML Documents	46
4.1 Introduction.....	46
4.1.1 Properties of HTML Documents.....	46
4.1.1.1. In-tag Text.....	47
4.1.1.2. Outside-tag Text.....	47
4.1.2 Processes of Proposed Method	48
4.1.2.1. Secret HTML Sharing Process.....	48
4.1.2.2. Secret HTML Recovery process	50
4.2 Proposed Detailed Processes for Sharing Secret HTML Documents	51
4.2.1 Process for Sharing the Text Component in Secret HTML	52
4.2.2 Process for Sharing Non-Text Components in Secret HTML	54
4.3 Experimental Results	57
4.4 Discussions and Summary	58
Chapter 5 Steganographic Method for Tamper Proofing of HTML Shares	62
5.1 Introduction.....	62
5.1.1 Idea of Proposed Approach.....	62
5.1.2 Overview of Processes.....	63
5.1.2.1. Authentication signal embedding process	63
5.1.2.2. Authentication Signal Extraction and Verification Process .	64
5.2 Proposed Detailed Processes for Authentication of HTML Shares	64
5.2.1 Non-text Component Authentication	64
5.2.2 Text Component Authentication	68
5.3 Experimental Results	70
5.4 Discussions and Summary	71
Chapter 6 Hierarchical Secret Sharing with Steganographic Effects for E-mail Documents	73

6.1	Introduction.....	73
6.1.1	Properties of E-mail Documents.....	73
6.1.1.1.	Overview of e-mail format: multipurpose internet mail extensions (MIME).....	74
6.1.1.2.	Main components: e-mail header, content, and attachment.....	74
6.1.1.3.	Content transfer encoding methods.....	75
6.1.2	Overview of Proposed Processes.....	76
6.1.2.1.	Secret E-mail Sharing process.....	76
6.1.2.2.	Secret E-mail Recovery process.....	77
6.2	Proposed Detailed Processes for Sharing Secret E-mail.....	78
6.2.1	Sharing E-mail Header.....	81
6.2.2	Sharing Content Component.....	83
6.2.3	Sharing Attachments.....	85
6.3	Experimental Results.....	85
6.4	Discussions and Summary.....	86
Chapter 7	Steganographic Method for Tamper Proofing of E-mail Shares.....	90
7.1	Introduction.....	90
7.1.1	Proposed Ideas.....	90
7.1.2	Processes of Proposed Method.....	91
7.1.2.1.	Authentication signal embedding process.....	91
7.1.2.2.	Authentication process.....	93
7.2	Proposed Detailed Processes for authenticating E-mail Shares.....	94
7.2.1	E-mail Header Authentication.....	97
7.2.2	Content and Attachment Authentication.....	97
7.3	Experimental Results.....	99
7.4	Discussions and Summary.....	99
Chapter 8	Conclusions and Suggestions for Future Works.....	104
8.1	Conclusions.....	104
8.2	Suggestions for Future Works.....	106
REFERENCES	107

LIST OF FIGURES

Figure 2.1 A hierarchical secret sharing example.	15
Figure 2.2 An example of hiding data in a text document.	16
Figure 2.3 A hierarchical sharing example after applying data magnitude control.	20
Figure 2.4 Flowchart of share data translation process.	20
Figure 2.5 Flowchart of creating authentication capability.	32
Figure 2.6 Flowchart of authenticating process.	33
Figure 3.1 Flowchart of secret pure text sharing process.	35
Figure 3.2 Flowchart of process of authentication signal embedding.	36
Figure 3.3 Flowchart of authenticating process.	37
Figure 3.4 Flowchart of process of recovering secret pure text.	37
Figure 3.5 Flowchart of process of embedding authentication signals into a stego-text.	41
Figure 3.6 Flowchart of verification process of share text.	41
Figure 3.7 The secret pure text.	42
Figure 3.8 Stego-texts (a) through (b) articles selected from an article database.	43
Figure 3.9 The last stego-text.	44
Figure 3.10 Shares (a) the last share; (b) one of the other shares.	44
Figure 3.11 Recovered secret pure text.	45
Figure 4.1 An HTML document. (a) the source code; (b) the display on a browser.	48
Figure 4.2 Flowchart of component extraction and sharing.	49
Figure 4.3 Flowchart of the steganography processes for the share data of components.	50
Figure 4.4 Flowchart of the process of stego-HTML document creation.	50
Figure 4.5 Flowchart of the process of share data extraction of components.	51
Figure 4.6 Flowchart of secret component recovery processes.	51
Figure 4.7 Flowchart of secret HTML recovery process.	52
Figure 4.8 A secret HTML document.	59
Figure 4.9 stego-HTML documents (a) through (b) stego-HTML documents generated from the secret HTML document in Figure 4.8.	60
Figure 4.10 Recovered secret HTML document.	60
Figure 4.11 Image components. (a) secret image component; (b) the corresponding stego-image component of the first stego-HTML document; (c) the corresponding stego-image component of the second share HTML.	61
Figure 5.1 Flowchart of process of embedding authentication signals.	65

Figure 5.2 Flowchart of share authenticating process.....	66
Figure 5.3 A stego-HTML document.	71
Figure 5.4 A share of the HTML document in Figure 5.3.....	72
Figure 5.5 A verified share of a tampered HTML share.	72
Figure 6.1 Conceptual expression of MIME format.	75
Figure 6.2 Flowchart of component extraction and sharing process.	77
Figure 6.3 Flowchart of steganographic effect creation process.....	78
Figure 6.4 Flowchart of process of share data extraction.	81
Figure 6.5 Flowchart of recovery process of secret e-mail.....	81
Figure 6.6 A secret e-mail.	88
Figure 6.7 Hierarchical framework relationship among four participants.....	88
Figure 6.8 Stego-e-mails. (a) through (b) Two of four stego-e-mails.	89
Figure 6.9 Recovered secret e-mail.....	89
Figure 7.1 Flowchart of share data extraction process.....	92
Figure 7.2 Flowchart of processes of authentication signal embedding.	92
Figure 7.3 Flowchart of process of share creation.	93
Figure 7.4 Flowchart of component extraction process.	93
Figure 7.5 Flowchart of authenticating process.	94
Figure 7.6 Flowchart of the process of verified share e-mail creation.....	94
Figure 7.7 A stego-e-mail.....	100
Figure 7.8 The share of the stego-e-mail in Figure 7.7.....	101
Figure 7.9 The contents of the HTML attachment.	101
Figure 7.10 A successfully verified share.	102
Figure 7.11 A verified share with an improper key.	102
Figure 7.12 The contents of the HTML attachment in the verified share in Figure 7.11.	103
Figure 7.13 The verified share of the share in Figure 7.8, which is tampered.....	103

Chapter 1

Introduction

Motivation

With the advance of the Internet and computer technologies, more and more documents are transformed into digital versions and can be transmitted on the Internet. For an organization, it is convenient to exchange messages and documents via the Internet. However, because digital documents can be copied easily and quickly, important documents, such as contracts, conference records, technical reports, source codes, strategic decisions, etc, which must be kept only by part of the staff of the organization should be dealt with carefully. One method to manage this kind of document is *secret sharing*. Although secret sharing has been applied to various kinds of files, such as images and videos, it is worthy to apply it to text-type documents, one kind of file, for their frequent usage.

Secret sharing is a way to transform secret data into many meaningless parts which are then assigned to participants and to recover the secret data by collecting a sufficient number of parts from the participants. Because of the meaninglessness of each part kept by the participants, they will be worried about where their own parts can be hidden to keep curious users from accessing their parts and how secure a network environment would be to provide secure transmissions for these important parts. If each part can be covered by or hidden in other meaningful media, the possibility that these parts bring about other users' awareness will be relatively lower. This effect can be achieved by applying techniques of *steganography* to the meaningless parts. How to accomplish such a goal is an interesting issue.

Although each part is meaningful and can avoid a certain degree of the awareness of curious users, we cannot know whether the part is changed or not. The part may be changed accidentally while, for example, being transmitted on an unstable network; or the part may be modified intentionally by an intended user. Therefore, authenticating the parts is necessary for ensuring the integrity of each part. How to conduct authentication of parts is another interesting issue which will be investigated in this study.

Review of Related Works

1.1.1 Review of Secret Sharing Methods

Secret sharing is a way to encrypt and distribute secret data into several parts so that each part, kept by a participant, contains only partial information of the secret. The secret data can be recovered if a pre-defined group of parts is collected. Moreover, collecting a group of parts different from the pre-defined group cannot recover the secret information.

Shamir [1] was the first to propose the concept of secret sharing in his (k, n) -threshold method, where n denotes the number of participants and the threshold k specifies the minimum number of parts in the pre-defined group. By this method, secret information is encrypted and then distributed into n parts, which are assigned to n participants, respectively. If and only if k or more than k participants get together, the secret information can be recovered by a certain method. Subsequently, many related topics were studied [2]-[7] and various kinds of secret sharing methods were proposed [8]-[13]. Nevertheless, these proposed methods are only suitable for data of short lengths, such as passwords, encryption and decryption keys, and so on.

Applied to images, many secret sharing methods based on the (k, n) -threshold method were proposed [14]-[18]. Especially, an efficient secret sharing method using exclusive-OR operations was proposed by Lin and Tsai [19]. This method is an (n, n) -threshold method. It simply applies the exclusive-OR operation to a secret image as well as $n-1$ images, all of the size of the secret one, to generate the n th image. The $n-1$ images and the n th image altogether are regarded as shares and are distributed to n participants, respectively. The secret image can be quickly recovered by exclusive-ORing the n images held by the n participants.

Based on ideas from company organizations, Lin and Tsai [20] proposed a method of *hierarchical* secret sharing, which is a new concept of sharing secret among groups of participants. Three types of secret sharing, cooperative sharing, independent sharing and dominant sharing, were proposed for realizing the concept. The method first specifies a hierarchical structure as a tree, in which each non-leaf tree node denotes one of the three operations and each leaf node denotes one of the parts. According to the tree, secret information can be encrypted and distributed into parts corresponding to the leaf nodes and recovered as well.

1.1.2 Review of Steganographic Methods for Text-Type Documents

A steganographic method for text-type documents is to embed message data into a text-type document to avoid awareness of the message. The capacity of the redundancy information of text-type documents for hiding data is smaller than that of images or videos. Methods for hiding data in text-type documents may embed data into the text itself or in the language describing the text format.

Wayner [21] proposed a method that creates cover texts according to the secret message using content-free grammars and selects “proper” productions that are relatively more meaningful than the improper ones, to achieve steganographic effect for covert communication. Therefore, a cover text itself is also the encoded secret message. The secret message can be decrypted by parsing the transmitted covert texts.

As for hiding data in the cover text, Maxemchuk et al. [22]-[23] described a steganographic method for text-type documents. Secret data are embedded by adjusting the distance between two successive between-word spaces or between two successive interline spaces. For example, if the distance between two successive interline spaces is larger than a threshold, the embedded information is “1”; otherwise “0.” For a soft-copy text, Bender et al. [24] proposed a similar method that exploits inter-sentence spacing, end-of-line spacing, and between-word spacing to embed secret information. For instance, a single between-word space means that “0” is embedded in and a double between-word space is interpreted as “1”.

For documents of complex formats, Chang and Tsai [25] proposed a method for covert communication using HTML documents. Because a web browser does not display a sequence of spaces following a leading space and tags in an HTML document, the method can encode secret information by adjusting the size of between-word spaces and the expression of tags.

Overview of Proposed Methods

1.1.3 Definitions of Terms

1. *Document*: A document is a text-type file, such as a piece of pure text, an HTML file, or an e-mail.

2. *Component*: A component of a document is an independent part in the document, which another part of the same format can be substituted for directly. For instance, text and non-text parts in HTML's, headers and attachments in e-mails are called *Components*.
3. *Secret*: A secret is certain information that is important and should be preserved properly.
4. *Share data*: share data are the secret sharing result of a secret.
5. *Stego-document*: A stego document is a document with share data embedded in. For example, An HTML document with share data embedded in is called a *stego-HTML* document and an e-mail with share data embedded in is called *stego-e-mail*.
6. *Share*: A share is the result after applying steganographic and authentication techniques on a piece of share data.



1.1.4 Brief Descriptions of Proposed Methods

A. Proposed Techniques of Information Sharing of Text-Type Documents with Steganographic Effects and Authentication Capabilities

Some possible techniques for sharing text-type documents with steganographic effects and authentication capabilities have been reviewed previously. Accordingly, several information sharing, steganographic effect creation and authentication techniques for text-type documents are proposed independently and in detail. These techniques will be selected properly and applied cooperatively later to documents of three different text-types, namely, pure text, HTML, and e-mails.

B. Proposed Method for Sharing Pure Text Documents with Steganographic Effect and Authentication Capability

Several techniques are combined in this study for sharing pure text documents with steganographic effects and authentication capabilities. The first proposed technique of sharing secrets is based on exclusive-OR operations and encodes and distributes a secret pure text document into several pieces of share data. The second proposed technique translates a piece of share data into several simple sentences to form a meaningful text, a stego text. The purpose of this technique is to make each piece of share data meaningful. Then, by inter-word, inter-sentence and inter-line spacing, authentication signals can be hidden in the stego text by the third proposed technique to generate a share, and accordingly the share can be authenticated later. By combining the three techniques together, a secret pure text document can be shared among the participants of the secret pure text document and each share can be verified.

C. Proposed Secret Sharing Method for HTML Documents with Steganographic Effect

One secret sharing technique for HTML documents and two steganographic methods for share data are combined to create a secret sharing method for HTML documents in this study. The method parses and locates components in an HTML document and then encrypts them into pieces of share data by the use of a certain modified version of the cooperative sharing operation mentioned previously. Since users of web browsers cannot be aware of certain symbols, such as spaces, new-lines, tabs, and text inside tags in HTML's, a steganographic method for HTML shares is also proposed and transforms the pieces of share data into HTML documents of the style of the secret HTML documents by exploiting some properties mentioned above. To simulate the style of the secret HTML, components of the secret HTML, such as texts, images, videos, etc, are replaced by cover ones of the same type, respectively. For instance, in a HTML document, an image link, which can show an image on

browsers, is substituted by another link which will display another image on the browsers. In addition, each cover component also contains the corresponding piece of share data.

D. Proposed Steganographic Method for Tamper Proofing of HTML Share Documents

Two steganographic methods are proposed in this study for authenticating HTML share documents. Authentication signals of share data are generated by segmenting share data into several strings of one length according to the size of the hiding space and exclusive-ORing the strings into a string. The properties of HTML documents are used again to hide authentication signals in tags, and inter-word, inter-line, and inter-sentence spaces. For text components, their authentication signals are embedded in the between-word spaces, where the corresponding share data are also embedded. By specifying the number of authentication signals embedded in each between-word space and the position relation between share data and authentication signals, the authentication technique can work well. For non-text components, the authentication signals of share data are hidden in tags of the HTML document with share data embedded in.

E. Proposed Hierarchical Secret Sharing Method for E-mail with Steganographic Effect

A hierarchical secret sharing method for secret e-mails and three cooperative steganographic methods for e-mail shares are proposed in this study. A secret e-mail contains many components which can be sorted the components into three kinds, header, body, and attachment. For header components, important information in each component is extracted and shared by a modified hierarchical secret sharing technique. Body components can be classified into two types: pure text and binary stream. As for attachment components, there are three types of the components, e.g. e-mail, pure text,

and binary stream. By applying the modified method of hierarchical secret sharing, body and attachment components of pure-text type and binary-stream type can be shared as well. The e-mail elements can be treated as a new secret e-mail and processed by the proposed method. The steganographic method refers to the framework of the secret e-mail, and replaces a header component with a cover header, a component of pure-text type with another pure text, and a component of binary-stream type with an HTML document. Component contains their corresponding share data so that illicit users opening the stego e-mail made up of the components cannot nose out anything different.

F. Proposed Steganographic Method for Tamper Proofing of E-mail share Documents

Three steganographic methods for tamper proofing of e-mail share documents are combined and proposed in this study. Each component's authentication signals are generated independently by the authentication signal generation method, as mentioned previously while generating authentication signals for stego HTML documents, and embedded into the cover host of the components. For header components, because an e-mail parser discards the string of space and tab symbols at the rear of lines in a header, the authentication signals are encoded in this study into several strings of tab and space symbols, which are concatenated at the rear of the lines. For pure-text type components, the between-word spacing scheme of the data hiding method proposed by Bender et al. [24] is used to embed authentication signals. For binary stream elements, an element's authentication signals are inserted in the tags of the corresponding cover host, an HTML document. After hiding authentication signals into the e-mail in which share data are embedded, a share is generated. While proofing, the method examines a share component by component and fingers out which components are tampered with.

Thesis Organization

In the remaining chapters of this thesis, the adopted techniques are reviewed and the proposed techniques are described briefly, in Chapter 2. The proposed secret sharing method for pure text documents with steganographic effects and authentication capabilities is described in Chapter 3. The proposed secret sharing method for HTML documents with steganographic effects is described in Chapter 4. The proposed steganographic method for assuring the fidelity of HTML shares is described in Chapter 5. In Chapter 6, the proposed method for sharing e-mail documents by modified hierarchical secret sharing method and for creating steganographic effects on share data of secret e-mails are described. In Chapter 7, the proposed steganographic method for temper proofing of e-mail shares is described. And in Chapter 8, conclusions will be made and future works for further study will be suggested.



Chapter 2

Proposed Techniques of Information Sharing of Text-Type Documents with Steganographic Effects and Authentication Capabilities

Introduction

In order to achieve the goal of this study, several related techniques are required. In this chapter, the adopted techniques and the proposed ones for realizing information sharing of text-type documents of different formats with steganographic effects and authentication capabilities are described.

In Section 2.2, the adopted techniques are reviewed in detail first. These techniques are not necessarily suitable for text-type documents. However, with modifications the techniques work well on them. In Section 2.3, the proposed techniques are described. Some of them are derived from the adopted techniques. Some of the proposed techniques are useful for general text-type documents and the others are suitable for texts of special types.

Review of Adopted Techniques

Four techniques will be reviewed in the following. The first technique is to share secret images by the exclusive-OR operation. A discussion on the feasibility of applying the technique to texts will also be made. The second and third ones are about the so-called hierarchical sharing. Brief discussions on their applications to text

sharing will be presented. The last one is about data hiding in text documents.

2.1.1 Secret Sharing Technique by Exclusive-OR

Operations

In this section, a technique of secret sharing using the exclusive-OR operation proposed by Lin and Tsai [19] will be reviewed. Originally, this technique is applied to share images. A secret image is exclusive-ORed pixel by pixel with some randomly selected images, all of the same size as that of the secret image. The selected images and the resulting image which is meaningless are regarded as shares and distributed to secret sharing participants.

Due to the constraint that the sizes of the selected images must be the same as that of the secret image, the technique can be utilized for texts only under the condition that the lengths of secret texts and those of selected texts are the same. This requirement can be relaxed for pure texts, which will be described in Chapter 3.

2.1.2 Secret Sharing Technique by Cooperative

Sharing Operations

The cooperative sharing operation is one of the three main operations of hierarchical secret sharing proposed by Lin [20]. A property of the operation is that only when all shares are collected can the secret be recovered. In addition, secret sharing by cooperative sharing operations itself is an (n, n) -threshold method.

The function of cooperative sharing for two participants is shown as follows:

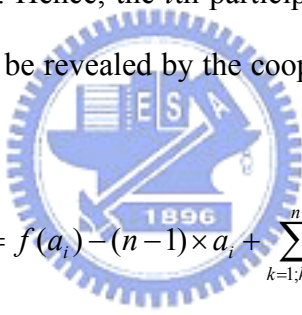
$$f(x) = (x - a) + (x - b) + s,$$

where s is the secret, and a and b are randomly selected integers. And $(a, f(a))$ and $(b, f(b))$ are distributed to the two participants as their own share data of s , respectively. The secret s can be recovered only when the two participants cooperate in the following way. From the viewpoint of the participant who keeps $(a, f(a))$, s equals to $f(a) - a + b$. Instead, from the other participant's point of view, s can be obtained by computing $f(b) - b + a$.

For more than two participants, the function can be revised as follows:

$$f(x) = \sum_{i=1}^n (x - a_i) + s,$$

where n is the number of participants, s is the secret, and all a_i are randomly selected integers, where $i = 1, 2, \dots, n$. Hence, the i th participant keeps share data $(a_i, f(a_i))$, and for each participant, s can be revealed by the cooperative recovery formula in the following:



$$s = f(a_i) - (n-1) \times a_i + \sum_{k=1; k \neq i}^n a_k.$$

An illustrative example is described as follows. Let the participant number n be 3, the secret s be 24, and the randomly selected integer a_i be in order 129, 3, and 10, for $i = 1, 2, 3$. The three participants' share data are $(129, 269)$, $(3, -109)$, $(10, -88)$, respectively. For the participant who keeps the share data $(3, -109)$, according to the cooperative sharing recovery function, the computed value is $-109 - [(3 - 1) \times 3] + [129 + 10] = 24$, which equals to s .

While trying to apply the technique to texts, a critical issue is encountered. As seen in the example illustrated above, the minimal space size for storing a piece of share data is not consistent. For share data $(129, 269)$, three bytes will be used; as for either of the remaining ones, two bytes will be used. How to limit the size of each

piece of share data by controlling their magnitudes will be mentioned in 2.3.1.

2.1.3 Hierarchical Secret Sharing Technique

Hierarchical secret sharing is a new concept of secret sharing. From a behavior point of view, a senior person initiates the secret sharing activity among participants; the new concept is first to share the secret among several groups formed by the participants and then, regarding each piece of share data as the *group secret* of the corresponding group, to continuously share the group secret of each group among the smaller groups formed by the participants of the group, until each participant of the secret gets his/her own share data. However, by applying three different sharing operations, namely, the cooperative sharing operation, the independent sharing operation, and the dominant sharing operation, the concept of “hierarchical” secret sharing can be truly realized. To understand the technique of hierarchical secret sharing, the three main sharing operations are first described and a description of how the hierarchical secret sharing technique works follows.

The description of the cooperative sharing operation will be given in 2.2.2 and is skipped here. The concept of independent sharing operation is that each participant knows the secret. Its corresponding function is described in the following:

$$f(x) = \prod_{i=1}^n (x - a_i) + s,$$

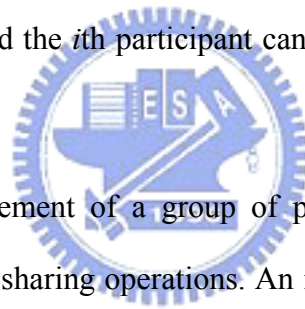
where s is the secret, n denotes the number of participants, and a_i denotes a randomly selected integer for the i th participant. The i th participant keeps share data $(a_i, f(a_i))$, where $f(a_i)$ equals to s .

The concept of dominant sharing operation is that only one “dominant”

participant can know the secret and the others can get the secret only after the permission of the dominant participant. The function of dominant sharing is as follows:

$$f(x) = \prod_{i=1}^n (x - a_i) + (x - a_1) + s ,$$

where s is the secret, n denotes the number of participants, and a_i denotes a randomly selected integer, for $i = 1$ through n . The first participant keeping share data $(a_1, f(a_1))$ knows the secret because $f(a_1)$ equals to s , while the other participants must get the permission of the first participant to know the secret. Note that the i th participant, except the first one, can compute the secret by the formula $s = f(a_i) - a_i + a_1$. That is, if these participants are trustworthy, the first participant can transmit just a_1 to the i th participant via the Internet and the i th participant can know the secret without caring about the secure problem.



According to the requirement of a group of participants, the group secret is processed by one of the three sharing operations. An illustrative example is presented in Figure 2.1.

Suppose that Participants 1 and 2 are two managers of a company, Participant 3 is the president of the company, and Participant 4 is the secretary of the president. Now, assume that a secret of the company is 5. The secret can be known only under the condition of acquiring the president's and one of the two managers' agreements. In order to avoid failure of secret recovery coming from the absence of the president and the hard time to reveal the secret, the secretary is standby for such the condition. Usually, the president, Participant 3, and one of the two managers, for example, Participant 1, can cooperate to get the secret after three recovery operations.

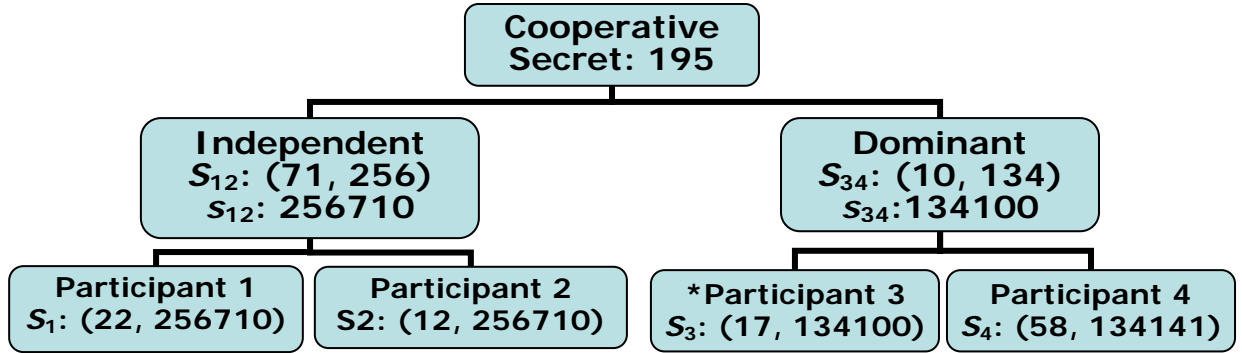


Figure 2.1 A hierarchical secret sharing example.

Firstly, Participant 1 can recover his/her and Participant 2's composite secret independently by simply extracting the second part of pairs. Secondly, Participant 3 can recover his/her and Participant 4's composite secret by himself/herself by directly taking the second part of his/her own share data out. Finally, Participant 1 and Participant 3 can use their secrets, which are just recovered, as shares to recover the secret by the cooperative recovery formula. Under the condition just mentioned, the absent president can send the first part of his/her share data to his/her secretary and then his/her secretary can get their composite secret as well by computation.

As discussed in Section 2.2.2, the data magnitude control problem also exists in the hierarchical secret sharing technique for real world applications. The method in Section 2.3.1 is proposed for solving this problem.

2.1.4 Data Hiding for Text Documents

Three techniques of data hiding that will be reviewed here are proposed by Bender et al. [24]. These techniques hide data into text documents through manipulations of three kinds of spaces, including inter-sentence space, end-of-line space, and inter-word space, in text documents. For instance, one space between two

successive words, sentences, or lines may be regarded to represent a “0”, while two spaces a “1”. Therefore, the article in Figure 2.2, for example, contains 8 bit data “01011110.”

W	e		a	r	e			T	h	e
	W	o	r	l	d	.			Y	o
u			a	r	e			m	y	
	s	u	n		s	h	i	n	e	.

Figure 2.2 An example of hiding data in a text document.

Proposed Techniques and Contributions

2.1.5 Data Magnitude Control by Modulus

Adjustment



In order to apply the cooperative sharing technique and the hierarchical secret sharing technique to text-type documents, some functions of the three operations are modified in this study. Consequently, the storage size of a share is exactly two times larger than that of the corresponding secret. That is, a share is stored in two bytes, while the secret is stored in one byte. In the following, the encountered problem is first described and investigated in detail. Then a technique of controlling the magnitude of shares is proposed. Finally, an example is given.

A piece of share data is of the form of a pair $(x, f(x))$, where x is randomly selected and the function f is one of the three functions of the three sharing operations. By inspection of these functions, the range of the first part of a pair can be controlled easily and the value range of $f(x)$ depends on the number of participants, the secret

value, and the first parts of the shares. Moreover, the value range of $f(x)$ is directly proportional to the number of participants. Although the second parts of the shares can be controlled by selecting the first parts of the shares carefully so that the distance between every two first parts is as short as possible, the significance of randomizing for sharing secret becomes meaningless. Furthermore, the probability of guessing the secret correctly by a brute force method will increase. How to control the magnitudes of share data without losing the meaning of randomization is described in the next paragraph.

By the concept of algebra, a set of all possible remainders of a prime divisor, an addition operator, and a multiplication operator can form a field, which has some good properties: (1) each element of the set has one and only one addition inverse and one and only one multiplication inverse; (2) there are one and only one addition unit element, and one and only one multiplication unit element. Let the prime be 7, for example. The remainder set of the prime divisor is $\{0, 1, 2, 3, 4, 5, 6\}$. Also let the addition and multiplication operators be the ordinary addition and multiplication operators, respectively, 0 be the addition unit element, and 1 be the multiplication unit element. Some examples are listed as follows:

1. $2 + 5_{(\text{mod } 7)} = 0;$

2. $4 \times 2_{(\text{mod } 7)} = 1.$

In the first example, 5 is called the addition inverse of 2 and vice versa; and in the second example, 4 is called the multiplication inverse of 2 and vice versa. Therefore, the magnitudes of share data can be restricted by modulus adjustment and the computation of recovering a secret can be performed correctly according these properties of the field. The functions of the three sharing operations are revised

accordingly in the following:

1. Revised Cooperative Operation: $f(x) = \left(\sum_{i=1}^n (x - a_i) + s \right)_{\text{mod } p}$,
2. Revised Independent Operation: $f(x) = \left(\prod_{i=1}^n (x - a_i) + s \right)_{\text{mod } p}$,
3. Revised Dominant Operation: $f(x) = \left(\prod_{i=1}^n (x - a_i) + (x - a_1) + s \right)_{\text{mod } p}$,

where s is the secret, n is the number of participants, p is a prime number, and a_i is a randomly selected integer from the remainder set of divider p . In the next paragraph, the proposed hierarchical secret sharing technique for texts will be described in detail.

Because a text is a byte-based file, the integer 257 is chosen as the divider p in this study. The value of the second part, $f(x)$, of a piece of share data now ranges from 0 to 256 after the modulo p operation, while the value of a byte ranges from 0 to 255. By dealing with the special case of value 256 in the second part of share data, two-byte space can store all possible values of a pair.

In cooperative sharing, because of the linearity of the function in the set of the selected random integers, the condition that the second parts of two pairs are identical implies the fact that the first parts of the two pairs are identical. Therefore, a constraint for the special case is specified as follows:

$$a_1 + \dots + a_n = [(n - 1) \times 256 + s]_{\text{mod } 257}$$

where s is the secret of one byte, n is the number of participants of s , and a_1, \dots, a_n are the first parts of the n pairs and different from each other, and range from 0 to 255. Under this constraint, the function value, the second part of a pair, of a_i can be limited to the interval from 0 to 255.

In dominant sharing, by inspection, the corresponding function is a linear function among the set of all randomly selected integers $\{a_1, a_2, \dots, a_n\}$. Moreover, changing the value of a_1 can influence all the function values of a_i while modifying the value of another randomly selected integer, say a_i , can only change the function value of a_i . Therefore, after specifying a_1 , the other integers a_2, \dots, a_n can be randomly selected under the condition that a_i must not be $(256 - s + a_1)_{\text{mod } 257}$ to avoid the function value to become 257.

As for independent sharing, if the secret value is not 256, the appearance of the special case is impossible. And the secret value is really impossible to be 256.

Using the example in 2.2.3 again, let's see how the revised hierarchical secret sharing technique works and what differences exist between the revised one and the original one in Figure 2.3.

In this example, the secret is first shared among two groups by cooperative sharing into two pieces of share data: (100, 201) and (94, 189). In turn, independent sharing is byte by byte applied to s_{12} , which is the composite secret of Participant 1 and Participant 2 and regarded as a two-byte secret. Next, the two-byte composite secret, s_{34} , of Participant 3 and Participant 4 is processed byte by byte by dominant sharing. Finally, each participant gets a four-byte piece of share data. Therefore, the hierarchical secret sharing technique for texts can run well. The recovery processes of the three sharing operations can also run correctly and well.

2.1.6 Steganographic Technique for Share Data by

Use of Simple Sentences

In this section, a technique of translating a piece of share data, which is almost

always meaningless, into several meaningful simple sentences is proposed. Firstly, a steganography procedure is shown. Secondly, the technique is described in detail. Finally, an illustrative example is described.

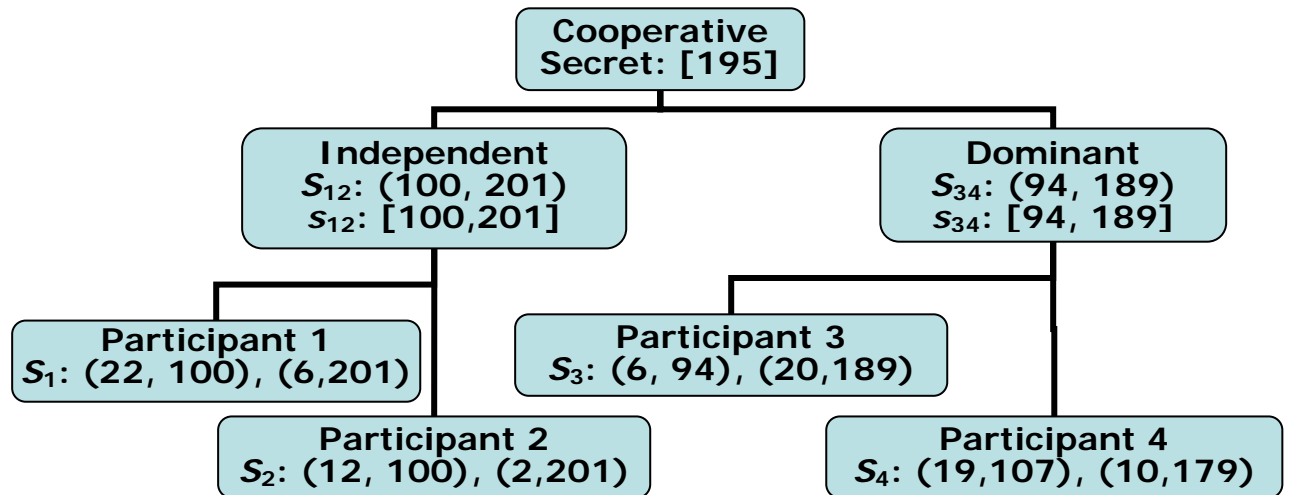


Figure 2.3 A hierarchical sharing example after applying data magnitude control.

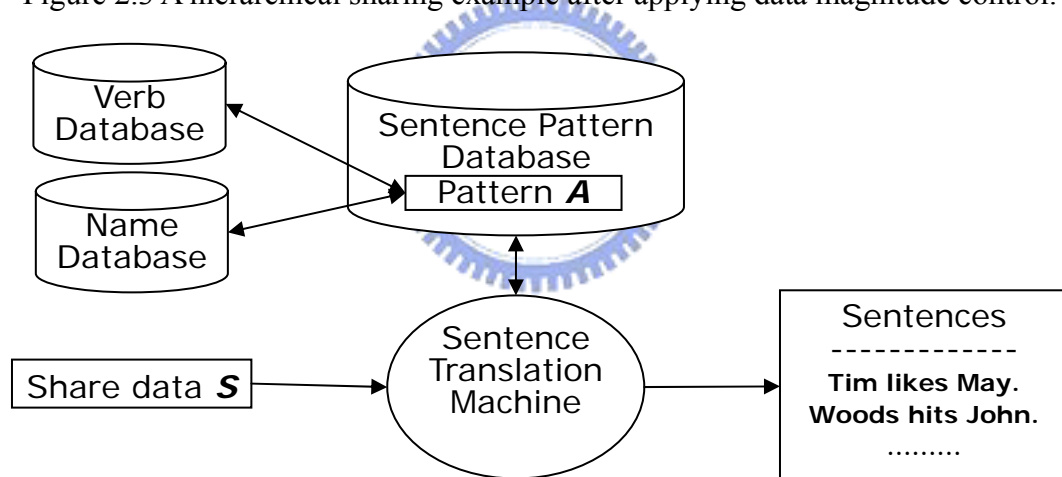


Figure 2.4 Flowchart of share data translation process.

In Figure 2.4, share data is first pushed into a simple *sentence translation machine* designed in this study. The machine divides the share data into several parts and encodes each part into a sentence according to its sentence pattern and its corresponding databases by an indexing technique. While recovering the share data, sentences are decoded by inverse indexing.

The technique uses one sentence pattern, the *subject-verb-object pattern*, and two corresponding databases, a *name database* and a *verb database*. Each database

contains 257 words, where the first 256 words are used for encryption and the last one is used as the *end-of-share-data word*. The translation procedure for encoding text share data is described in detail as follows.

Algorithm 2.1 *Translation from share data to simple sentences.*

Input: share data S .

Output: an encoded string S_e .

Steps.

1. Pad S with the “super” byte of the value 257, until the length of S is divisible by 3.
2. Divide S into three-byte strings.
3. For each string, replace the first byte and the third byte with names in the name database and the second byte with a verb in the verb database all by indexing.
4. For each word of the third bytes, concatenate an end-of-sentence symbol in the tail.
5. For each word, concatenate a space in the tail.
6. Concatenate all sentences into a string as S_e

The corresponding recovery procedure is as follows.

Algorithm 2.2 *Translation from simple sentences to share data.*

Input: a string of sentences S' .

Output: a decoded share data S_d .

Steps.

1. Divide the sentences in S' into many ordered sentences.
2. For each sentence, decode the first and the third words by inverse indexing using the name database and the second words by inverse indexing using

the verb database.

3. Concatenate these bytes according to the order of the sentences into a decoded string.
4. Remove the “supper” bytes in the tail of the decoded string to get S_d .

Now, an example is given. For convenience, let share data S be “HELLO.” First, S is divided into “HEL” and “LO θ ”, where θ denotes the “super” byte. After encoding by indexing, they become a sentence like “Tim likes May.” and “Woods hits John.” The name “John” is the end-of-share-data word. In the end, the encoded string S_e is “Tim likes May. Woods hits John.” In the decoding stage, S_e is first divided into two single sentences, “Tim likes May.” and “Woods hits John.”, and then, by inverse indexing and concatenation, a string “HELLO θ ” is acquired. Finally, the decoded share data S_d is “HELLO” by eliminating the “supper” byte.



2.1.7 Steganographic Technique for Text Components in HTML Documents

A steganographic technique for text components in HTML documents is proposed. According to a property of the HTML, text components that can be seen on browsers can be substituted by fake text components and the data of text components are translated and hidden in a substitute one without arousing any awareness of the hidden data. First, the behavior of text contents on a browser is described. Finally, the proposed steganographic technique is described.

The text components in an HTML document are the texts outside the tags and can be displayed on a browser. Only one space symbol between two successive words is displayed on a browser while, actually, a sequence of tab symbols, new-line

symbols, and space symbols are bundled into the position between the two consequent words in an HTML document. An additional symbol of ANSI code 0x0C can behave as the three symbols in the environment of the Internet Explore (IE) browser. For other browsers, some different symbols have the same property and can be used as the three symbols. These symbols can be put in use in the proposed steganographic technique. In this study, Internet Explore browser is used as the HTML browser.

Let a space symbol denote a “0,” a tab symbol denote a “1,” a new-line symbol denote a “2,” a symbol with ASCII code 0x0C denote a “3,” $L(x)$ be the length of text x and $N(x)$ be the number of inter-word, interline and inter-sentence spaces in text x . The proposed procedure for creating steganographic effects on important parts of an HTML document as a fake HTML document is as follows.

Algorithm 2.3 *Steganographic effect creation for text components.*

Input: an HTML document H and an English article A .

Output: an HTML document H_s .

Steps.

1. Extract every text content segment S_i between two successive tags in H .
2. For each text content segment S_i , perform the following steps.
 - (i) Cut an appropriate number of sentences in front of the remainder of A as C_i such that the difference between $L(C_i)$ and $L(S_i)$ is minimal.
 - (ii) Translate S_i into a string E_i made up of tab, new-line and space symbols and the symbol of ANSI code 0x0C.
3. For each inter-word, inter-line, or inter-sentence space in C_i , embed a space symbol and $\lceil N(C_i)/N(E_i) \rceil$ symbols of E_i into C_i to generate D_i .
4. Assign D_i to S_i .
5. Put back each extracted and processed segments S_i to form the result H_s .

The corresponding recovery procedure is described in the following.

Algorithm 2.4 *Recovery process of Algorithm 2.3.*

Input: an HTML document H .

Output: a recovered HTML document H_r .

Steps.

1. Extract every text content segment S_i between two successive tags in H .
2. Create an empty string S .
3. For each text content segment S_i , perform the following steps.
 - (i) Clean up S .
 - (ii) Extract all inter-word, inter-line, and inter-sentence sequences L_j of space symbols, new-line symbols, symbols of ANSI code 0x0C and tab symbols, except the leading symbol.
4. For each sequence L_j , append L_j to the rear of S .
5. Assign S to S_i .
6. Replace all extracted and processed segment S_i with the original ones in H to form the result H_r .

Although this technique is applied to the text components of an HTML document, it can also be utilized for the share data of the text components without any modification.

2.1.8 Steganographic Techniques for Non-text

Components in HTML Documents

A steganographic technique for non-text components in HTML documents is proposed. Non-text contents that can be displayed on browsers are of the form of links

in the tags of an HTML document. According to the concept of “dynamic” links, non-text contents can be replaced with a fake link and become part of the fake link. In the following, how a dynamic link works on the internet is introduced first, how to replace a link into a workable dynamic one that contains the original link is proposed then, and, in the end, an illustrative example is given.

A “static” link is the link which indicates directly the address of the corresponding source on the internet. However, a “dynamic” link is the link which contains the address of an agent in a server on the internet and information for the agent following the address. A dynamic link, for example, is of the form: “http://www.cis.nctu.edu.tw/~gis91568/agent?123456”, where the string before the question mark is the address of the agent and the string following the question mark is the information for the agent. After a browser gets the address of the agent from the link and informs the agent the information in the link, the agent returns the corresponding source according to the information. For example, a dynamic image link containing the address of an image database agent and the information of the image index in the database can let a browser know where the image database agent is on the internet and which image should be retrieved from the database according to the image index. Finally, the agent returns the image of the index in the image database.

Suppose that a multimedia database MD and an agent A of MD are created and that the address of the agent on the internet is a dynamic link ADD . The procedure of the proposed steganographic technique is described as an algorithm as follows.

Algorithm 2.5 *Steganographic effect creation for non-text components.*

Input: a secret link L , ADD .

Output: a dynamic link DL .

Steps.

1. Translate L into a string S in hexadecimal form byte by byte.
2. Create a dynamic link DL .
3. Set $DL=ADD?S$, where ‘?’ is the special symbol for separating the address ADD and its information S .

The string S is used as an index in the above algorithm. A dynamic link created in this way can be understood by browsers and work well on the internet.

For instance, an image link “http://tw.yahoo.com/a.jpg” is first translated into “687474703A2F74772E7961686F6F2E636F6D2F612E6A7067” in hexadecimal and the corresponding dynamic link is “http://www.nctu.edu.tw/agent?687474703A2F74772E7961686F6F2E636F6D2F612E6A7067”, where the address of the image database agent is “http://www.nctu.edu.tw/agent”.

The corresponding recovery procedure is as follows.

Algorithm 2.6 *Recovery process of Algorithm 2.5.*

Input: a dynamic link DL .

Output: a secret link L .

Steps.

1. Extract a string S from the parameter part of DL .
2. Translate S back to a string TS of symbols.
3. Set $L=TS$.

This technique can also be applied directly to the share data of non-text components in a secret HTML.

2.1.9 Steganographic and Authentication Techniques for Header Components in E-Mails

A steganographic technique and an authentication technique for header components in E-mails are proposed. By utilizing certain properties of the e-mail standard format, an e-mail header can be hidden in another e-mail header generated in this study with authentication signals to achieve steganography and authentication effects. In the following paragraphs, some properties of e-mail headers are first introduced, and the steganographic technique and authentication technique are then proposed in turn.

A header in an e-mail contains many kinds of information, such as receivers' e-mail address, sender's e-mail address, e-mail subject, launch time, etc. In light of the e-mail standard format, every item of information in a header is expressed as a pair of title and value. Some titles are constant and must be in a header, while some titles can be specified with a leading string "X-" by an e-mail programmer. For example, the title "From" is constant and necessary for identifying the source of an e-mail in an e-mail header and "X-SMART" is a specified title. In addition, in order to transmit e-mail correctly on the internet, the values of some titles, such as "Subject", may be encoded by one of two coding methods, the base-64 and the quoted-printable methods, which are described in e-mail format standard.

The values of constant titles can be seen on e-mail software. For a secret e-mail, the values of the constant titles are important. The proposed procedure to embed these important values into a pseudo header to create steganographic effects is described as an algorithm as follows.

Algorithm 2.7 *Steganographic effect creation for e-mail headers.*

Input: an e-mail header H .

Output: a pseudo e-mail header H' .

Steps.

1. Let the titles “Subject”, “From”, “To”, “Date”, “Cc” and “Bcc” be denoted as $T_1, T_2, T_3, T_4,$ and $T_5,$ respectively.
2. For $i = 1$ to $5,$ extract all the values $V_{i,1}, \dots, V_{i,t_i}$ of T_i in $H,$ where t_i denotes the number of the values of $T_i.$
3. For each value $V_{i,j},$ encode $V_{i,j}$ by base-64 encoding method into $v_{i,j}.$
4. Create a string S of the following form:
$$t_1 | v_{1,1} | \dots | v_{1,t_1} | t_2 | v_{2,1} | \dots | v_{2,t_2} | \dots | t_5 | v_{5,1} | \dots | v_{5,t_5}$$
5. Create an attribute A with title “X-scanInfo” and value $S.$
6. For $i = 1$ to $5,$ randomly select a pseudo value $v'_{i,1}$ of T_i for the corresponding database.
7. Create an empty pseudo e-mail header $H'.$
8. For $i = 1$ to $5,$ add an attribute of T_i into $H'.$
9. Add attribute A with title “X-scanInfo” and value S into $H'.$

The corresponding recovery procedure is in the following.

Algorithm 2.8 *Recovery process of Algorithm 2.7.*

Input: a e-mail header $H'.$

Output: a recovered e-mail header $H_r.$

Steps.

1. Extract the value S of the attribute of title “X-scanInfo.”
2. Separate S into $v_{1,1}, \dots, v_{1,t_1}; v_{2,1}, \dots, v_{2,t_2}; \dots; v_{5,1}, \dots, v_{5,t_5}.$
3. For each $v_{i,j},$ decode $v_{i,j}$ into $V_{i,j}$ by a base-64 decoding method

4. Create H_r .
5. For $i = 1$ to 5, add an attribute with title T_i and values $v_{i,1}, \dots, v_{i,t_i}$.

This steganographic effect creation technique will be applied later to the share data of important parts in an e-mail header without modifications.

In order to embed authentication signals of important parts of an e-mail header into the pseudo header, in which the important values are embedded, one property of e-mail headers is utilized. The property is that a string of tab and space symbols concatenated at the rear of one header string does not influence the interpretation of the attributes of an e-mail header except the attribute of title “Subject”. Therefore, by specifying the number of tab and space symbols in advance, the number of authentication signals can be obtained and the authentication signals can be then generated, divided, and concatenated at the rear of the lines.

Let H be an e-mail header with important information S inside, H_a be the e-mail header after embedding authentication signals of S , a tab symbol denote ‘1’, a space symbol denote ‘0’. And $AUTHEN_GEN(x, l, k)$ denotes the l -bit string of authentication signals generated from a string x with a key k , where l is the number of the bits that will be concatenated at the rear of a string in H . The proposed authentication signal embedding procedure is shown in the following:

Algorithm 2.9 *Authentication signal embedding process.*

Input: H, K .

Output: H_a .

Steps.

1. Extract the value S of the attribute of title “X-scanInfo” from H .
2. Locate all strings L_1, \dots, L_m , which are not used to express the attribute of title “Subject”, in H , where m the number of such lines in H .

3. Compute the authentication signal string $AS = AUTHEN_GEN(S, l \times m, K)$.
4. Divide AS into m l -bit strings AS_1, AS_2, \dots, AS_m .
5. Set $H_a = H$.
6. For $i = 1$ to m , do the following two steps.
 - (i) Translate AS_i into a string TAS_i made up of space and tab symbols.
 - (ii) Append TAS_i to the rear of L_i to form $TASL_i$.
 - (iii) Replace L_i in H_a with $TASL_i$.

Let H be an e-mail header and K be a key for authentication. The corresponding authenticating procedure is as follows.

Algorithm 2.10 *Authenticating process.*

Input: H, K .

Output: a Boolean value b , which denotes the authentication result of H .

Steps.

1. Extract the value S of the attribute of title "X-scanInfo" from H .
2. Locate all strings L_1, \dots, L_n , which are not used to express the attribute of title "Subject", in H , where n the number of such lines in H .
3. Set $CAS = AUTHEN_GEN(S, l \times n, K)$.
4. Set the extracted authentication signal string EAS to be empty.
5. For $i = 1$ through n , do the following two steps.
 - (i) Extract all tab and space symbols at the rear of L_i to form a string $TEAS_i$.
 - (ii) Translate $TEAS_i$ back into binary string EAS_i .
 - (iii) Append EAS_i to the rear of EAS .
6. If EAS equals CAS , set b to be *TRUE*; else *FALSE*.

The adopted procedure, the procedure of authentication signal generation, will be

described in the following section.

2.1.10 Authentication Technique for Verifying Share Data

An authentication technique for verifying share data is proposed. A way to generate authentication signals is first proposed. And an authentication procedure for verifying share data is then described.

Let S be a piece of share data, H be the size of space for hiding authentication signals, K be the corresponding key, $R(s, l)$ be the random number of size l , where s is a seed, $L(x)$ denote the size of data x in bit, and $y(i)$ denote the i th bit of y where y is a string of data. Authentication signals of S can be generated by the following procedure.

Algorithm 2.11 *Authentication signal generation.*

Input: S , H , and K .

Output: a string of authentication signals AS .

Steps.

1. Create AS of size H .
2. Set AS to $R(K, H)$.
3. For $i = 1$ to $L(S)$, set

$$AS([(i - 1) \pmod H] + 1) = AS([i \pmod H] + 1) \text{ XOR } S(i),$$

where XOR denotes the exclusive-OR operator.

Because of the use of exclusive-OR operations, authentication signals can be generated quickly by the above procedure.

The authentication capability creation procedure is shown as Figure 2.5 below.

The authentication signals of the share data are generated by the procedure of authentication signal generation mentioned above and then embedded into the host to generate the result.

The authenticating procedure is as shown in Figure 2.6. To authenticate a text, share data and authentication signals are extracted from the text in the beginning. Then, other authentication signals are generated by applying the procedure of authentication signal generation to the extracted share data. By comparing the generated authentication signals with the extracted ones, the fidelity of the text can be authenticated.

In these two procedures, the processes of how to embed and extract data is not mentioned. This pair of corresponding procedures is regarded as a template authentication technique. Various kinds of data hiding techniques can be imported as the two processes for the uses of the template authentication technique.

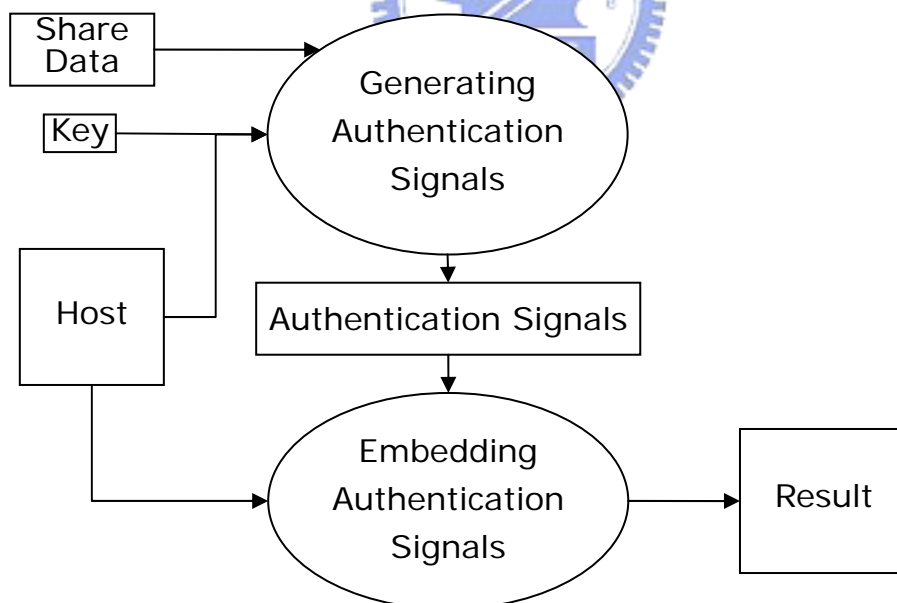


Figure 2.5 Flowchart of creating authentication capability.

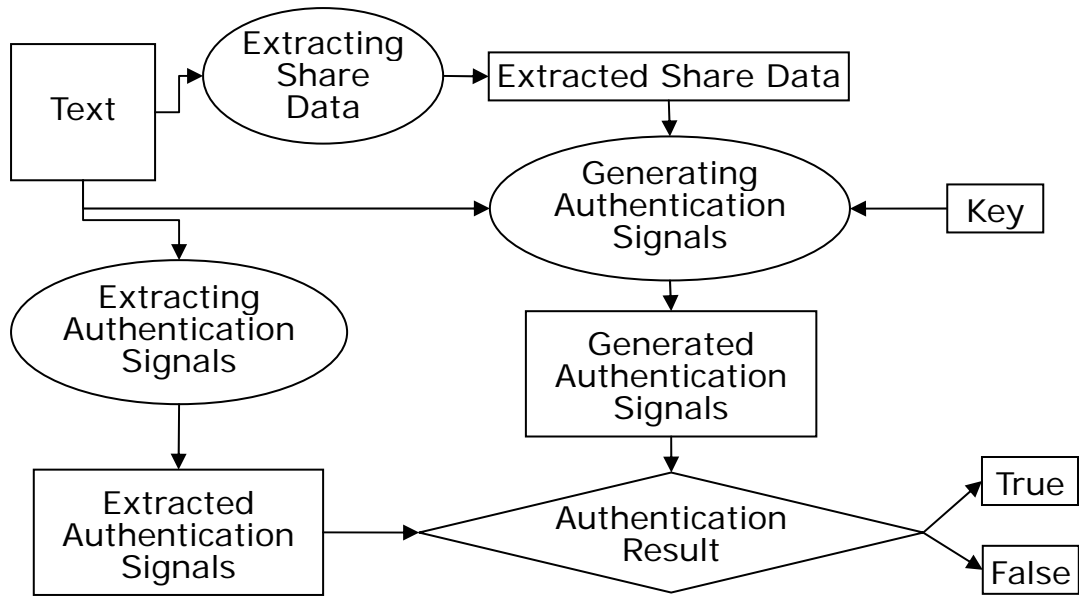


Figure 2.6 Flowchart of authenticating process.



Chapter 3

Secret Sharing Method with Steganographic Effects and Authentication Capability for Pure Text Documents

Introduction

In this chapter, a secret sharing method for pure text documents with steganographic effects and authentication capability is proposed. The remainder of this section will introduce some properties of plain text documents. In the following sections, an overview of the method is first introduced in Section 3.2. Next, the detail of the method is described in Section 3.3. And then, in Section 3.4, some experimental results are shown. Finally, some discussions and a summary of this chapter are made in Section 3.5.

For pure text documents in English, the hexadecimal values of meaningful ASCII symbols range from 20 to 7E. However, for pure text documents in character languages, such as Chinese, Japanese, Korean, etc, possible hexadecimal values of bytes in a pure text document of a character language are from 0x00 to 0xFF. Therefore, in general, the hexadecimal value range of a symbol in pure text documents is from 0x00 to 0xFF, which is also the range that should be handled for secret sharing.

In addition, because every symbol in a pure text document is visible, one can identify every symbol by inspection. It is also the reason why the capacity of the

redundancy information of pure text documents for hiding data is quite less, compared with images, or videos.

Overview of Proposed Method

In this section, the focus is on the processes of sharing and recovery. In Section 3.2.1, a secret pure text sharing process is shown and, in Section 3.2.2, the corresponding secret recovery process is presented.

3.1.1 Secret Pure Text Sharing Process

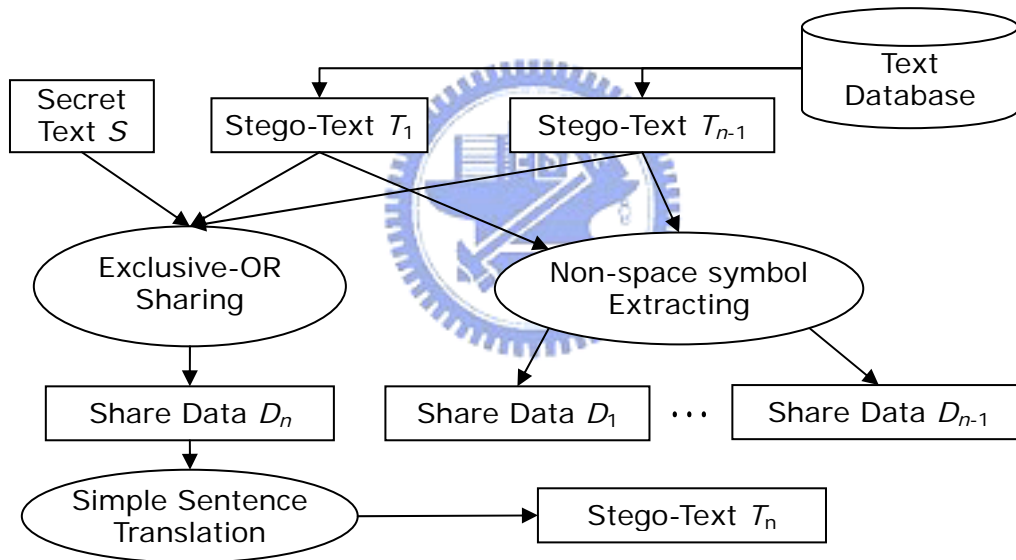


Figure 3.1 Flowchart of secret pure text sharing process.

Figure 3.1 shows the process of sharing the secret text S among participants and creating steganographic effects on the last piece of share data D_n . Suppose that the number of participants is n and each participant P_i holds a key K_i . First, $n - 1$ stego-texts are selected from the text database. Second, S is exclusive-ORed with the $n - 1$ stego-texts to generate share data D_n , which is a sequence of meaningless symbols. And share data D_1, D_2, \dots , and D_{n-1} are the results from extracting non-space symbols from T_1, T_2, \dots , and T_{n-1} , respectively. Third, D_n is translated into

simple sentences to form a stego-text, which is meaningful. T_n and $T_1, T_2, \dots,$ and T_{n-1} are themselves the $n - 1$ stego-texts.

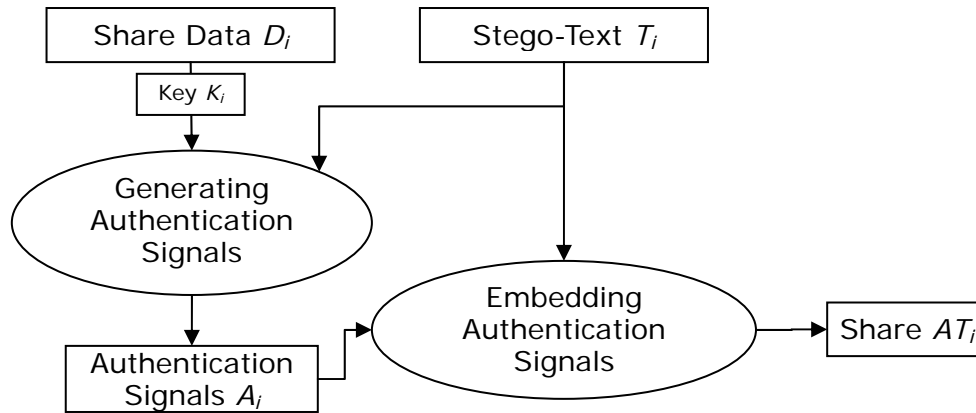


Figure 3.2 Flowchart of process of authentication signal embedding.

Finally, as shown in Figure 3.2, the authentication signals of share data D_i is generated with the corresponding key K_i and then embedded into stego-text T_i , for $i = 1$ through n to form share AT_i . Now, n shares with authentication capability for the n participants each are created.



3.1.2 Secret Pure Text Recovery Process

Two stages are performed in the secret pure text recovery process. The first stage is the authentication process for verifying the integrity of given shares. The second one is the recovery process for recovering the secret pure text from the given shares.

In Figure 3.3, the authentication process for shares is shown. Suppose n shares with n corresponding keys are used for recovering the secret pure text. Because of the difference between share data extraction processes, an authentication process is utilized for the last share and another authentication process is exploited for the other shares. If all of the n shares are authenticated successfully, the second stage begins. If not, an authentication failure report about the texts is issued.

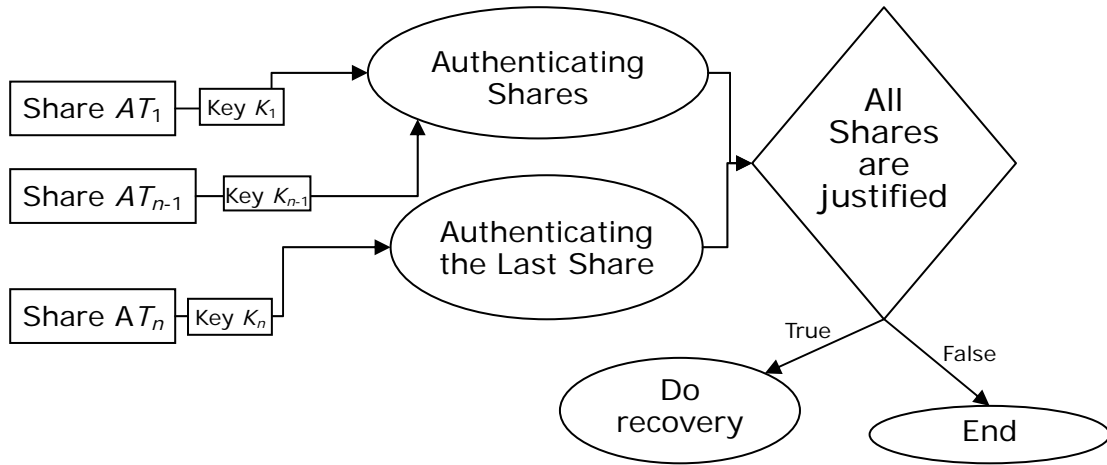


Figure 3.3 Flowchart of authenticating process.

If all share texts are authenticated and legal, the recovery process as shown in Figure 3.4 is performed. The first step of the process is to extract share data from shares. For the last share AT_n , inverse simple sentence translation is applied to obtain share data D_n . For other shares, authentication signals are simply removed from the shares to get share data D_i , for $i = 1$ to $n - 1$. Finally, a recovered secret pure text is acquired by exclusive-ORing all pieces of share data.

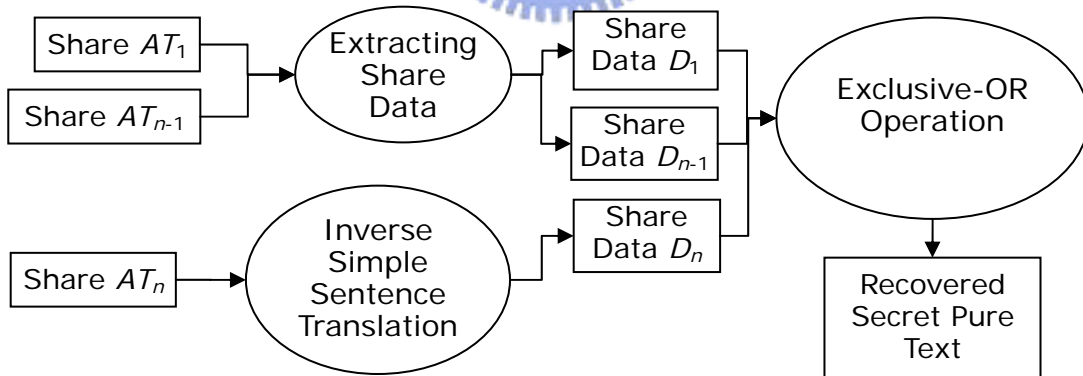


Figure 3.4 Flowchart of process of recovering secret pure text.

Proposed Detailed Processes for Sharing Secret Pure Texts

In this section, three detailed processes for sharing secret pure texts are described.

In Section 3.3.1, the application of exclusive-OR operations to secret pure text sharing is proposed. In Section 3.3.2, the steganographic technique for the last piece of share data is described. Finally, the technique of embedding authentication signals into shares is described.

3.1.3 Use of Exclusive-OR Operations for Sharing

Because of the variance of the length of texts, the technique proposed in Section 2.2.1 cannot be simply applied to texts. However, this problem can be solved by choosing texts from a text database carefully and some modifications.

Let S be the secret pure text, n be the number of participants, $L(x)$ be the length of text x in byte, $D(x)$ be the length of text x , excluding the space symbols in x , $S(x)$ be the string of text x with the absence of space symbols, TDB be the text database, $x(j)$ denote the j th byte of text x , $R()$ denote the random number generator, and $TDB(j)$ denote the j th text in TDB . The procedure of secret pure text sharing by exclusive-OR operations is as follows.

Algorithm 3.1 *Secret pure text sharing.*

Input: S and TDB .

Output: n share data S_j , for $j = 1$ to n

Steps.

1. For $j = 1$ to n , create empty text T_j .
2. For $j = 1$ to $n - 1$, do the following steps.
 - (i) Set $T_j = TDB(R())$.
 - (ii) If $(D(T_j) < L(S))$ go to (i); else do the following steps.
3. Set $T_n = S$.

4. For $k = 1$ to $n - 1$, set $S_k = S(T_k)$.
5. For $j = 1$ to $L(T_n)$, do the following step.
 - (i) For $k = 1$ to $n - 1$, set $S_n(j) = S_n(j) \text{ XOR } S_k(j)$.

By choosing texts in which the number of non-space symbols is larger than the number of symbols in the secret pure text, the technique of secret sharing by exclusive-OR operations for secret pure texts can work well.

While recovering, n given pieces of share data with an index of the last piece of share data are used as input to the following procedure.

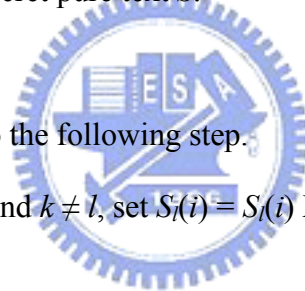
Algorithm 3.2 *Secret pure text recovery.*

Input: S_1, \dots, S_n , and the index l of the last piece of share data.

Output: the recovered secret pure text S .

Steps.

1. For $i = 1$ to $L(S_l)$, do the following step.
 - (i) For $k = 1$ to n and $k \neq l$, set $S_l(i) = S_l(i) \text{ XOR } S_k(i)$.
2. Set $S = S_l$.



The secret sharing technique is quite simple, but oppositely, it is practical for its efficiency.

3.1.4 Steganographic Technique for Last Piece of Share Data

Although share data in the texts retrieved from the text database can be meaningful by directly regarding the retrieved texts as stego-texts, the last piece of share data generated by a sequence of exclusive-OR operations is meaningless. In order to make all share data meaningful, manipulating the last share data into a

meaningful text is the main issue in this section.

Two candidate techniques proposed in Section 2.2.4 and 2.3.2 can be applied to make the last share data meaningful. The result by the technique of data hiding for text documents, proposed in Section 2.2.4, is more meaningful than that by the technique of translating share data into simple sentences. However, under practical consideration of avoiding participants from changing share data intentionally or accidentally, the later technique was chosen for making the last share data meaningful and the former technique for authentication.

Therefore, the last piece of share data is encoded into several simple sentences by the technique of translating share data into simple sentences without any modification as the last stego-text. And now all stego-texts are meaningful texts with share data embedded in.



3.1.5 Steganographic Technique for Authentication of Text Shares

In order to avoid participants from modifying stego-texts intentionally or accidentally, authentication signals of share data are embedded in given texts. Two techniques, proposed in Sections 2.2.4 and 2.3.6 are integrated to achieve this goal.

By importing the data hiding technique in Section 2.2.4 to the pattern authentication technique in Section 2.3.6, the authentication embedding process is shown in Figure 3.5 below. Share data, the corresponding stego-text and the key are used as input to the authentication generation technique mentioned in Section 2.3.6 to generate authentication signals. Then, the authentication signals are embedded into the stego-texts by the data hiding technique described in Section 2.2.4. After the

process, the resulting text is called a share.

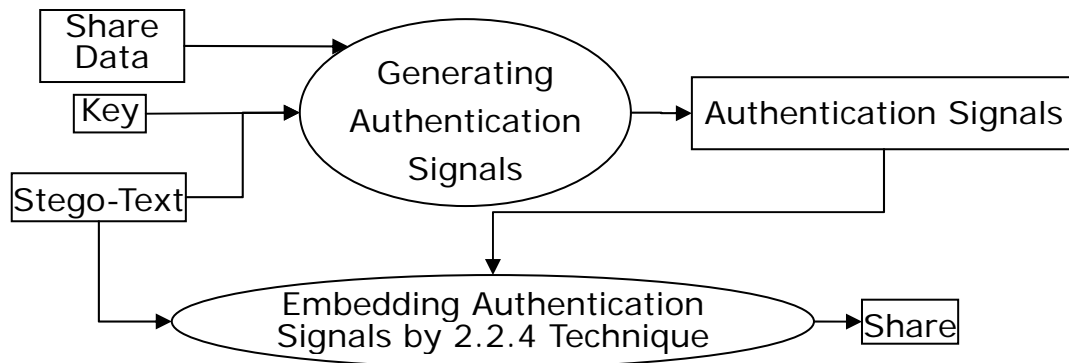


Figure 3.5 Flowchart of process of embedding authentication signals into a stego-text.

The authenticating process after importing the technique in Section 2.2.4 is illustrated in Figure 3.6. While extracting share data, the technique used for the last share is different from the technique utilized for the other shares. The former is the inverse simple sentence translation technique and the later is the technique of extracting non-space symbols from an input text. After getting share data, every share can do authentication in the way shown above.

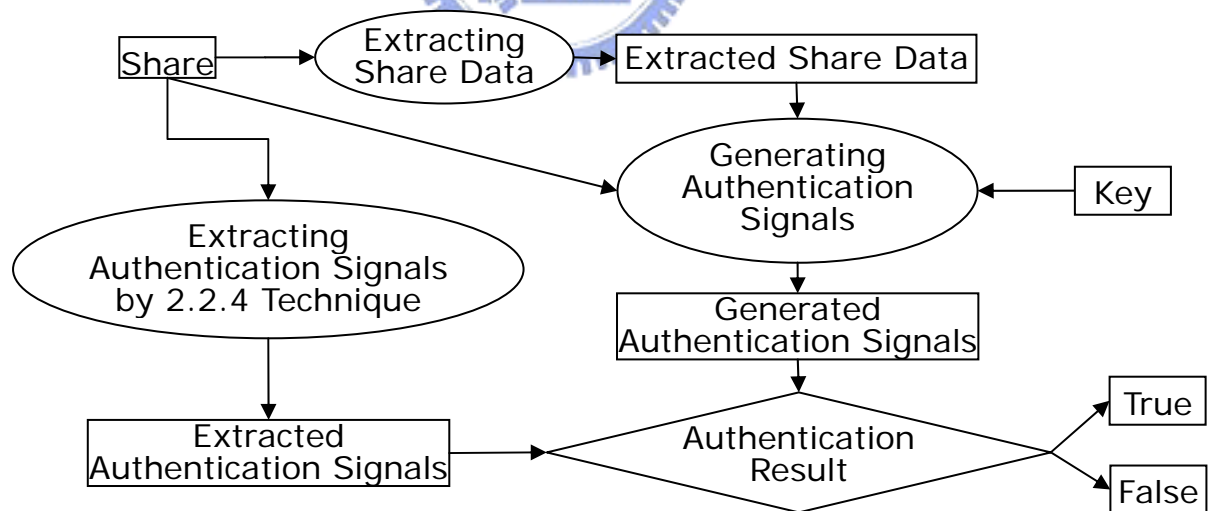


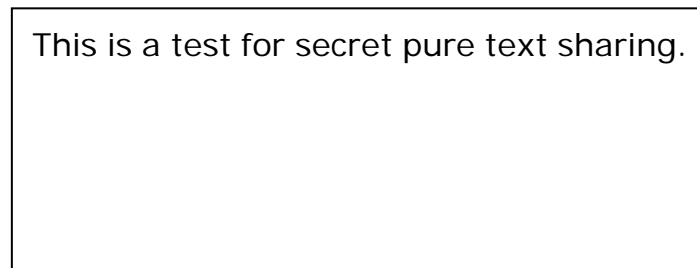
Figure 3.6 Flowchart of verification process of share text.

In order to applying the integrated method successfully, the texts chosen in the sharing stage are further limited to articles. Therefore, texts chosen from the text database should be articles in which the number of non-space symbols must be larger

than that in a secret text.

Experimental Results

An example of sharing a pure text secret among three participants is illustrated in the following. Figure 3.7 shows a secret text. The content of a secret pure text can contain any word, character, or symbol.



This is a test for secret pure text sharing.

Figure 3.7 The secret pure text.

Before sharing the secret pure text, two articles are selected from a database and shown in Figure 3.8 below. Obviously, the number of non-space symbols in the two articles are each more than that of symbols in the secret text in Figure 3.7. For the last share, after sharing the secret text and creating steganographic effect on the last piece of share data, the generated last stego-text is shown in Figure 3.9.

After embedding the corresponding authentication signals into each stego-text by the use of inter-word spacing and inter-sentence spacing, two of the resulting shares are shown in Figure 3.10 below. If all of shares are authenticated successful, the secret pure text can be recovered correctly as illustrated in Figure 3.11. Otherwise, failure to pass the authentication process is reported.

Discussions and Summary

In this chapter, we have described an efficient solution for sharing secret pure texts. The proposed method catches the spirit of secret sharing-to encode secret into parts such that each part contains partial information of the secret, even though the

sharing operation used is the exclusive-OR operation, which is simple.

A method for sharing secret pure text with steganographic effect and authentication capability is proposed by integrating three techniques proposed in Chapter 2. With some modifications, the secret sharing technique in Section 2.2.1 is applied to pure texts; the steganographic effect creation technique in Section 2.3.2 is applied directly to the last piece of share data to form the last stego-text; the technique by integrating the two techniques in Section 2.2.4 and Section 2.3.6 are used for authentication of each stego-text to form a share.

The first bombings occurred outside three police stations in central Basra during Wednesday's morning rush hour.

Many of the dead and injured were children traveling in passing buses on their way to school.

(a)

It is designed purely for research. Professor Wilmut has stressed that his team has no intention of producing cloned babies, and said the embryos would be destroyed after experimentation.

(b)

Figure 3.8 Stego-texts (a) through (b) articles selected from an article database.

Benedict examines Buck. Burton doubts Chaney. Carol kidnaps Bryce. Alton tricks Bryant. Caleb argues with Auburn. Carlyle aims at Bryon. Amyot misses Buck. Bryant suspects Carroll. Chaney looks at Casey. Buck interrogates Arnold. Aldred suspects Bruno. Calvin examines Aubert. Carlyle evades Chester. Bentley snubs Byrne. Cassidy hugs Laura.

Figure 3.9 The last stego-text.

Benedict examines Buck. Burton doubts Chaney. Carol kidnaps Bryce. Alton tricks Bryant. Caleb argues with Auburn. Carlyle aims at Bryon. Amyot misses Buck. Bryant suspects Carroll. Chaney looks at Casey. Buck interrogates Arnold. Aldred suspects Bruno. Calvin examines Aubert. Carlyle evades Chester. Bentley snubs

(a)

The first bombings occurred outside three police stations in central Basra during Wednesday's morning rush hour.

Many of the dead and injured were children traveling in passing buses on their way to school.

(b)

Figure 3.10 Shares (a) the last share; (b) one of the other shares.

This is a test for secret pure text sharing.

Figure 3.11 Recovered secret pure text.

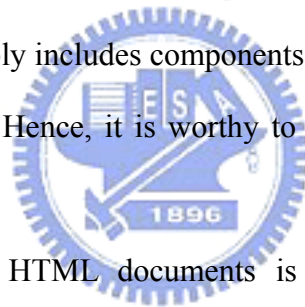


Chapter 4

Secret Sharing Method with Steganographic Effects for HTML Documents

Introduction

With the development of digital multimedia techniques, a digital document can contain components of various types. One of these kinds of documents is of HTML, which is an open format and integrates many types of components into one document. A secret document also possibly includes components of different types and, therefore, can be an HTML document. Hence, it is worthy to apply secret sharing on HTML documents.



A strategy for sharing HTML documents is to extract important parts of components in a secret HTML document and conduct secret sharing. As for the strategy for steganographic effect creation for the share data, one way is to transform share data into an HTML document of the same appearance of the secret HTML document. According to these strategies, in the remainder of this section, certain properties of HTML documents and the proposed method for sharing secret HTML documents are investigated in turn. Then, in Section 4.2, detailed processes for sharing secret HTML documents are proposed. In Section 4.3, some experimental results are illustrated. Finally, in Section 4.4, discussions and a summary are given.

4.1.1 Properties of HTML Documents

An HTML document contains tags which are utilized as descriptions of the components in the HTML document, such as fonts of text components, sources of image components, layouts of all components, etc. In the following, the behavior of in-tag texts in an HTML document on browsers is described first, and then the behavior of outside-tag texts in an HTML document is stated.

4.1.1.1. In-tag Text

A tag in HTML denotes certain elements. An HTML tag contains a left angle bracket (<), a tag name, and a right angle bracket (>), e.g., <HTML>. Tags usually appear in pairs to start and end the tag instruction. Some tags include attributes which are additional information inside the start tag. An attribute in a tag is a name-value pair with an equal mark (=) between the name and the value. The end tag is of the form of a left angle bracket (<) followed by a slash (/), a tag name like that in the start tag, and a right angle bracket (>). Texts in tags are not displayed on browsers.

In a tag, the tag name and the attribute names are not case-sensitive. That is, for instance, a tag <BODY> can operate as well as another tag <BoDy>, and so does an attribute. On the other hand, values of attributes may or may not case-sensitive, depending on what attribute it is. In addition, in an HTML document, multimedia components shown on browsers are specified as a source path kept within a tag as the value of an attribute of the tag.

A special tag in the HTML document is the comment tag which is formed with a leading string, “<!--“, followed by the comment words and an end string “-->”. This kind of tag is useful for HTML editors to write readable HTML documents.

4.1.1.2. Outside-tag Text

Texts outside tags in an HTML document are just the components that are

displayed on browsers. A noticeable property of outside-tag text is that a sequence of three symbols, space symbol, new-line symbol, and tab symbol, in an outside-tag text is displayed just as a single space symbol on browsers. As shown in Figure 4.1, Figure 4.1(a) is the source of an HTML and Figure 4.1(b) is the corresponding result on a browser. Although there is a sequence of the three symbols between “Hello” and “world!!!”, only one space symbol shows up on the browser.

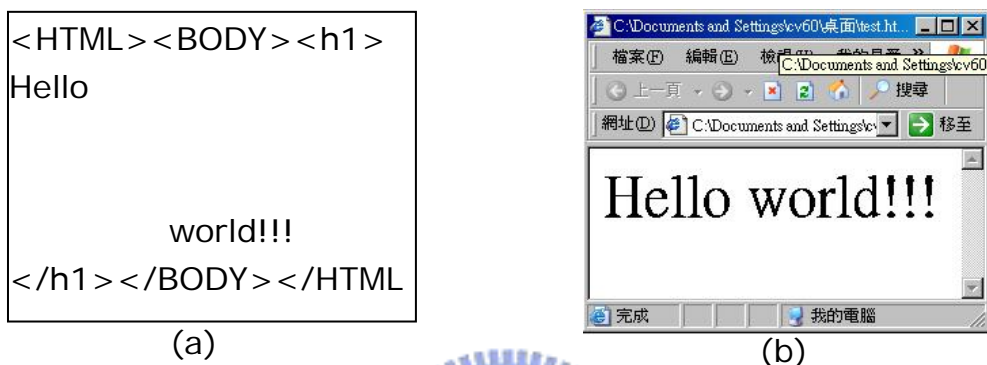


Figure 4.1 An HTML document. (a) the source code; (b) the display on a browser.

4.1.2 Processes of Proposed Method

Let the secret HTML be H , and the number of participants be n . In the following, the secret sharing process and the corresponding recovery process for secret HTML documents will be described in turn.

4.1.2.1. Secret HTML Sharing Process

In order to make the style of stego-HTML documents identical to that of the corresponding secret HTML document, the layout of the secret HTML document is reserved and the components of a secret HTML is replaced with other components of the same type. According to this strategy adopted in this study, the sharing process is stated in the following paragraphs.

In Figure 4.2, the text component T of secret HTML H and the non-text components C_1, C_2, \dots, C_v of H are first extracted from H , where v denotes the number of non-text components in H . Next, for each component, cooperative sharing is applied to the important part of the component to generate n pieces of share data, where n is the number of participants of H . Note that the location of the important part of a component is dependent on the component type.

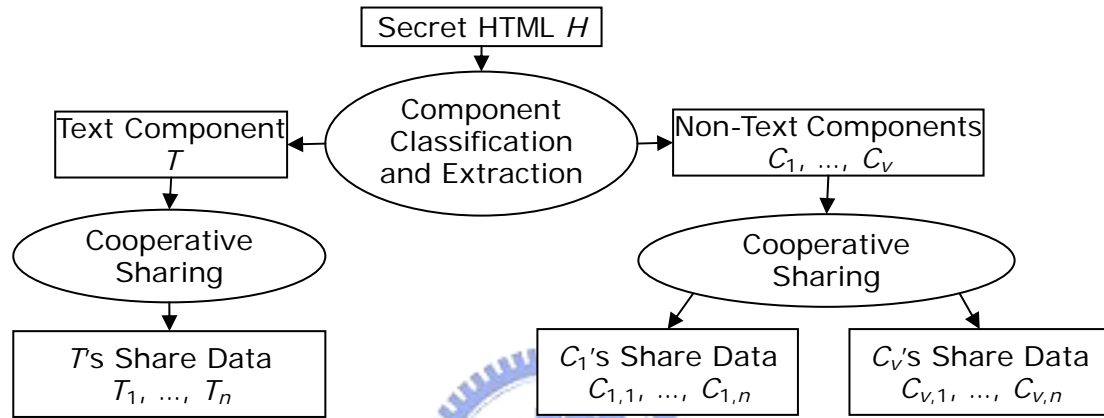


Figure 4.2 Flowchart of component extraction and sharing.

Then, as shown in Figure 4.3, share data T_j , which are one piece of the sharing result of T and will be assigned to the j th participant later, are embedded into article G_j , which is selected from an article database, to form stego-text component A_j . Besides, for share data C_{ij} of the i th non-text component of the j th participant, a fake component is created with C_{ij} embedded in to form stego-non-text component $L_{i,j}$.

After steganographic effect creation for the share data of components, the j th participant gets a stego-HTML document H_j generated by combining all the components according to the component layout of the secret HTML H , as shown in Figure 4.4. By this way, each participant obtains his/her own stego-HTML document.

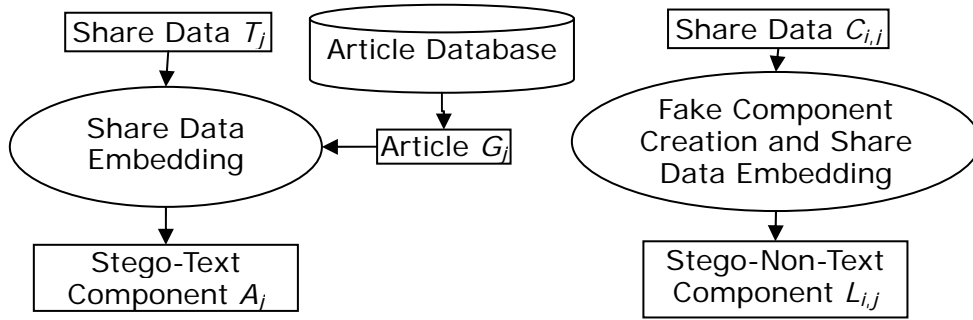


Figure 4.3 Flowchart of the steganography processes for the share data of components.

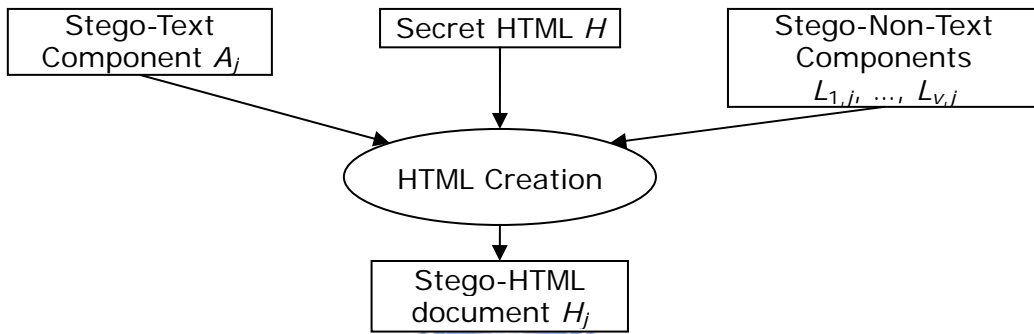


Figure 4.4 Flowchart of the process of stego-HTML document creation.

4.1.2.2. Secret HTML Recovery process

In the recovery process, n stego-HTML documents are used as input and the output is a recovered secret HTML document.

In Figure 4.5, for j th stego-HTML document H_j , stego-text component A_j and stego-non-text components, $L_{1,j}, \dots, L_{v,j}$, are first classified and extracted. Then share data extractions are performed to get share data T_j from A_j and share data $C_{1,j}, \dots, C_{v,j}$ from $L_{1,j}, \dots, L_{v,j}$, respectively.

Figure 4.6 shows the secret component recovery processes. The recovered secret text component can be obtained by pooling and applying cooperation sharing recovery operations to all the pieces of share data, T_1, \dots, T_n . The same procedure can be applied to the share data, $C_{i,1}, \dots, C_{i,n}$, to get the recovered non-text component RC_i , the i th non-text component of the secret HTML document.

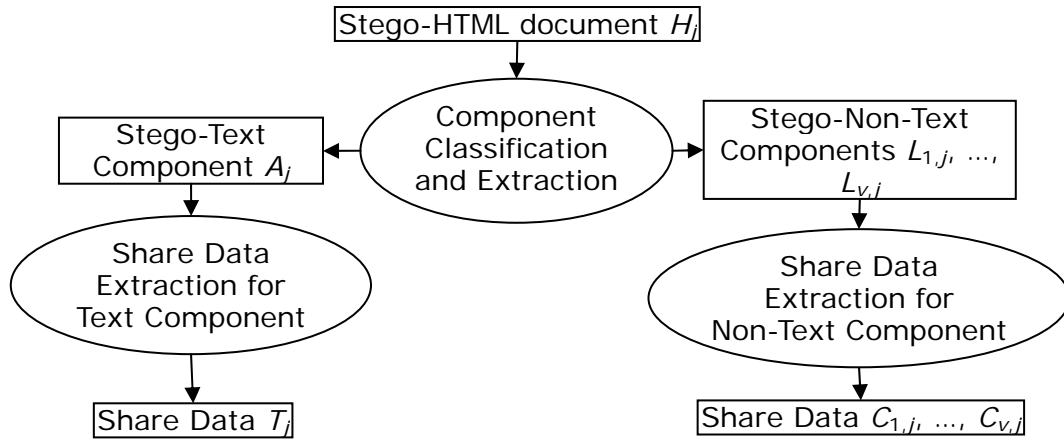


Figure 4.5 Flowchart of the process of share data extraction of components.

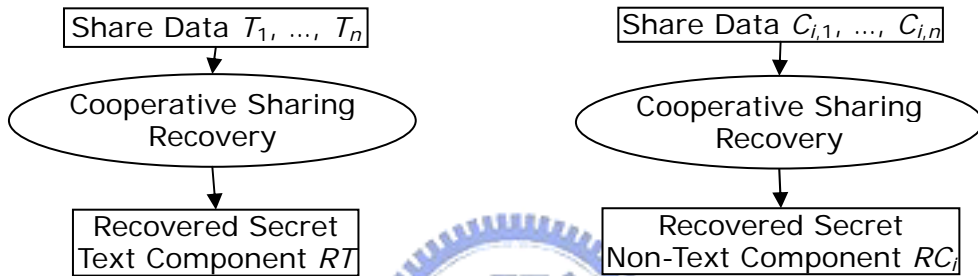


Figure 4.6 Flowchart of secret component recovery processes.

Due to the identicalness of the style of the secret HTML document and that of all stego-HTML documents, by referring to the layout of the first stego-HTML document H_1 , the secret HTML can be recovered by replacing the recovered components with the corresponding ones in H_1 , as shown in Figure 4.7.

Proposed Detailed Processes for Sharing Secret HTML Documents

After an overview of the sharing process and the recovery process described previously, the details in these two processes are described. In Section 4.2.1, the detailed process for sharing the text component of a secret HTML document and creating steganographic effects on the share data of the text component are proposed.

In Section 4.2.2, the detailed process for sharing non-text components in secret HTML documents and applying steganographic techniques to the share data of the components are described.

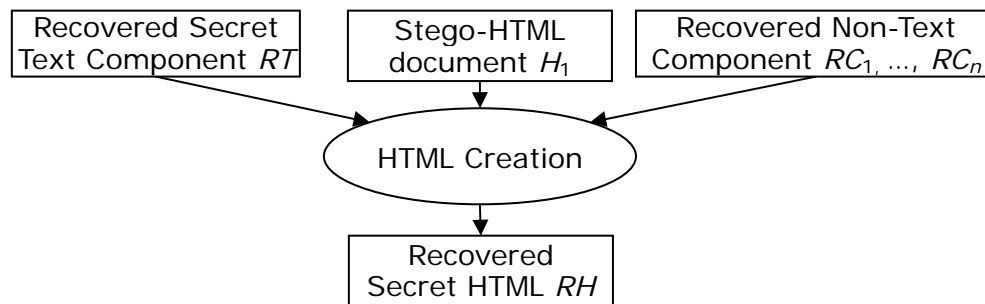


Figure 4.7 Flowchart of secret HTML recovery process.

4.1.3 Process for Sharing the Text Component in

Secret HTML

The text component of a secret HTML is the text outside the tags of the secret HTML. Because the segments of the text component distribute over the secret HTML, cooperative sharing operation with data magnitude control, proposed in Section 2.3.1, may be applied on the text component segment by segment. In order to achieve steganographic effects on the share data of the component, a steganographic technique described in Section 2.3.3 is adopted.

Let H be the secret HTML, n be the number of participants of H , G_1, \dots, G_n be the selected English articles, $L(x)$ be the length of string x , and $N(x)$ be the number of inter-word, interline and inter-sentence spaces in string x . The procedure combining cooperative sharing operations and the technique described in Section 2.3.3 is as follows.

Algorithm 4.1 *Secret sharing for text components with steganographic effects.*

Input: H, G_1, \dots, G_n .

Output: n stego-HTML documents H_1, \dots, H_n .

Steps.

1. Parse and extract m text component segments S_i between two successive tags in H , where m is the total number of segments of the text component in H .
2. For each text component segment S_i , do following steps.
 - (i) For $j = 1$ to n , cut several sentences in front of the remainder of G_j as a string $C_{i,j}$ such that the distance between $L(C_{i,j})$ and $L(S_i)$ is minimal.
 - (ii) Encode S_i into n share data $SD_{i,1}, \dots, SD_{i,n}$ by cooperative sharing operations with data magnitude control.
 - (iii) For $j = 1$ to n , translate $SD_{i,j}$ into a string $E_{i,j}$ made up of tab symbols, new-line symbols, and space symbols.
 - (iv) For $j = 1$ to n , embed a leading space symbol and $\lceil L(E_{i,j}) / N(C_{i,j}) \rceil$ symbols of $E_{i,j}$ into the inter-word, inter-line, and inter-sentence spaces in $C_{i,j}$ to generate $D_{i,j}$.
3. For $j = 1$ to n , do the following steps.
 - (i) Set $H_j = H$.
 - (ii) For $i = 1$ to m , replace S_i in H_j with $D_{i,j}$.

After the process, a stego-text component is generated.

The corresponding recovery procedure for the secret text component is as follows.

Algorithm 4.2 *Recovery of secret text components.*

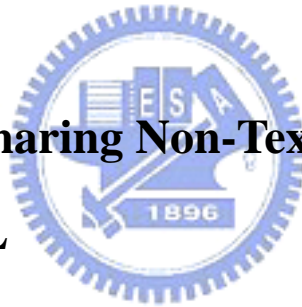
Input: H_1, \dots, H_n .

Output: recovered secret HTML RH .

Steps.

1. For $j = 1$ to n , do the following steps.
 - (i) Parse and extract m text component segments $S_{1,j}, \dots, S_{m,j}$ between two successive tags in H_j .
 - (ii) Extract translated strings $E_{1,j}, \dots, E_{m,j}$ from $S_{1,j}, \dots, S_{m,j}$, respectively
 - (iii) Translate $E_{1,j}, \dots, E_{m,j}$ back into share data $SD_{1,j}, \dots, SD_{m,j}$.
2. For $i = 1$ to m , recover the secret text component segment S_i from $SD_{i,1}, \dots, SD_{i,n}$ by cooperative sharing recovery operations with data magnitude control.
3. Set $RH = H_1$.
4. For $i = 1$ to m , replace $S_{i,1}$ with S_i .

4.1.4 Process for Sharing Non-Text Components in Secret HTML



Non-text components include multimedia files, such as images, audios, videos, Flash files, etc, and hyperlinks. Such components are represented by source paths in the HTML document. The technique proposed in Section 2.3.1 is used for sharing non-text components and the technique described in Section 2.3.4 is adopted for creating steganographic effects on the non-text component shares, instead of the component itself. In addition, a hyperlink contains not only a source path inside tags but also a section of texts between the start and the end hyperlink tags as a medium on browsers for clicking and accessing the hyperlink source. Hyperlinks in the HTML document are treated as a special case for sharing. During the process of steganographic effect creation, the procedure for the special case will be executed

while the currently encountered component is a hyperlink.

Let H be the secret HTML, n be the number of participants of H , DIL be the dynamic image link address, DAL be the dynamic audio link address, DVL be the dynamic video link address, DFL be the dynamic Flash link address, and DHL be the dynamic hyperlink address. The procedure combining 2.3.1's and 2.3.4's techniques is as follows.

Algorithm 4.3 *Secret sharing of non-text components with steganographic effects.*

Input: $H, DIL, DAL, DVL, DFL, DHL$.

Output: n stego-HTML documents H_1, \dots, H_n .

Steps.

1. Parse and extract v non-text components C_1, \dots, C_v inside tags of H , where v is the total number of non-text components.
2. For each C_i , do the following steps.
 - (i) Encode C_i into n share data $C_{i,1}, \dots, C_{i,n}$ by cooperative sharing operations with data magnitude control.
 - (ii) For $j = 1$ to n , do the steps as follows.
 - (a) Translate $C_{i,j}$ into a string $E_{i,j}$ in hexadecimal.
 - (b) Set $L_{i,j} = DL?E_{i,j}$, where DL is one of the dynamic links DIL, DAL, DVL, DFL and DHL according to the component type of C_i , and symbol “?” is the special notation for separating the dynamic link address DL and its information $E_{i,j}$.
 - (iii) If C_i is a hyperlink, apply the special case procedure described later to the corresponding hyperlink text string HL_i to generate fake text strings $FL_{i,1}, \dots, FL_{i,n}$.
3. For $j = 1$ to n , do the following steps.

- (i) Set $H_j = H$.
- (ii) For each i , do the following steps.
 - (a) Replace C_i with $L_{i,j}$.
 - (b) If C_i is a hyperlink, replace the corresponding hyperlink text string HL_i of C_i in H_j with $FL_{i,j}$.

Here, the corresponding hyperlink text string means the outside-tag text string between a start hyperlink tag and a corresponding end hyperlink tag. For example, “hyperlink_text” is a workable hyperlink in the HTML document, where “” is the start hyperlink tag containing the source path of the hyperlink “http://tw.yahoo.com”, “” is the end hyperlink tag and the hyperlink text string is “hyperlink_text”.

Let HL be the hyperlink text string, $HLDB$ be the database of hyperlink source paths, T be a threshold, $LEN(x)$ be the length of a text string x , and $ABS(l)$ be the absolute value of value l . The special case procedure is described in the following.

Algorithm 4.4 *Secret sharing of non-text components of hyperlink-type with steganographic effects.*

Input: $HL, HLDB$.

Output: n text strings FL_1, \dots, FL_n .

Steps.

1. Encode HL into SHL_1, \dots, SHL_n by cooperative sharing operations with data magnitude control.
2. For $i = 1$ to n , do the following steps.
 - (i) Translate SHL_i into a string EHL_i in hexadecimal form byte by byte.
 - (ii) Get a text string TS from $HLDB$ such that $ABS(LEN(TS) - LEN(HL)) < T$.

(iii) Set $FL_i = \langle!--EHL_i--\rangle TS$.

The corresponding recovery procedure for secret non-text components is as follows.

Algorithm 4.5 *Recovery of secret non-text components.*

Input: H_1, \dots, H_n .

Output: recovered secret HTML RH .

Steps.

1. For $j = 1$ to n , do the following steps.
 - (i) Parse and extract v non-text components $FL_{1,j}, \dots, FL_{v,j}$ from H_j .
 - (ii) Extract strings $E_{1,j}, \dots, E_{m,j}$ from $FL_{1,j}, \dots, FL_{m,j}$, respectively.
 - (iii) Translate $E_{1,j}, \dots, E_{m,j}$ back into share data $C_{1,j}, \dots, C_{m,j}$.
2. For $i = 1$ to m , recover secret non-text component C_i from $C_{i,1}, \dots, C_{i,n}$ by cooperative sharing recovery operations with data magnitude control.
3. Set $RH = H_1$.
4. For $i = 1$ to m , replace $S_{i,1}$ in RH with C_i .

By combining the techniques in Sections 4.2.1 and 4.2.2, complete stego-HTML documents can be generated. That is, each stego-HTML document is formed by replacing the text component and the non-text components of a secret HTML document with a stego-text component and some stego-non-text components.

Experimental Results

Suppose the number of secret sharing participants is two. Figure 4.8 shows a secret HTML which contains two text component segments, “This is a secret sharing test for HTML.” at the top and at the bottom, a hyperlink with hyperlink text string “”, an image at the middle left, a video component at the middle center, and a Flash file at

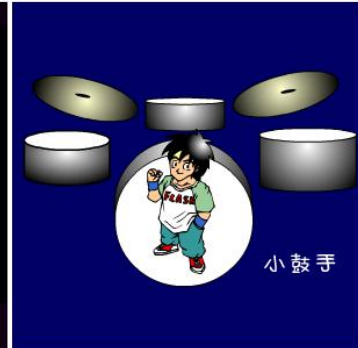
the middle right. The two pictures in Figure 4.9 are the resulting stego-HTML documents. The layout of each stego-HTML document is the same as that of the secret HTML document, while the components in each stego-HTML document are different from those in the secret HTML document. The HTML document in Figure 4.10 is the recovered secret HTML which is the same as the original secret HTML on browsers. Figure 4.11 (a) shows an image source path, the secret image component of the secret HTML, and two corresponding stego-image components are shown in Figure 4.11 (b) and (c). In the image component, the string before the question mark is the address of the image agent and the under line string following the question mark is one of the translated share data of the secret image component.

Discussions and Summary

Because the important parts of a secret HTML are the components that can be displayed or be accessed on browsers, these components are shared among participants of the secret HTML by cooperative sharing operations with data magnitude control. In order to create steganographic effects on the share data of the components, two steganographic techniques for the text component and the non-text components of the secret HTML document are proposed. For text component share data, the technique substitutes the original text component by an article with share data hidden into inter-word, inter-sentence and inter-line spaces. For non-text component share data, the technique uses a dynamic link with share data as the parameter of the link to create steganographic effects. After applying the two steganographic techniques to the share data of the components in the secret HTML document, the share data becomes stego-HTML documents of the style identical to the secret HTML document.

This is a secret sharing test for HTML.

This is a secret sharing link.

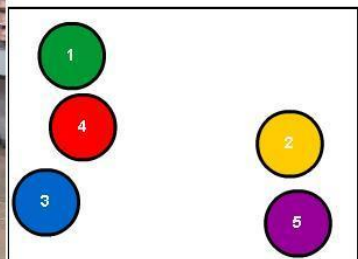


This is a secret sharing test for HTML.

Figure 4.8 A secret HTML document.

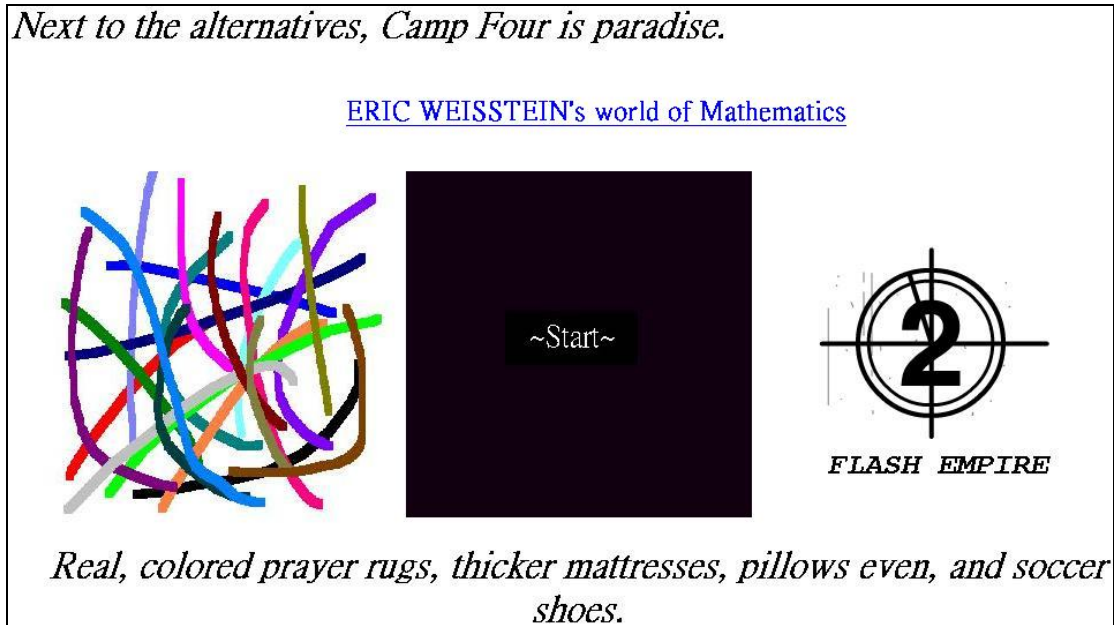
There are so many accidents in the world.

Introduction to Email



They are the reason Dr. David Ho has come to China.

(a)



(b)

Figure 4.9 stego-HTML documents (a) through (b) stego-HTML documents generated from the secret HTML document in Figure 4.8.

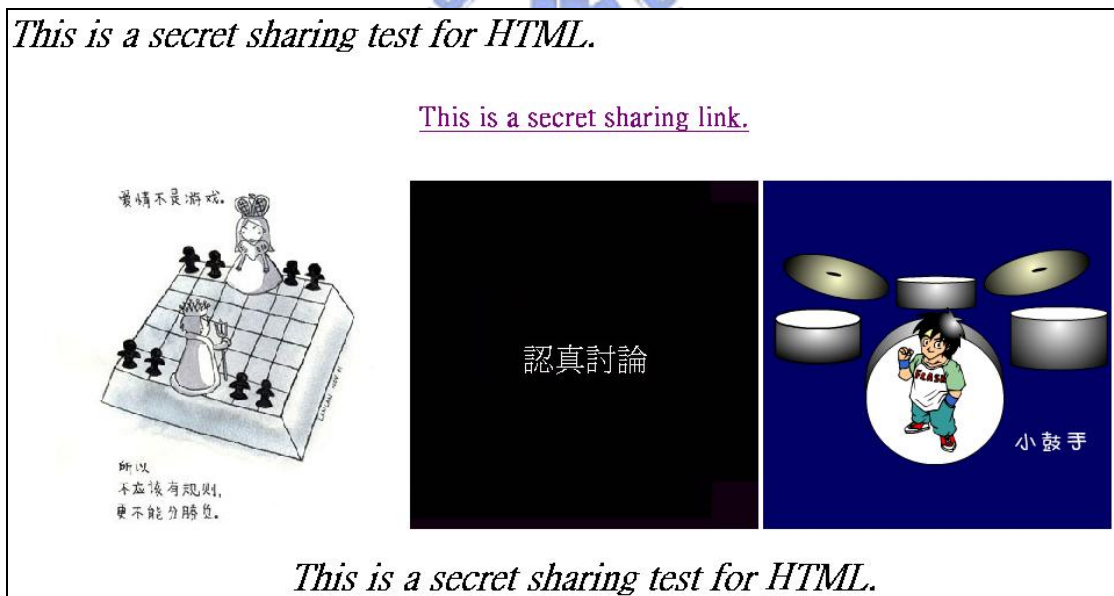


Figure 4.10 Recovered secret HTML document.

<http://www.cis.nctu.edu.tw/~gis91568/Multimedia/image03.jpg>

(a)

<http://www.cis.nctu.edu.tw/~gis91568/Multimedia/image03.cgi?2856C1B0F09906517F697DCDBC7A4B3647B39D48C398B0ACF6F714D477F77C811C8249ACF49C125BE87E08646D12129698ABD97AEC511606EC71F0C01ABEDF768792B9CB1B963CC0F7D50054CFA5A676F7103A54B575B6CECA0D1A2D54BC24EF8CA547877122020072F90D350AD708C1354055878F0DBC>

(b)

<http://www.cis.nctu.edu.tw/~gis91568/Multimedia/image03.cgi?2856C1B0F09906517F697DCDBC7A4B3647B39D48C398B0ACF6F714D477F77C811C8249ACF49C125BE87E08646D12129698ABD97AEC511606EC71F0C01ABEDF768792B9CB1B963CC0F7D50054CFA5A676F7103A54B575B6CECA0D1A2D54BC24EF8CA547877122020072F90D350AD708C1354055878F0DBC>

(c)

Figure 4.11 Image components. (a) secret image component; (b) the corresponding stego-image component of the first stego-HTML document; (c) the corresponding stego-image component of the second share HTML.

Chapter 5

Steganographic Method for Tamper Proofing of HTML Shares

Introduction

In Chapter 4, a secret HTML document can be encoded into several pieces of share data which are embedded into HTML documents to yield stego-HTML documents of the same styles and distributed among the participants of the secret sharing activities. The stego-HTML documents will be regarded as common files by common users. However, to prevent the stego-HTML documents from being modified intentionally or accidentally, it is desired to add authentication signals into the stego-HTML documents so that the documents can be checked for their fidelity and integrity. It is also desired simultaneously to maintain the steganographic effect in the resulting documents. The result of this approach is the proposed steganographic method for tamper proofing of HTML shares described in this chapter.

In the remainder of this section, the proposed idea for authentication of shares is first described and an overview of processes is then presented. In the following sections, the details of the proposed process for creating steganographic effects on the embedded authentication signals in the shares will be described in Section 5.2, some experimental results will be illustrated in Section 5.3, and finally, some discussions and a summary will be given in Section 5.4.

5.1.1 Idea of Proposed Approach

In order to keep stego-HTML documents from being changed intentionally or accidentally by participants or illicit users, authentication signals for the share data in a stego-HTML document are embedded in the document. In addition, because the components in an HTML document are discrete, a component can be removed from or added to the HTML document without affecting the other components. Consequently, the authentication signal embedding and extraction processes are performed component by component in this study.

5.1.2 Overview of Processes

In the following, the proposed authentication signal embedding process is described in Section 5.1.2.1 and the proposed process of authentication signal extraction and document verification is described in Section 5.1.2.2.

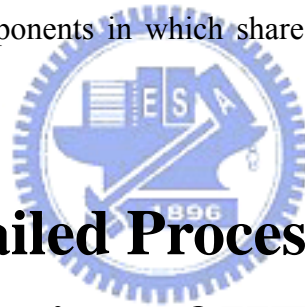
5.1.2.1. Authentication signal embedding process

Suppose the stego-HTML document H_j of the j th participant is to be processed. The authentication signal embedding process is shown in Figure 5.1.

First, stego-text component A_j , a piece of share data T_j of A_j , and share data of stego-non-text components are extracted by parsing H_j . Stego-text component A_j is treated as the cover of authentication signals and share data T_j are used to generate authentication signals TA_j , which are then embedded into A_j to form AA_j . Tags in H_j are also utilized as the cover of the authentication signals and share data $C_{1,j}, \dots, C_{v,j}$ are used to produce authentication signals CA_j . By embedding CA_j in the tags of H_j and replacing A_j in H_j with AA_j , the j th share AH_j with a steganographic effect is created.

5.1.2.2. Authentication Signal Extraction and Verification Process

While verifying a share, the process shown in Figure 5.2 is performed, which verifies the fidelity of components in an HTML document AH_j . The input AH_j is first parsed to get stego-text component AA_j , share data C_{1j}, \dots, C_{vj} , of stego-non-text components and stego-non-text component authentication signals CA_j . After extracting share data T_j of the stego-text component of AH_j and the corresponding authentication signals TA_j from AA_j , T_j and TA_j then are used for verifying the fidelity and integrity of T_j . In the same way, C_{1j}, \dots, C_{vj} are verified by using CA_j and themselves. A verified share VAH_j finally is obtained by collecting the authentication results and marking the components in which share data are verified to have been tampered with.



Proposed Detailed Processes for Authentication of HTML Shares

In this section, non-text and text component authentication processes will be investigated each in detail. In Section 5.2.1, the non-text component authentication process is described; and in Section 5.2.2, the text component authentication process is described.

5.1.3 Non-text Component Authentication

The pattern authentication technique mentioned in Section 2.3.6 can be applied to stego-non-text components of stego-HTML documents by importing a certain data

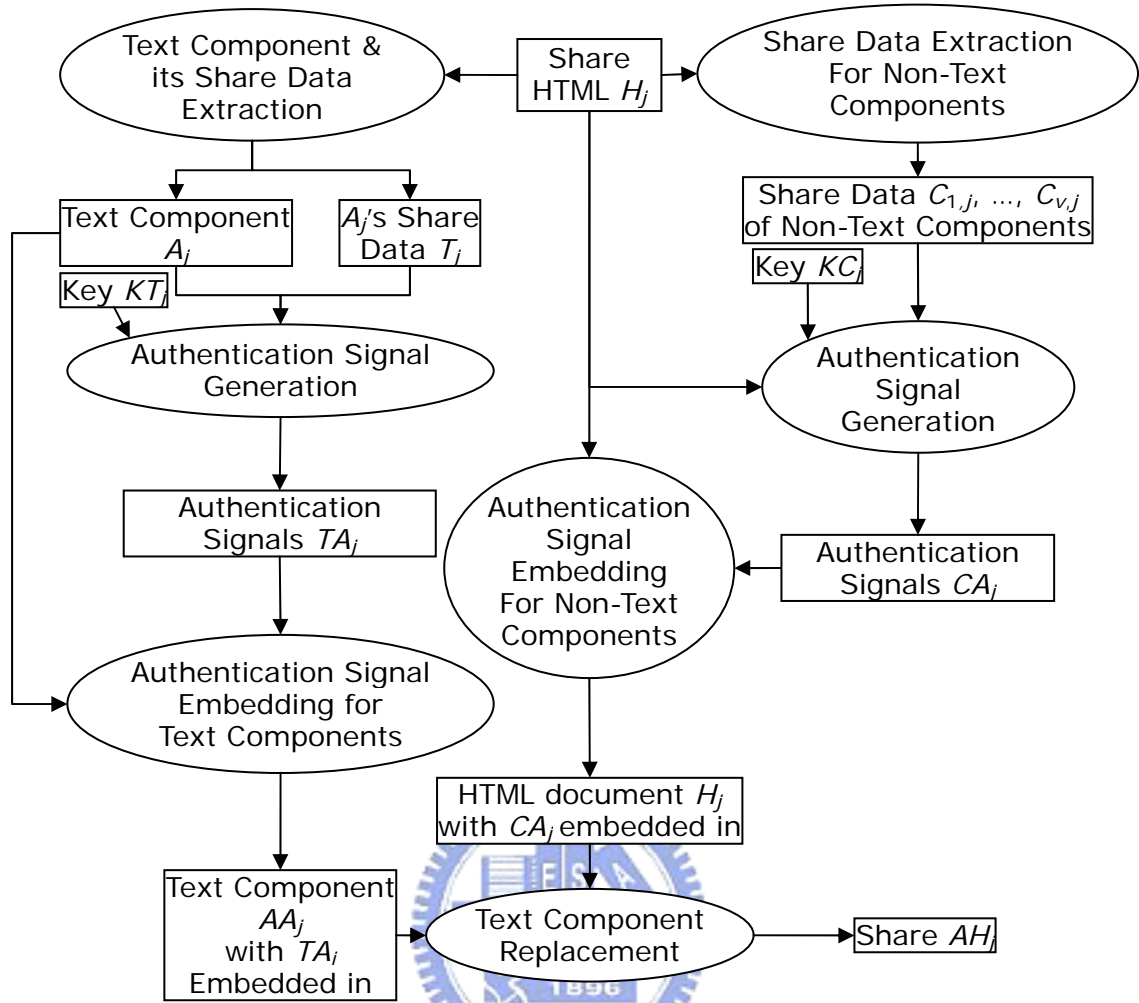


Figure 5.1 Flowchart of process of embedding authentication signals.

hiding technique for HTML documents. Because the letters of tag titles and attribute names are not case-sensitive, information can be hidden in by controlling the case of the letters. Therefore, authentication signals can be embedded inside the tags of HTML by utilizing the property mentioned above.

Let H_j be the stego-HTML document, K be the corresponding key, AH_j be the share, the uppercase of a letter denote '1', the lowercase of a letter denote '0', $AUTHEN_GEN(s, h, k)$ be the procedure, proposed in Section 2.3.6, of authentication signal generation with three input parameters share data s , space size h for hiding authentication signals and a key k , $R_k()$ be the random number generator of key k , $x(i)$ denote the i th bit of string x , $L(x)$ denote the length of a string x and $N(x)$ denote the number of letters of tag titles and attribute names in an HTML document x . The

proposed authentication signal embedding process for share data in stego-non-text components is as follows:

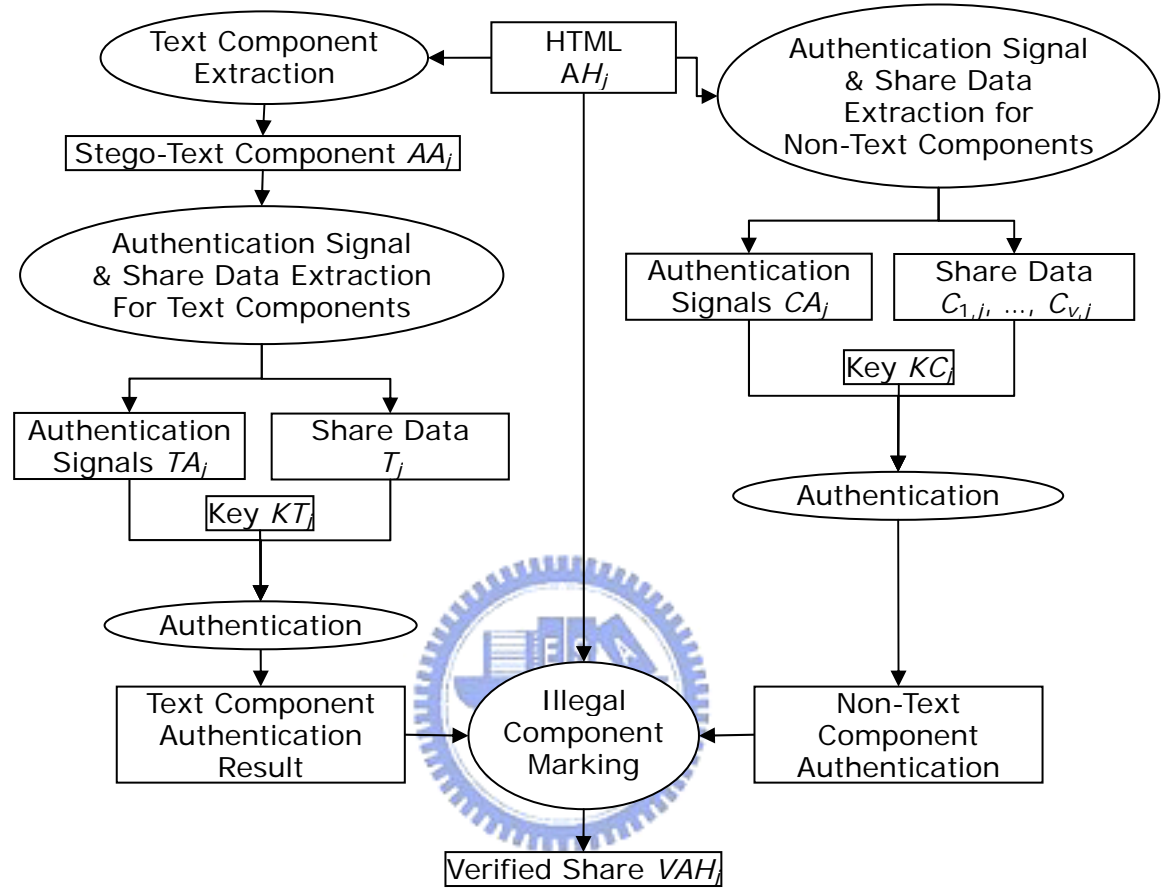


Figure 5.2 Flowchart of share authenticating process.

Algorithm 5.1 *Authentication signal embedding for stego-non-text components.*

Input: H_j, K .

Output: AH_j .

Steps.

1. Parse and extract stego-non-text components $L_{1,j}, \dots, L_{v,j}$ in H_j , where v is the number of stego-non-text components.
2. Extract share data $C_{1,j}, \dots, C_{v,j}$ from $L_{1,j}, \dots, L_{v,j}$, respectively.
3. Compute $LEN = \lfloor N(H_j) / v \rfloor$.

4. For $i = 1$ to v , do the following two steps.
 - (iv) Generate sub-key $k_i = R_k()$.
 - (v) Compute $AH_{i,j} = AUTHEN_GEN(C_{i,j}, LEN, k_i)$.
5. Concatenate $AH_{1,j}, \dots, AH_{v,j}$ to form CA_j .
6. For $i = 1$ to $L(CA_j)$, if $CA_j(i)$ equals '0', set the case of the i th letter of the tag titles and the attribute names in H_j to be lowercase; else, uppercase.
7. Set AH_j to be the modified H_j .

At the component authentication stage, the authentication signal extraction and verification process is as follows:

Algorithm 5.2 *Verification of stego-non-text components.*

Input: AH_j .

Output: a verified share VAH_j .

Steps.

1. Parse and extract authentication signals CA_j hidden in the letters of the tag titles and the attribute names in AH_j .
2. Parse and extract stego-non-text components $L_{1,j}, \dots, L_{v,j}$ in AH_j , where v is the number of stego-non-text components in AH_j .
3. Extract share data $C_{1,j}, \dots, C_{v,j}$ from $L_{1,j}, \dots, L_{v,j}$, respectively.
4. Compute $LEN = \lfloor N(x) / v \rfloor$.
5. For $i = 1$ to v , do the following two steps.
 - (i) Generate sub-key $k_i = R_k()$.
 - (ii) Compute $AH_{i,j} = AUTHEN_GEN(C_{i,j}, LEN, k_i)$.
6. Divide CA_j into v strings $CA_{1,j}, \dots, CA_{v,j}$, which are of the length LEN in bit.
7. Set $VAH_j = AH_j$.
8. For $i = 1$ to v , if $CA_{i,j}$ is different from $AH_{i,j}$, set $L_{i,j}$ in VAH_j to be a

pre-defined error component of the type of L_{ij} .

5.1.4 Text Component Authentication

The pattern authentication technique mentioned in Section 2.3.6 can also be applied to the stego-text component of stego-HTML documents by adopting another data hiding technique for the text components of HTML documents. Here, the technique of hiding authentication signals of share data in between-word spaces of a stego-text component, which was proposed in Section 2.3.3 and applied in 4.2.2, is exploited again with modifications.

By specifying the relative positions between the authentication signals and the share data both in a between-word space, a data hiding technique for authentication of the share data in the stego-text component of each stego-HTML document is described in the following.

Let H_j be a stego-HTML document, K be the key of H_j , N be the specified number of bits of authentication signals hidden in a space, AH_j be the share derived from H_j , $AUTHEN_GEN(s, h, k)$ denote the procedure, proposed in Section 2.3.6, of authentication signal generation with three input parameters share data s , space size h for hiding authentication signals and a key k , $R_k()$ be the random number generator of key k , $L(x)$ denote the length of string x and $N(x)$ denote the number of inter-word, interline and inter-sentence spaces in string x . The process of embedding authentication signals for authentication of share data in the stego-text component of a stego HTML document is as follows.

Algorithm 5.3 *Authentication signal embedding for stego-text components.*

Input: H_j, K .

Output: AH_j .

Steps.

1. Parse and extract stego-text component segments $S_{1,j}, \dots, S_{m,j}$ between two successive tags in H_j , where m is the total number of segments of H_j 's stego-text component.
2. Extract strings $E_{1,j}, \dots, E_{m,j}$ of translated share data from $S_{1,j}, \dots, S_{m,j}$, respectively.
3. Translate $E_{1,j}, \dots, E_{m,j}$ back into share data $SD_{1,j}, \dots, SD_{m,j}$.
4. Concatenate $SD_{1,j}, \dots, SD_{m,j}$ together to form a synthetic share data SD_j .
5. Set a key $k = R_k()$.
6. Set authentication signals $TA_j = AUTHEN_GEN(SD_j, m \times N, k)$.
7. Divide TA_j into m strings $TA_{1,j}, \dots, TA_{m,j}$.
8. For $i = 1$ to m , set $TASD_{i,j} = TA_{i,j}SD_{i,j}$.
9. Translate $TASD_{1,j}, \dots, TASD_{m,j}$ into strings $AE_{1,j}, \dots, AE_{m,j}$, which are made up of tab symbols, new-line symbols, and space symbols.
10. For $i = 1$ to m , embed $\lfloor L(E_{i,j})/N(C_{i,j}) \rfloor + N + 1$ symbols of $AE_{i,j}$ into inter-word, inter-line, and inter-sentence spaces in $S_{i,j}$ to form $AS_{i,j}$.
11. Set $AH_j = H_j$.
12. For $i = 1$ to m , replace $S_{1,j}, \dots, S_{m,j}$ in AH_j with $AS_{1,j}, \dots, AS_{m,j}$.

The corresponding authentication signal extraction and verification process is described in the following.

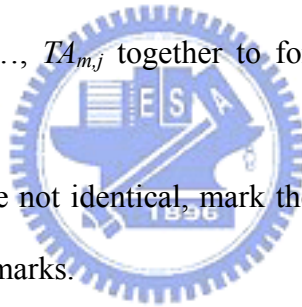
Algorithm 5.4 *Verification of stego--text components.*

Input: AH_j, K .

Output: VAH_j .

Steps.

1. Parse and extract m stego-text component segments $AS_{1,j}, \dots, AS_{m,j}$ between two successive tags in AH_j , where m is the total number of segments of H_j 's stego-text component.
2. Extract translated strings $AE_{1,j}, \dots, AE_{m,j}$ of share data from $AS_{1,j}, \dots, AS_{m,j}$, respectively.
3. Translate $AE_{1,j}, \dots, AE_{m,j}$ back into strings $TASD_{1,j}, \dots, TASD_{m,j}$.
4. For $i = 1$ to m , separate $TASD_{i,j}$ into $TA_{i,j}$ and $SD_{i,j}$.
5. Concatenate $SD_{1,j}, \dots, SD_{m,j}$ together to form a synthetic share data SD_j .
6. Set a key $k = R_K()$.
7. Set the computed authentication signals CTA_j to be $AUTHEN_GEN(SD_j, m \times N, k)$.
8. Concatenate $TA_{1,j}, \dots, TA_{m,j}$ together to form the extracted authentication signals ETA_j .
9. If ETA_j and CTA_j are not identical, mark the share data hidden in $AS_{1,j}, \dots, AS_{m,j}$ with question marks.



Experimental Results

Some experimental results are described in this section. Figure 5.3 in the following is a stego-HTML document, in which two stego-text component segments are located in the top and the bottom, respectively, a stego-image component is in the middle-left side, a stego-video component is at the center, and a stego-flash component is in the middle-right side. Figure 5.4 is a share of the document and the display of it on browsers is identical to that of the document itself. Suppose that the share data in the share is modified. The authentication result is shown as Figure 5.5. The source path, a dynamic link, of the image is replaced with that of the pre-determinate image and the strings of symbols, which are translated from the

corresponding share data and located at between-word spaces, are marked as a string of question marks.

Discussions and Summary

In order to ensure the fidelity and integrity of the share data of the components of stego-HTML documents, authentication signals of the share data are embedded in each stego-HTML document. For the share data in the stego-text component, authentication signals are distributed and hidden into between-word spaces by controlling the number of authentication signals hidden in per space. As for share data in the stego-non-text components, the authentication signals are embedded in the tags of each stego-HTML document by controlling the case of the letters of the tag titles and the attribute names in the tags. The process of authenticating the share data is conducted component by component.



Figure 5.3 A stego-HTML document.

There are so many accidents in the world.

Introduction to Email



They are the reason Dr. David Ho has come to China.

Figure 5.4 A share of the HTML document in Figure 5.3.

There????are????so????many????accidents????in????the????world.

Introduction to Email



*They????are????the????reason????Dr.???? David????Ho????has????
come????to????China.*

Figure 5.5 A verified share of a tampered HTML share.

Chapter 6

Hierarchical Secret Sharing with Steganographic Effects for E-mail Documents

Introduction

Because of the population of exchanging messages via e-mail, most of users use e-mail for communication frequently. In addition, properties of e-mail for carrying any kinds of files by e-mails make it possible and convenient to put a bundle of related files in an e-mail. Therefore, applying secret sharing to e-mails that contain secret files of various types is worthwhile. And the hierarchical secret sharing is adopted and applied to secret e-mails.

In the remainder of this section, properties of e-mail are first investigated and then an overview of sharing and recovery processes is proposed. In the following sections, the detailed processes for sharing secret e-mails are described in Section 6.2, experimental results of sharing secret e-mail documents are shown and illustrated in Section 6.3, and, finally, discussions and summary are proposed.

6.1.1 Properties of E-mail Documents

One of e-mail formats, called multipurpose internet mail extensions (MIME), is a specification for formatting non-ASCII messages so that they can be sent over the Internet. Some properties about MIME for sharing secret e-mails are investigated in the following. In Section 6.1.1.1, an overview of MIME format is proposed. Next,

three main components of e-mail is described in Section 6.1.1.2. Finally, content transfer encoding methods for transmitting an e-mail successfully are introduced in Section 6.1.1.3.

6.1.1.1. Overview of e-mail format: multipurpose internet mail extensions (MIME)

Each e-mail of MIME is of the form as shown in Figure 6.1. A header which contains necessary information and the description of its corresponding body is followed by the corresponding body. A body includes one or many such header-body pairs or data of a file of any type. Therefore, the important parts of an e-mail are some information of headers and bodies that contains file data.

6.1.1.2. Main components: e-mail header, content, and attachment



In the viewpoint of e-mail users, an e-mail contains three main components: e-mail header components, content components and attachment components. An e-mail contains one and only one e-mail header component and one and only one content component and, however, attachment components are optional.

Information that the e-mail users can obtain from an e-mail header component on browsers is about sender's e-mail address, receivers' addresses, the subject of an e-mail, etc. Content components and attachment components are made up of a body that contains file data and a header that contains information about the file data.

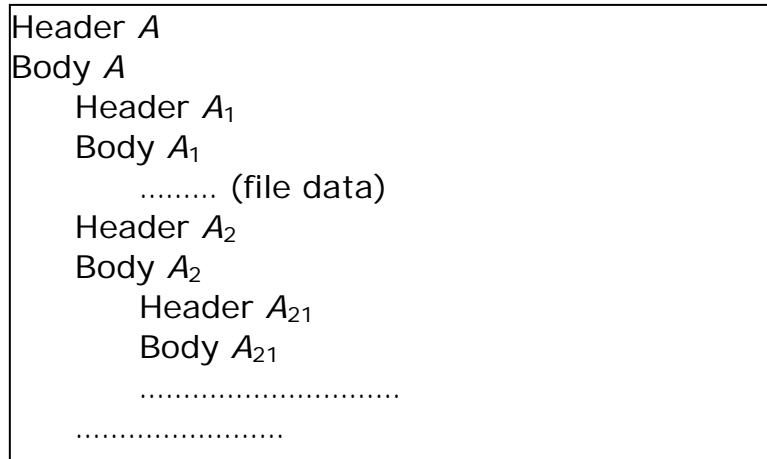


Figure 6.1 Conceptual expression of MIME format.

An e-mail header component contains many attributes in the form of title-value pair. Some attributes are necessary and constant and should appear in a header, such as sender’s e-mail address, receivers’ e-mail address, the subject of an e-mail, etc. Some attributes are conditionally necessary. Furthermore, the other attributes can be defined by any one. On the other hand, adding strings of tab and space symbols to the rear of the lines in which the information of some specific attributes is described does not affect the display of the e-mail on e-mail browsers.

An attachment component must have an attribute of title “Content-Disposition” in its corresponding header; while a content component must “not” contain such an attribute.

6.1.1.3. Content transfer encoding methods

There are many transfer encoding methods, including the methods specified in MIME and methods invented and used by an organization or somebody. Applying a transfer encoding method to an e-mail can assure successful transmission on the internet. Two main methods of content transfer encoding specified in MIME format will be introduced.

The first method is the quoted-printable encoding method which represents file

data as symbols that largely correspond to printable characters in the US-ASCII character set. It encodes the data such that the resulting symbols are unlikely to be changed by mail transport and can be easily recognized by human beings.

The other is called base-64 method. The method translates data into a string of 64 specified symbols, which correspond to different printable characters in the US-ASCII character set, conditionally followed by a short string of a pad symbol '=' at the rear of the string. The result string can not be easily understood by humans.

6.1.2 Overview of Proposed Processes

The important parts in a secret e-mail are identified, extracted, and shared among secret sharing participants. To sharing a secret e-mail with steganographic effect, the framework of the secret e-mail is preserved for share e-mails by only changing the components of the secret e-mail. An overview of secret e-mail sharing and corresponding secret recovery processed is described in Section 6.1.2.1 and 6.1.2.2, respectively.

6.1.2.1. Secret E-mail Sharing process

Suppose the number of participants of the secret e-mail is n . A secret e-mail E is first divided and classified into the three components. H is the e-mail header component, C denotes the content component, and A_1, \dots, A_v are the attachment components.

The component extraction and sharing process is illustrated in Figure 6.2. For e-mail header component H , the important parts of H are extracted, rearranged, and then shared out among H_1, \dots, H_n by hierarchical secret sharing with data magnitude control. For the content component C , just the body, the file data, is encoded and

distributed into C_1, \dots, C_n . As for the attachment component A_i , the important information in the header and the body are combined as the input of hierarchical sharing to generate n share data $A_{i,1}, \dots, A_{i,n}$. In Figure 6.3, for the j th participant, steganographic effect on share data of each component is created according to the type of the component. After applying steganography independently and accordingly to H_j, C_j , and $A_{1,j}, \dots, A_{v,j}$, the corresponding stego-components SH_j, SC_j , and $SA_{1,j}, \dots, SA_{v,j}$ are formed. The stego-e-mail H_j is then generated by referring to the framework of the secret e-mail and replacing the components with the corresponding stego-components.

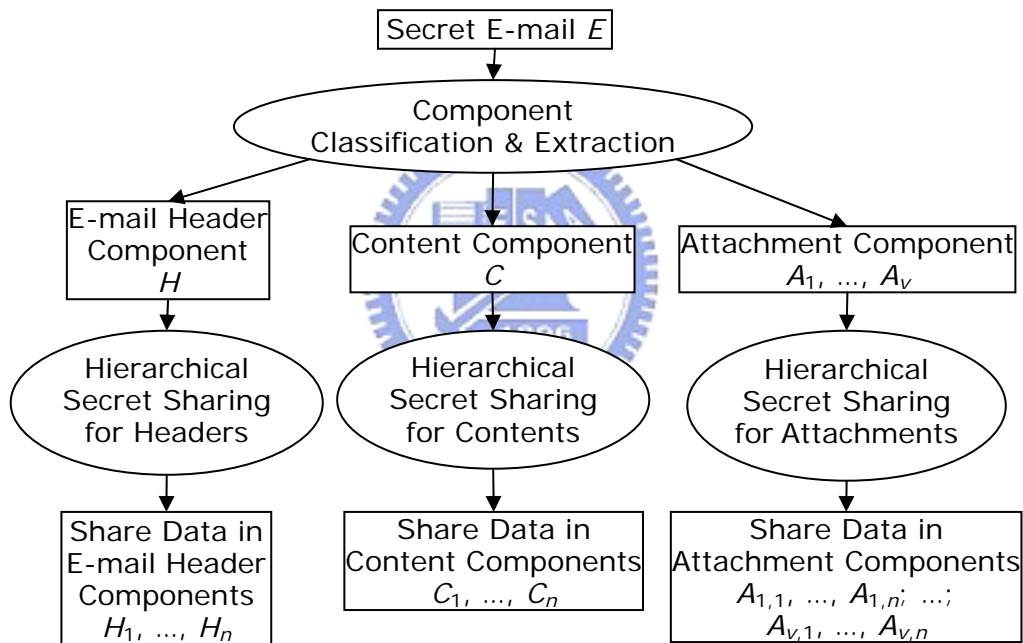


Figure 6.2 Flowchart of component extraction and sharing process.

6.1.2.2. Secret E-mail Recovery process

In the recovery process, each share e-mail is also first divided and classified into the three components. Using the j th participant's stego-e-mail E_j as an example, The stego-e-mail header component SH_j , the stego-content component SC_j and the stego-attachment components $SA_{1,j}, \dots, SA_{t,j}$ are extracted from E_j . By applying

corresponding extraction technique, share data in the components of the three types are obtained. The share data extraction is shown in Figure 6.4.

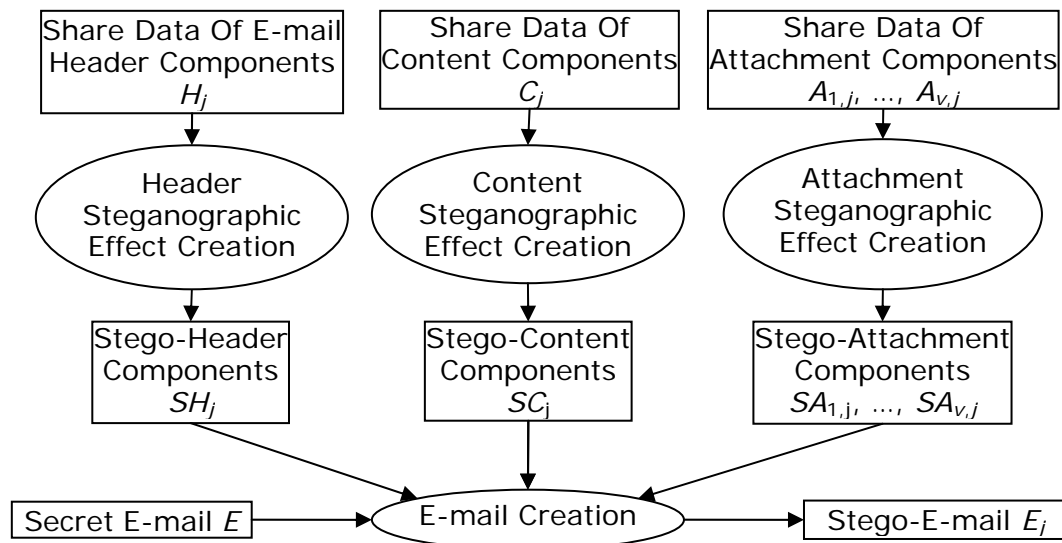


Figure 6.3 Flowchart of steganographic effect creation process.

After share data in each stego-e-mail are obtained, as shown in Figure 6.5, hierarchical secret sharing recovery with different pre-processes or post-processes according to the component type is applied to n related pieces of share data from participants. For instance, n piece of share data H_1, \dots, H_n extracted from their own stego-e-mail header components are recovered to form a string. Then, according to the pre-determinate format, the string is parsed to get the values of each important attribute. Finally, a recovered e-mail header component is generated by creating attributes with the corresponding values.

Proposed Detailed Processes for Sharing Secret E-mail

While parsing an e-mail, a component got each time is just one of the three types. The sharing and recovery procedures for the e-mail except the detailed processes of sharing the components of the three types are proposed in the following paragraphs.

After describing the dual procedures, the corresponding sharing process of the three components is proposed next.

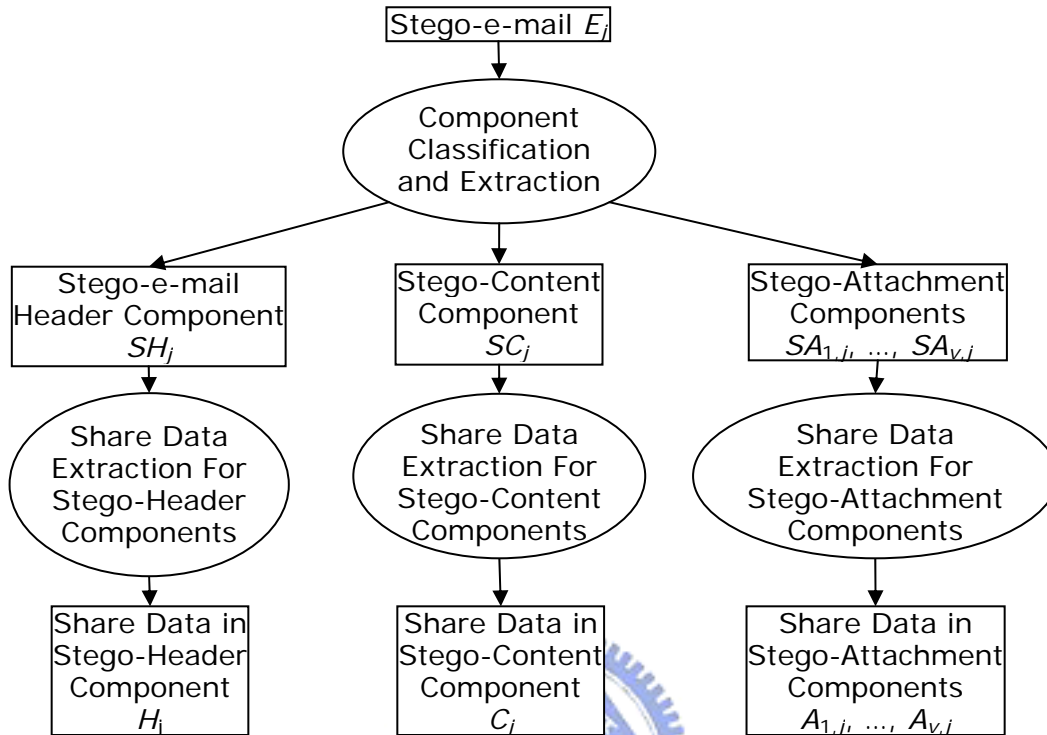


Figure 6.4 Flowchart of process of share data extraction.

Let the secret e-mail be E , the number of participants of E is n , $EMAIL_HEADER_SHARING(e)$ denoted the e-mail header sharing process of an component e , $CONTENT_SHARING(e)$ denote the content component sharing process of an component e , and $ATTACHMENT_SHARING(e)$ denoted the attachment sharing process of an component e . The sharing procedure is as below.

Algorithm 6.1 *Secret sharing of secret e-mails with steganographic effect.*

Input: E .

Output: n stego-e-mails E_1, \dots, E_n .

Steps.

1. Parse E and extract m components e_1, \dots, e_m in turn from E , where m is the number of components.
2. For $i = 1$ to m , do the following steps.

- (i) If e_i is an e-mail header component, do $EMAIL_HEADER_SHARING(e_i)$ to generate $Se_{i,1}, \dots, Se_{i,n}$.
 - (ii) Else if e_i is a content component, do $CONTENT_SHARING(e_i)$ to generate $Se_{i,1}, \dots, Se_{i,n}$.
 - (iii) Else if e_i is an attachment component, do $ATTACHMENT_SHARING(e_i)$ to generate $Se_{i,1}, \dots, Se_{i,n}$.
3. For $j = 1$ to n , do the following steps.
 - (i) Set $E_j = E$.
 - (ii) For $i=1$ to m , replace e_i in E_j with $Se_{i,j}$.

Let $EMAIL_HEADER_RECOVERY(n, e_1, \dots)$ denoted the e-mail header sharing recovery process of n related components e_1, \dots, e_n , $CONTENT_RECOVERY(n, e_1, \dots)$ denoted the content component sharing recovery process of n related components e_1, \dots, e_n , and $ATTACHMENT_RECOVERY(n, e_1, \dots)$ denote the attachment sharing recovery process of n related components e_1, \dots, e_n . The corresponding recovery procedure is as follows.

Algorithm 6.2 *Recovery of secret e-mails.*

Input: E_1, \dots, E_n .

Output: a recovered secret e-mail RE .

Steps.

1. For $j = 1$ to n , do the following steps.
 - (i) Parse E_j and extract m stego components $Se_{1,j}, \dots, Se_{m,j}$ in turn from E_j .
2. Set $RE = E_k$, where k is an integer randomly selected from $\{1, \dots, n\}$.
3. For $i = 1$ to m , do the following steps.
 - (i) If $Se_{i,1}, \dots, Se_{i,n}$ all are stego-e-mail header components, do $EMAIL_HEADER_SHARING(n, Se_{i,1}, \dots, Se_{i,n})$ to obtain e_i .

- (ii) Else if $Se_{i,1}, \dots, Se_{i,n}$ all are stego-content components, do $CONTENT_SHARING(n, Se_{i,1}, \dots, Se_{i,n})$ to obtain e_i .
- (iii) Else if $Se_{i,1}, \dots, Se_{i,n}$ all are stego-attachment components, do $ATTACHMENT_SHARING(n, Se_{i,1}, \dots, Se_{i,n})$ to obtain e_i .
- (iv) Replace $Se_{i,k}$ in RE with e_i .

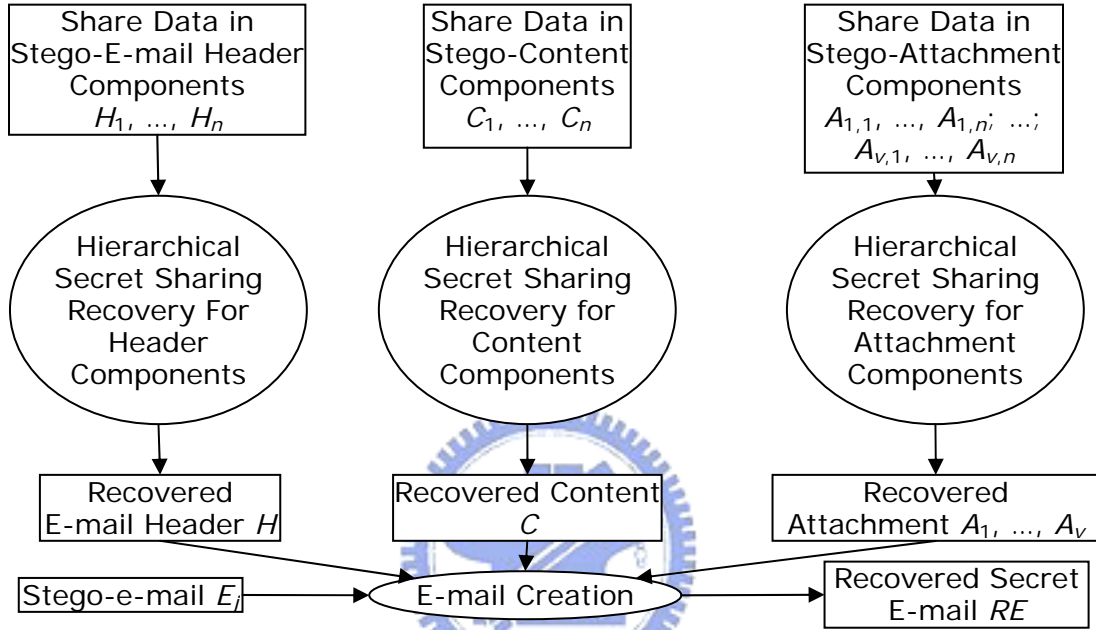


Figure 6.5 Flowchart of recovery process of secret e-mail.

The hierarchical secret sharing with data magnitude control proposed in Section 2.3.1 will be applied in the following sections. The process of sharing e-mail header components is first proposed in Section 6.2.1. In Section 6.2.2, the sharing process for content components is described. Finally, the attachment sharing process is proposed in Section 6.2.3.

6.1.3 Sharing E-mail Header

The steganographic technique proposed in Section 2.3.5 is adopted with modifications to create steganographic effect on the share data of the important

information of the secret e-mail header component. Let n be the number of secret sharing participants, $HIERARCHI_SHARING(s)$ denote the procedure of hierarchical secret sharing with data magnitude control of string s and the titles “Subject”, “From”, “To”, “Date”, “Cc” and “Bcc” denote $T_1, T_2, T_3, T_4,$ and $T_5,$ respectively. The modified steganographic procedure is as follows.

Algorithm 6.3 *Secret sharing of e-mail header components with steganographic effect.*

Input: an e-mail header component H .

Output: n stego-e-mail header components H_1, \dots, H_n .

Steps.

1. For $i = 1$ to 5, extract all the values $V_{i,1}, \dots, V_{i,t_i}$ of T_i in H , where t_i denotes the number of the values of T_i .
2. For $i = 1$ to 5 and for $j = 1$ to t_i , do the following steps.
 - (i) $HIERARCHI_SHARING(V_{i,j})$ to generate $V_{ij,1}, \dots, V_{ij,n}$.
 - (ii) Apply base-64 encoding method to $V_{ij,1}, \dots, V_{ij,n}$ to get $TV_{ij,1}, \dots, TV_{ij,n}$.
3. For $k = 1$ to n , do the following steps.
 - (i) Create a string S of the following form:
$$t_1 | TV_{1,1,k} | \dots | TV_{1,t_1,k} | t_2 | TV_{2,1,k} | \dots | TV_{2,t_2,k} | \dots | t_5 | TV_{5,1,k} | \dots | TV_{5,t_5,k} .$$
 - (ii) Create an attribute A of title “X-scanInfo” and value S .
 - (iii) For $i = 1$ to 5, randomly select a pseudo value v_i of T_i from the corresponding database.
 - (iv) Create an empty pseudo e-mail header H_k .
 - (v) for $i=1$ to 5, Add an attribute of title T_i and value v_i into H_k .
 - (vi) Add attribute A of title “X-scanInfo” and value S into H_k .

(vii) Add other necessary attributes into H_k .

Let $HIERARCHI_RECOVERY(n, s_1, \dots)$ denote the procedure of hierarchical secret sharing recovery of string s_1, \dots, s_n . The corresponding recovery procedure is as follows.

Algorithm 6.4 Recovery of secret e-mail header components.

Input: H_1, \dots, H_n .

Output: Recovered e-mail header component RH .

Steps.

1. For $k = 1$ to n , do the following steps.
 - (i) Extract the value S of the attribute of title “X-ScanInfo” from H_k .
 - (ii) Separate S into $TV_{1,1,k}, \dots, TV_{1,t_1,k}; TV_{2,1,k}, \dots, TV_{2,t_2,k}; \dots; TV_{5,1,k}, \dots, TV_{5,t_5,k}$
2. For $i = 1$ to 5 and $j = 1$ to t_i , do the following steps.
 - (iii) Apply inverse base-64 encoding method to $TV_{i,j,1}, \dots, TV_{i,j,n}$ to obtain $V_{i,j,1}, \dots, V_{i,j,n}$.
 - (iv) Recovered value $RV_{i,j} = HIERARCHI_RECOVERY(n, V_{i,j,1}, \dots, V_{i,j,n})$.
3. Set $RH = H_1$.
4. for $i=1$ to 5, replace the value of the attribute of title T_i with value $RV_{i,1}, \dots, RV_{i,t_i}$ in RH .

Here, share data is just transformed into an attribute value. One may transform the share data into many values belonging to different self-defined attributes to increase the security of the share data.

6.1.4 Sharing Content Component

A content component in an e-mail is in the form of pure text or HTML. If a

content component c is of pure text, the steganographic technique proposed in Section 2.3.2 is independently applied to the shares generated by $HIERARCHI_SHARING(c)$. As for the component of the form HTML, HTML files are used for creating steganographic effect. Let HDB be the database of HTML files. The detailed procedure is as follows.

Algorithm 6.5 *Secret sharing of content components with steganographic effect.*

Input: a content component c .

Output: n stego-content component SC_1, \dots, SC_n .

Steps.

1. Generate C_1, \dots, C_n by executing $HIERARCHI_SHARING(c)$.
2. Randomly select n HTML files H_1, \dots, H_n from HDB .
3. For $i = 1$ to n , do the following steps.
 - (i) Compute the number m of between-word space outside tags of H_i .
 - (ii) Translate C_i into a string TC_i made up of tab, new-line and space symbols.
 - (iii) Compute the length l of TC_i .
 - (iv) Separate TC_i into $TC_{i,1}, \dots, TC_{i,m}$, each is of the length $\lceil l/m \rceil$ or shorter.
 - (v) For $j = 1$ to m , replace of the j th between-word space of H_j with $TC_{i,j}$.

While recovery, the procedure is performed as follows.

Algorithm 6.6 *Recovery of secret content components.*

Input: n stego-content components SC_1, \dots, SC_n .

Output: a recovered content component Rc .

Steps.

1. For $i = 1$ to n , do the following steps.

- (i) Extract strings $TC_{i,1}, \dots, TC_{i,m}$ of tab, new-line, and space symbols from the between-word space SC_i .
 - (ii) Concatenate $TC_{i,1}, \dots, TC_{i,m}$ to form TC_i .
 - (iii) Translate back to original share data C_i .
2. $Rc = HIERARCHI_RECOVERY(n, C_1, \dots, C_n)$.

6.1.5 Sharing Attachments

An attachment component in an e-mail is in the form of the three types, e.g., pure text, e-mail, and binary stream. Here, an attachment component of binary stream type means that the format of the file attached in this component is unknown. If an attachment component a is of e-mail, the component is treated as a new secret e-mail and is processed by the e-mail sharing procedure again.

As for the pure text and the binary stream types, the important information of an attachment component includes the attached file name in the header part and the file data in the body part. By concatenating the file name string followed by a separate symbol '|' with the string of the file data, the resulting string s can be formed.

For the case of the pure text attachment, the hierarchical sharing technique $HIERARCHI_SHARING(.)$ is then applied to s to generate share data SA_1, \dots, SA_n . Then, the steganographic technique proposed in Section 2.3.2 can be applied to each share data SA_i to create a meaningful text.

As for the type of binary stream, the steganographic technique proposed in Section 6.2.2 is adopted and s is directly regarded as input of the procedure of the technique to generate n stego-HTML documents as the stego-attachment components.

Experimental Results

The secret e-mail document in Figure 6.6 contains a content component of pure text which appears at the top line of the browsing box, a word file attachment, an e-mail attachment, and an attachment of a jpeg image which is also shown at the bottom of the browsing box.

Suppose the number of the participants of this secret e-mail is four. Their relationship is shown as Figure 6.7. The 3rd participant is the dominant participant, which is marked by labeling a leading star symbol in the block of the participant in Figure 6.7, of the dominant sharing between 3rd participant and 4th participant.

After hierarchical sharing, four stego-e-mails are generated and two of the four shares are shown in Figure 6.8 (a) and (b). The Microsoft word file and the jpeg image attachments of the secret e-mail are replaced with HTML files. The e-mail attachment is manipulated as another e-mail attachment in each share e-mail. The recovered secret e-mail is obtained by applying independent sharing recovery to the shares kept by 1st participant and 2nd participant, dominant sharing recovery to the shares kept by 3rd participant and 4th participant, and then cooperative sharing recovery to the two recovery results from the independent sharing and dominant sharing.

The final result is in Figure 6.9. Note that because only the important information of the e-mail attachment is kept, the size of the e-mail attachment is smaller than the original e-mail attachment.

Discussions and Summary

The hierarchical secret sharing is applied to secret e-mails. Because the locations of the important information of components are not identical, a secret e-mail is first parsed and classified into components of three different types, e.g. e-mail header, content, and attachment.

After collecting the important information and applying hierarchical sharing to the string to generate several pieces of share data, different steganographic techniques for components of different types are performed. For e-mail header components, a substitute e-mail header component is generated with share data, which is regarded as the value of a created attribute, embedded in. For content components, different steganographic techniques deal with the components of different types, e.g. pure text, or HTML and, for attachment components, different steganographic techniques are applied for different types, e.g. pure text, binary stream or e-mail. If the type of a content component or an attachment component is pure text, the corresponding share data is translated into simple sentences by the technique proposed in Section 2.3.2. Else if the type of a content component or an attachment component is HTML or binary stream, a cover HTML file is selected from an HTML database for embedding the share data into the between-word space outside of the tags in the cover HTML. As for e-mail attachment components, such a component is treated and handled as a secret e-mail. A share e-mail is eventually generated by replacing the components of the secret e-mail with the accordingly generated components.

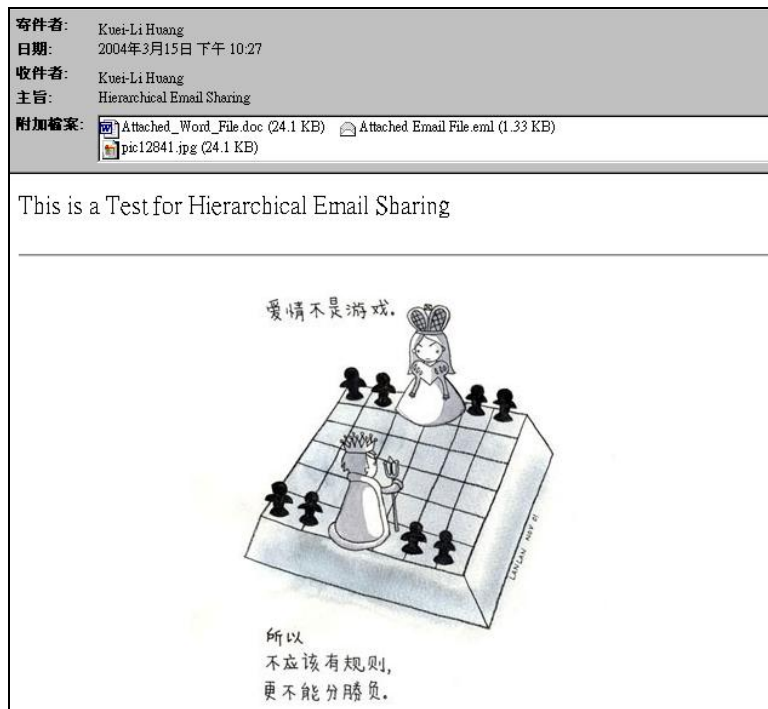


Figure 6.6 A secret e-mail.

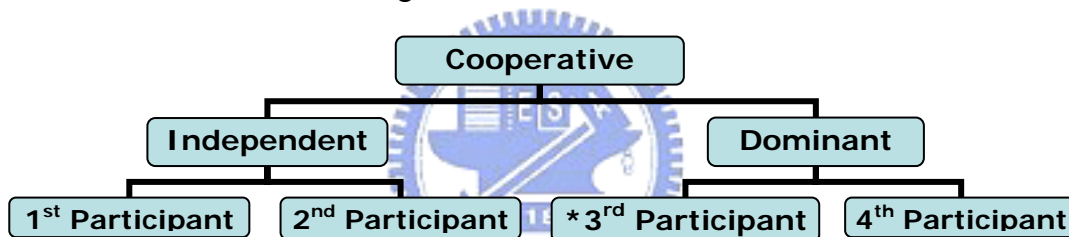
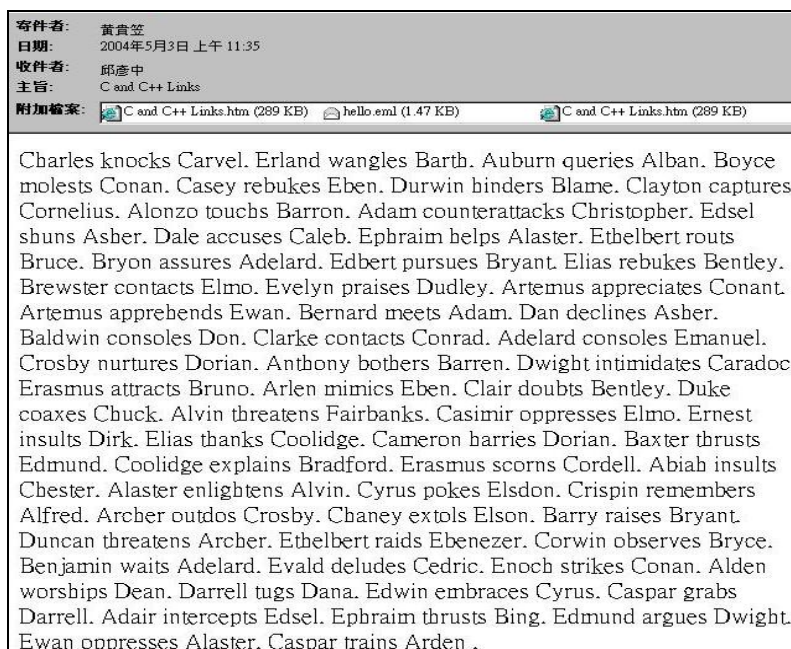
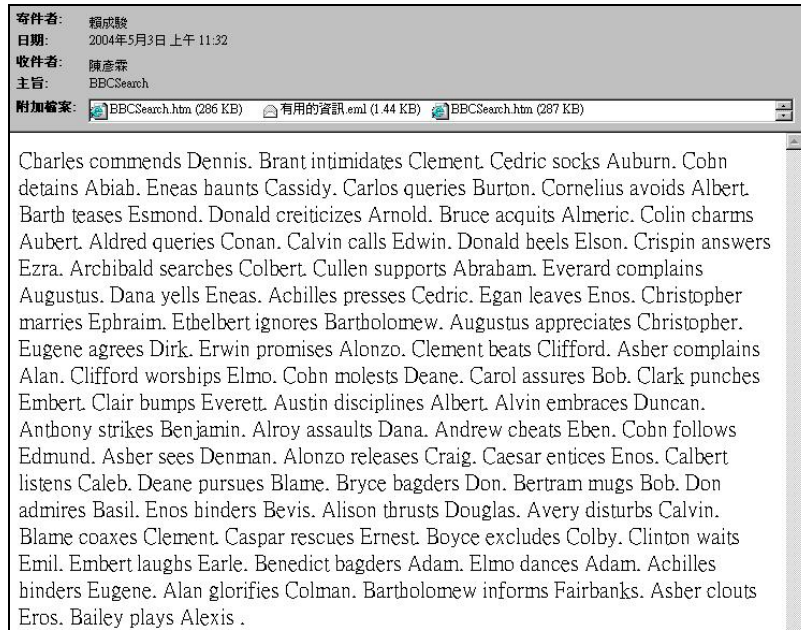


Figure 6.7 Hierarchical framework relationship among four participants.



(a)



(b)

Figure 6.8 Stego-e-mails. (a) through (b) Two of four stego-e-mails.



Figure 6.9 Recovered secret e-mail.

Chapter 7

Steganographic Method for Tamper Proofing of E-mail Shares

Introduction

The fidelity and integrity of a stego-e-mail may be perturbed or destroyed by the manipulations of illicit users, the mindlessness of the unwitting owner of the stego-e-mail, or transmission interference of an unstable network channel. In order to assure fidelity and integrity of a stego-e-mail, a steganographic method for tamper proofing of e-mail shares, stego-e-mails with authentication signals embedded in, is investigated in this chapter.

In the reminder of this section, the ideas of the steganographic method are proposed in Section 7.1.1 and an overview of processes of the proposed method is illustrated in Section 7.1.2. In the following sections, the detailed processes are proposed in Section 7.2. In Section 7.3, some experimental results are shown and, finally, discussions and a summary of this chapter are described.

7.1.1 Proposed Ideas

Due to the completeness of the framework of the e-mail format, MIME, an e-mail can be separated into several components without overlapping. Therefore, a component in the e-mail can be substituted by another component of identical type without any additional constraints. Moreover, in a stego-e-mail, by utilizing redundant space in the components, further data can be hidden in for any other purposes.

Therefore, the fidelity and integrity of a stego-e-mail can be proofed by embedding authentication signals into the stego-e-mail in advance.

7.1.2 Processes of Proposed Method

The proposed method can be roughly divided into the stage of embedding authentication signals and the stage of authentication of shares. The flowcharts of the processes of the two stages are illustrated with brief explanation. Section 7.1.2.1 will talk about the process of embedding authentication signals of share data. And the process of e-mail share data authentication will be stated in Section 7.1.2.2.

7.1.2.1. Authentication signal embedding process

Suppose a stego-e-mail is E_j , which belongs to the j th participant of a certain secret e-mail and the corresponding key for authentication is K .

In Figure 7.1, components are extracted from E_j and classified into the three different types, which is mentioned in 6.1.1.2. Different techniques of share data extraction are performed for components of different types. For instance, share data H_j in stego-e-mail header component SH_j is obtained by applying the technique of share data extraction of stego-e-mail header to SH_j .

Figure 7.2 shows that, for each piece of share data, an appropriate process of authentication signal embedding are applied to the extracted component of the share data to generate a component with authentication signals embedded in according to the type of the extracted component. Three different processes of embedding authentication signals are utilized for components of the three different types each.

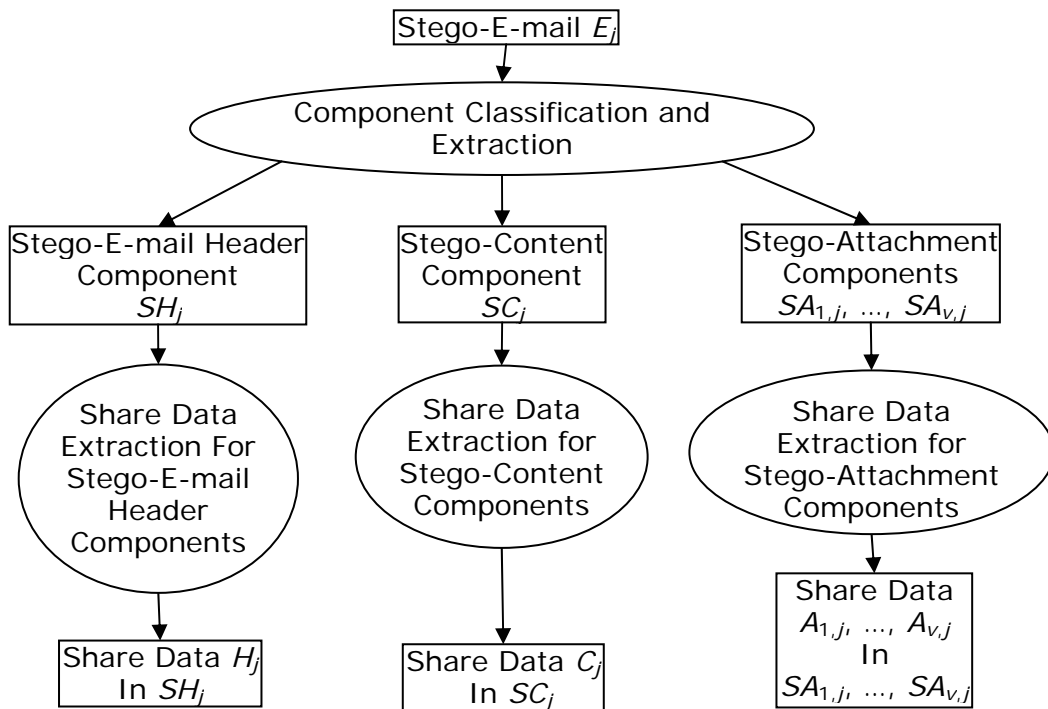


Figure 7.1 Flowchart of share data extraction process.

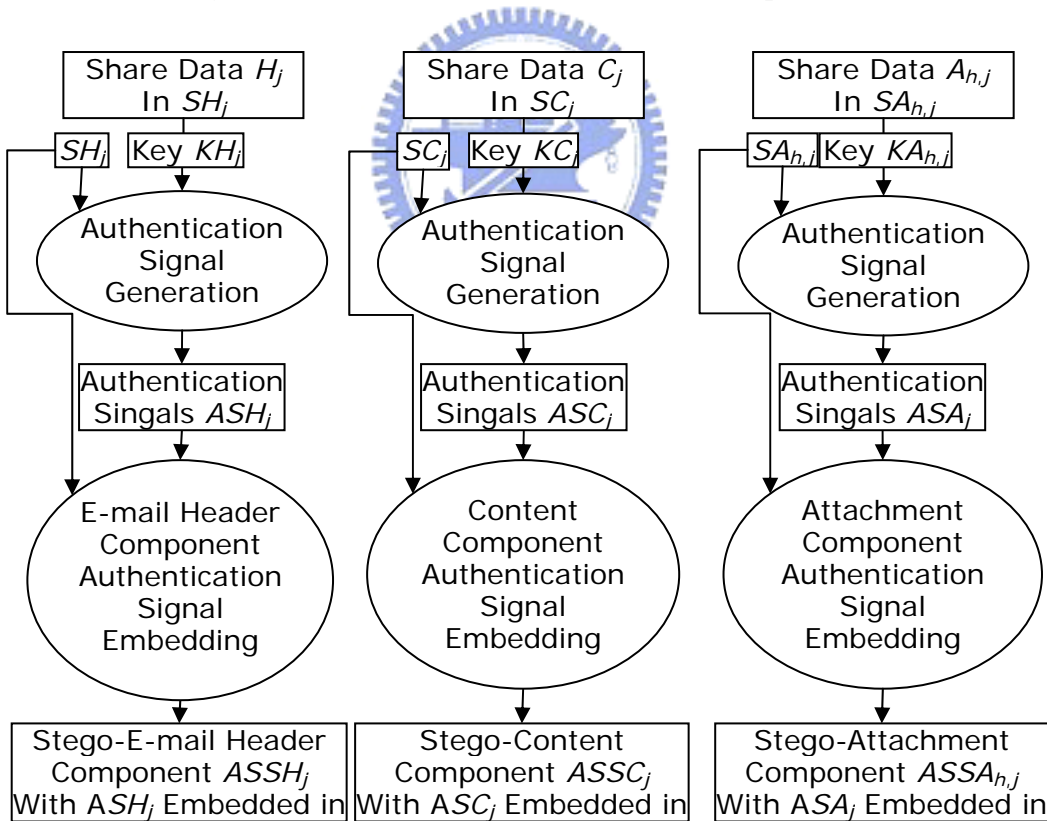


Figure 7.2 Flowchart of processes of authentication signal embedding.

As Figure 7.3 shown below, by replacing all the components in E_j with the

corresponding components, in which authentication signals are embedded, a share AE_j can be obtained. For example, the content component SC_j in E_j is replaced with the content component $ASSC_j$.

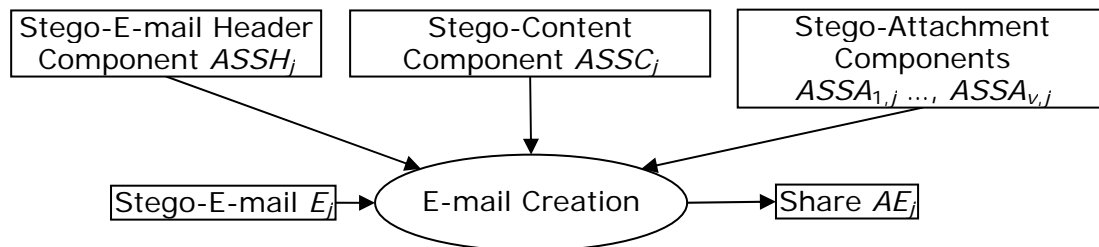


Figure 7.3 Flowchart of process of share creation.

7.1.2.2. Authentication process

In the stage of share data authentication, the first step of the process, shown in Figure 7.4, is component classification and extraction to get e-mail header component SH_j , content component SC_j , and attachment components $SA_{1,j}, \dots, SA_{v,j}$, where v is the number of the attachment components of E .

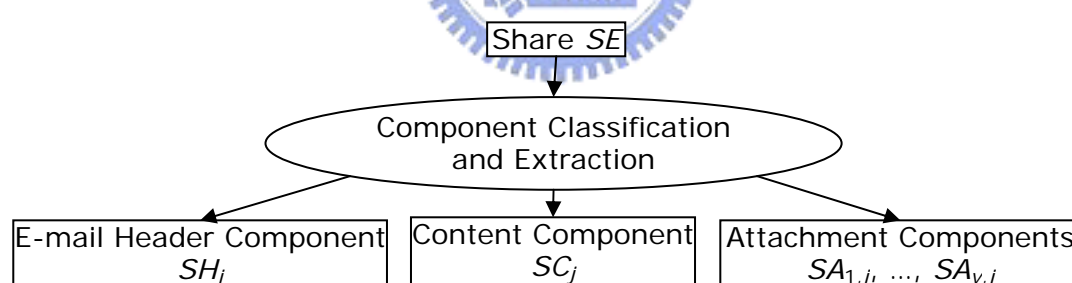


Figure 7.4 Flowchart of component extraction process.

As input, each component of E and its corresponding key are put in a process of verifying the share data in the component. There are three candidate processes of share data authentication. It depends on the type of the component to pick the right process among the three candidate processes for authentication of the share data. Each time, after the authentication of a component of E , an authentication result, is generated. Note that an authentication result just indicates that the authentication of the component is successful or failed. Figure 7.5 shows the three processes described

above in this paragraph.

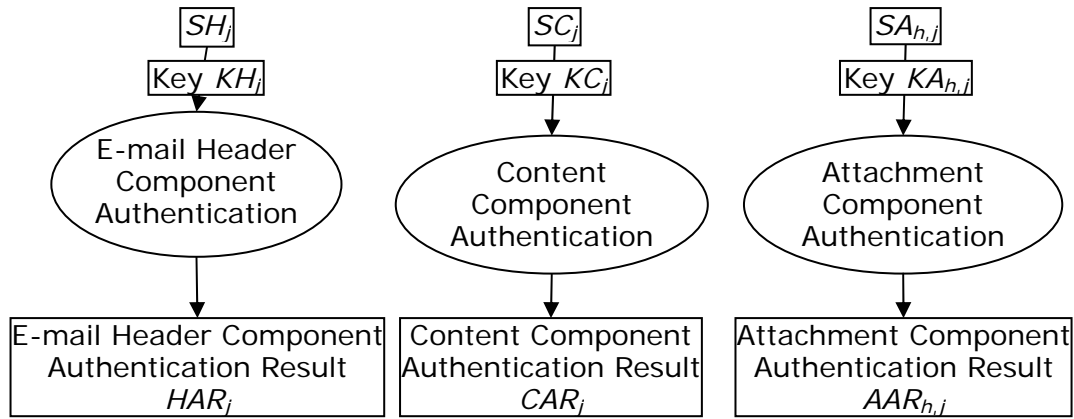


Figure 7.5 Flowchart of authenticating process.

Finally, in Figure 7.6, a component is replaced with another component which will show messages about the authentication fail or retained in the light of the corresponding authentication result.



Figure 7.6 Flowchart of the process of verified share e-mail creation.

Proposed Detailed Processes for authenticating E-mail Shares

The organization of this section is as follows:

1. An overall procedure for tamper proofing of e-mail shares is first proposed.
2. The detailed process of the authentication of e-mail header components is proposed in Section 7.2.1.
3. The authentication process for content components is described in detail in Section 7.2.2.
4. The detailed authentication process for attachment components is described in Section 7.2.3.

The overall procedure of tamper proofing of e-mail shares can be separated into the two stages, the stage of authentication signal embedding and the stage of share data authentication.

Suppose a stego-e-mail E_j , which belongs to the j th participant of a secret e-mail, has m components and a key K is used for authentication. Let $GEN_k()$ be the random number generator of the key k , $EMAIL_HEADER_AUTHEN_EMBED(c, k)$ denote the e-mail header component after embedding authentication signals into stego-e-mail header component c with an authentication key k , $CONTENT_AUTHEN_EMBED(c, k)$ denote the content component after embedding authentication signals into stego-content component c with an authentication key k and $ATTACHMENT_AUTHEN_EMBED(c, k)$ denote the attachment component after embedding authentication signals into stego-attachment component c with an authentication key k . The procedure of embedding authentication signals of share data is as follows.



Algorithm 7.1 *Embedding authentication signals of stego-e-mails.*

Input: E, K .

Output: A share AE .

Steps.

1. Parse E and extract m components Se_1, \dots, Se_m in turn from E .
2. For $i = 1$ to m , do the following steps.
3. Set a key $k=GEN_k()$.
4. If Se_i is a stego-e-mail header component, do $EMAIL_HEADER_AUTHEN_EMBED(Se_i, k)$ to generate a stego-e-mail header component ASe_i .
5. Else if Se_i is a stego-content component, do $CONTENT_AUTHEN_EMBED(Se_i, k)$ to generate a stego-content component ASe_i .

6. Else if e_i is a stego-attachment component, do $ATTACHMENT_AUTHEN_EMBED(Se_i, k)$ to generate a stego-attachment component ASe_i .
7. Set $AE=E$.
8. For $i=1$ to m , replace Se_i in AE with ASe_{ij} .

Let a share SE has n components, $EMAIL_HEADER_AUTHENTICATION(c, k)$ denote the authentication result of a stego-e-mail header component c with an authentication key k , $CONTENT_AUTHENTICATION(c, k)$ denote the authentication result of a stego-content component c with an authentication key k and $ATTACHMENT_AUTHENTICATION(c, k)$ denote the authentication result of a stego-attachment component c with an authentication key k . The authentication procedure for share data is in the following.

Algorithm 7.2 *Verification of shares.*

Input: A share SE .

Output: A verified share ASE .

Steps.

1. Parse SE and extract m components Se_1, \dots, Se_n in turn from SE .
2. For $i = 1$ to n , do the following steps.
3. Set a key $k=GEN_k()$.
4. If Se_i is a stego-e-mail header component, do $EMAIL_HEADER_AUTHENTICATION(Se_i, k)$ to generate an authentication result $ASeR_i$.
5. Else if Se_i is a stego-content component, do $CONTENT_AUTHENTICATION(Se_i, k)$ to generate an authentication result $ASeR_i$.
6. Else if e_i is a stego-attachment component, do $ATTACHMENT_AUTHENTICATION(Se_i, k)$ to generate an authentication result $ASeR_i$.
7. Set $ASE=SE$.



8. For $i=1$ to n , if $ASeR_i$ is failed, replace Se_i in AE with another component, in which error messages will shown, of Se_i 's type.

7.1.3 E-mail Header Authentication

The authentication technique proposed in Section 2.3.5 can be directly applied for the purpose of e-mail header component authentication. In the procedure of the technique, although the value of the attribute of title “X-ScanInfo” in the e-mail header in Section 2.3.5 is different from that in the e-mail header in this chapter in their generation ways, the values in the two e-mail headers are the important data that should be protected and taken care. Therefore, applying the authentication technique to a stego-e-mail header component with an authentication key can generate a stego-e-mail header component with authentication capability.

7.1.4 Content and Attachment Authentication

The body of a component of content or attachment type contains a file, in which the share data has been embedded, of HTML or pure text. Therefore, authentication signals should be hidden in the body part of a component without modifying the header part of the component.

About the component which contains a pure text file in the body part, the authentication technique proposed in 3.3.3 can be applied for the pure text file.

As for the component which contains an HTML file in the body part, the authentication signals of the share data can be hidden inside tags of the HTML file-the idea borrowed from the technique proposed in 5.2.1. By using the technique of authentication signal generation proposed in Section 2.3.6, a solution to content and

attachment component authentication is generated. Let SC be a component of content type or attachment type, K be the authentication key of SC , $AUTHEN_GEN(s, h, k)$ denote the procedure of authentication signal generation with three input parameters share data s , space size h for hiding authentication signals and a key k and $x(i)$ denote the i th bit of string x . The detailed procedure is as follows.

Algorithm 7.3 *Embedding authentication signals of stego-attachment and stego-content components.*

Input: SC, K .

Output: a stego component ASC , which is authenticable.

Steps.

1. Extract the body part B of SC .
2. Extract and concatenate each piece of the share data SD , which distributes among between-word spaces of B .
3. Compute the number N of the letters of tag titles and attribute names in B .
4. Generate the authentication signal string $AS=AUTHEN_GEN(SD, N, K)$.
5. Set $B'=B$.
6. For $i = 1$ to N , if $AS(i)$ equals to '0', set the case of i th letter of tag titles or attribute names in B' to be lowercase; else, uppercase.
7. Set $ASC=SC$.
8. Replace B in ASC with B' .

The procedure of verifying the share data in a stego-component ASC' is as follows.

Algorithm 7.4 *Verification of stego-attachment and stego-content components.*

Input: ASC', K .

Output: an authentication result AR .

Steps.

1. Extract the body part B of ASC' .

2. Parse and extract authentication signals EAS hidden in the letters of tag titles and attribute names and share data SD in B .
3. Compute the number N of the letters of tag titles and attribute names in B .
4. Generate the authentication signal string $CAS=AUTHEN_GEN(SD, N, K)$.
5. if CAS equals to EAS , $AR=true$; else, $AR=false$.

Experimental Results

The document in Figure 7.7 is a stego-e-mail before embedding authentication signals of its share data. After embedding of the authentication signals, the corresponding share is shown in Figure 7.8. The display of the share is identical to the stego-e-mail. The content of the attachment of the file name “聯合新聞網 國內要聞 綜合 台灣水資源 貧乏得可憐.htm.” is shown in Figure 7.9 If the authenticating process to the share is successful, the verified share shown in Figure 7.10 is the same as the share while displaying on browsers.

If an improper key is used as the authentication key of the share, the content of pure text as shown in the bottom of Figure 7.11 will be marked with many strings of “ERROR!!!!!!!!.” And inside the attachments, the contents of attachments of HTML will be replaced with strings of “ERROR!!!!,” as shown in Figure 7.12. Each component will be replaced with a component in which false alert messages are marked.

When the authentication key is correct but some share data in components are modified or tampered, the verified e-mail share will be like Figure 7.13. In Figure 7.13, only the share data in the component of pure text content is modified, so a bundle of strings of “ERROR!!!!!!!!” is substituted for the content of the component.

Discussions and Summary

An e-mail share is first parsed to extract components inside the e-mail share.

Each component is classified as one of the three types: e-mail header, content, component. For each e-mail header component, the authentication signals of the share data are embedded into the rear of strings of the attributes of header. The contents of the content components are of HTML or pure text. For the HTML contents of the content components, the authentication signals of share data are hidden inside tags of the HTML and, for the content components of pure text, the authentication signals are embedded by inter-word, inter-line and inter-sentence spacing. As for the attachment components, the contents of the components can be classified into three types, e.g. HTML, pure text and e-mail. The methods for embedding authentication signals of the share data in the components of HTML or pure text are the same as those applied to the content components. As for the e-mail type, the components are parsed further and divided into several components to which the corresponding methods mentioned above are applied independently.



Figure 7.7 A stego-e-mail.



Figure 7.8 The share of the stego-e-mail in Figure 7.7.

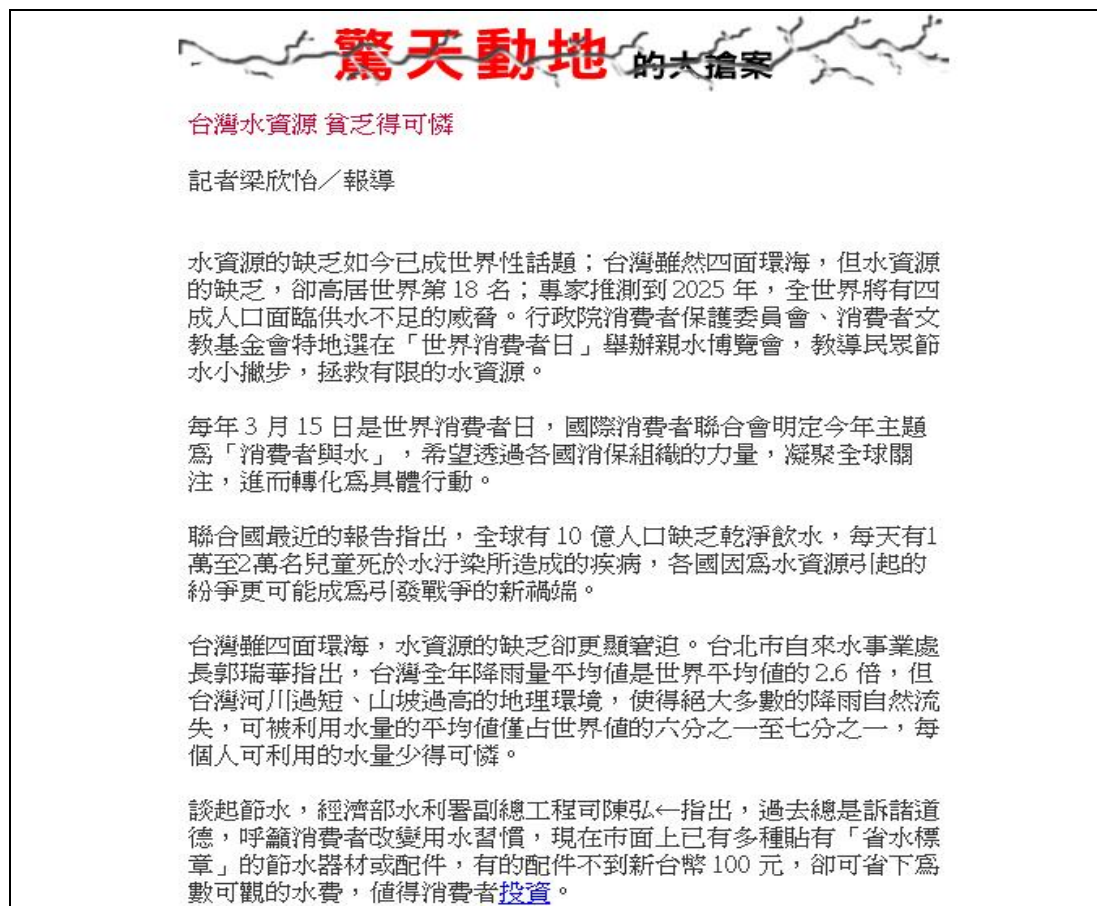


Figure 7.9 The contents of the HTML attachment.



Figure 7.10 A successfully verified share.

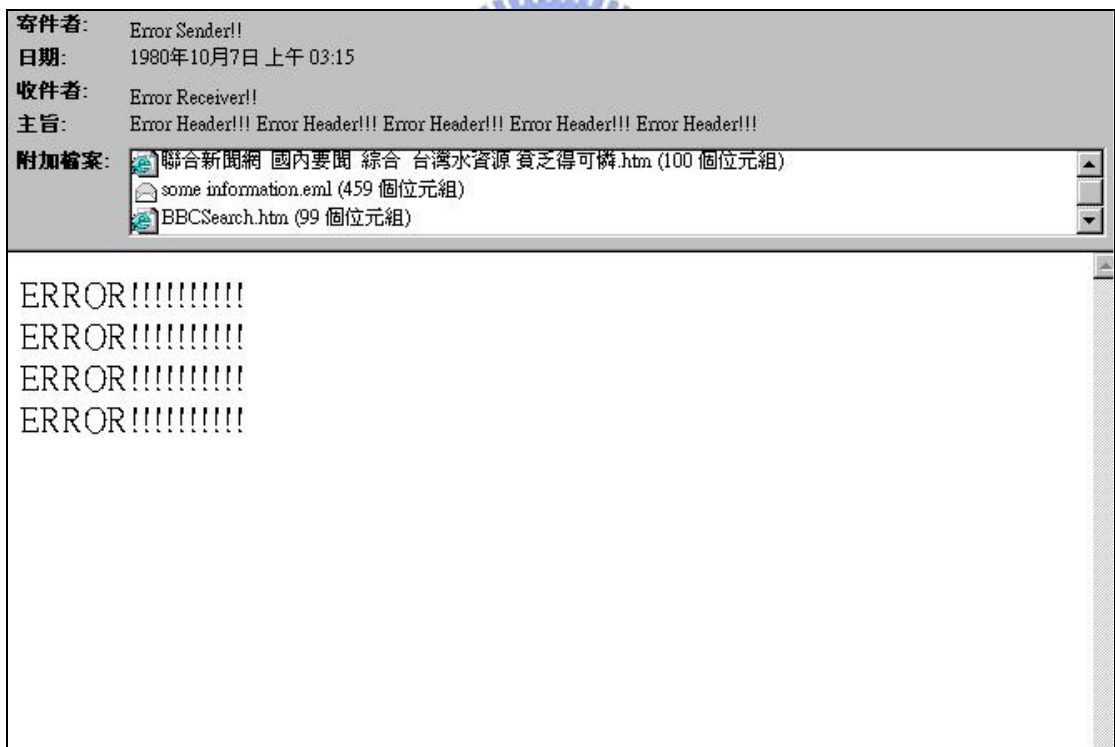


Figure 7.11 A verified share with an improper key.

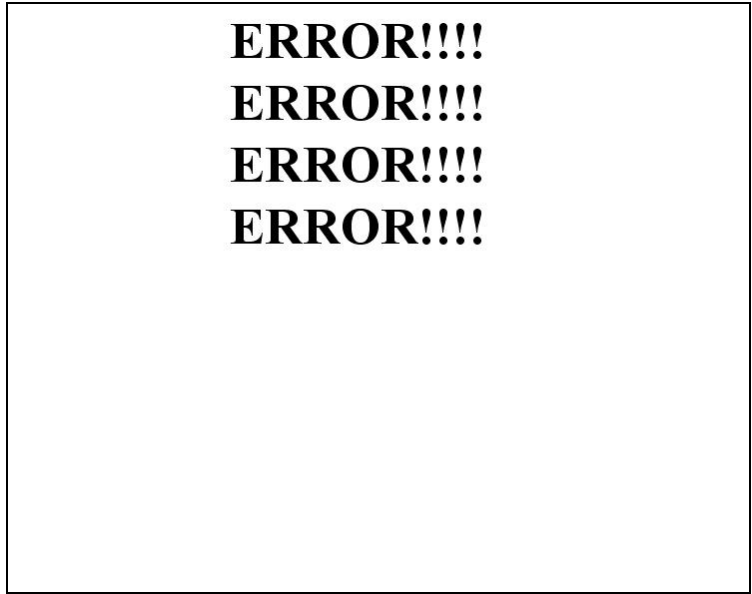


Figure 7.12 The contents of the HTML attachment in the verified share in Figure 7.11.

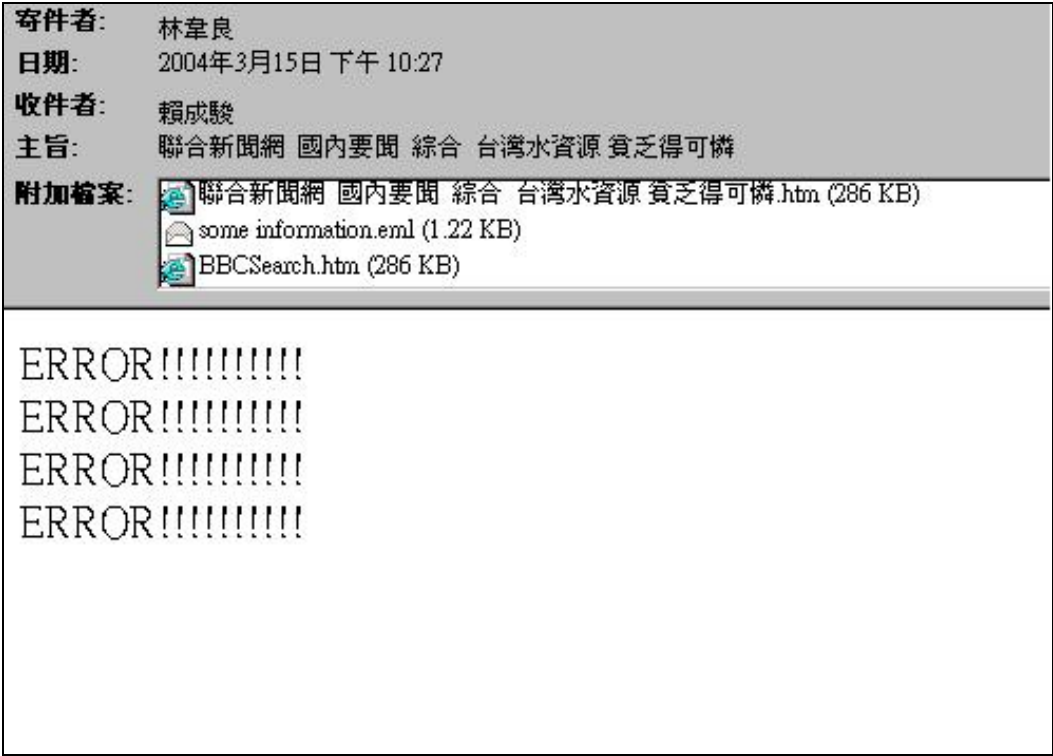


Figure 7.13 The verified share of the share in Figure 7.8, which is tampered.

Chapter 8

Conclusions and Suggestions for Future Works

Conclusions

In this study, we have proposed three information sharing methods for text-type documents of three different formats, pure text, HTML, and e-mail as well as three steganographic methods for covering the sharing result. Three authentication methods are also proposed to create secure sharing scenarios.

For pure text documents, a method for sharing secret information by exclusive-OR operations is applied to a secret text and several articles are chosen from an article database to generate a piece of meaningless data which is then translated into simple sentences to generate a meaningful article by a steganographic method. The articles chosen from the database and the translation result are collectively regarded as stego-texts of the secret text. Through the authentication method by inter-word, inter-sentence, and inter-line spacing for the stego-texts, the fidelity and integrity of each share can be verified. In brief, we encode and distribute a secret text into several shares, which are meaningful articles and can be authenticated.

For HTML documents, by the method of cooperative sharing with data magnitude control, the visible contents, e.g., text contents and multimedia contents, of a secret HTML document are encoded and distributed into many pieces of share data. For text components, the share data become another text component after applying the steganographic method that replaces the original text component with an article, from a database. The share data are embedded in between-word spaces in the article. As for

multimedia components, fake but workable multimedia components are generated with the corresponding share data embedded in. The resulting HTML shares are of the style of the secret HTML. In other words, the layout of the components of an HTML share on browsers is the same as that of the secret HTML. The authentication signals of all the multimedia components are equally embedded in the letter cases of tag names and attribute titles inside tags of a share HTML and the authentication signals of the text component are also embedded in the between-word spaces by controlling the number of authentication signals in each space. Therefore, a secret HTML can be encoded and distributed into several HTML shares which can be authenticated.

For e-mails, a secret e-mail is encoded and distributed into several authenticable e-mail shares by hierarchical sharing with data magnitude control, steganographic methods, and authentication techniques. The technique of data magnitude control makes it possible to apply hierarchical sharing to a string of data. By extracting and sharing the important information of the components in the secret e-mail, pieces of share data are generated. Then, stego components are created by steganographic methods for the share data. For e-mail header components, the share data are embedded in a generated e-mail as the values of some self-defined attributes. For the other components, which contain file data each, the share data are translated into pure texts of simple sentences or embedded in the between-word spaces, which are outside tags, of HTML files according to the type of file data. In the end, each share retains the framework of the secret e-mail and contains the components with the share data of the corresponding secret component embedded in. For an e-mail header component in a share, the authentication signals of the share data of the component are concatenated at the rear of the expressions of the e-mail header. For a component with an HTML file inside, the authentication signals of the share data embedded in the HTML are hidden in the letter cases of tag names and attribute titles inside the tags of the HTML.

As for a component containing a pure text, the authentication technique used for pure text shares can be applied again to justify its fidelity.

Suggestions for Future Works

Several suggestions for future research topics are listed in the following.

1. Improvement on the security of steganographic effect creation for the share data of pure texts.
2. More testing on different operation systems and browsers for practical applications.
3. Information sharing techniques for PDF and Flash files.
4. Steganography techniques for PDF and Flash files.
5. Authentication techniques for PDF and Flash files.
6. Efficient secret sharing, steganographic and authentication techniques for real-time applications.
7. New applications of information sharing, steganography, authentication techniques.



REFERENCES

- [1] A. Shamir, "How to share a secret," *Communications of the Association for Computing Machinery*, vol. 22, no. 11, pp 612-613, 1979.
- [2] Y. Desmedt and Y. Frankel, "Threshold cryptosystem," *Advances in Cryptology --- CRYPTO'89*, pp 307-315, 1989.
- [3] T. P. Pedersen, "A threshold cryptosystem without a trusted party," *Advances in Cryptology --- EUROCRYPT'91*, pp 522-526, 1991.
- [4] C. S. Lai and L. Harn, "Generalized threshold cryptosystems," *Advances in Cryptology --- ASIACRYPT'91*, pp 159-169, 1991.
- [5] D. R. Stinson, "An explication of secret sharing schemes," *Designs, Codes, and Cryptography*, vol. 2, pp 357-390, 1992.
- [6] C. C. Chang and H. C. Lee, "A new generalized group-oriented cryptoscheme without trusted centers," *IEEE Journal on Selected Areas in Communications*, vol. 11, no. 5, pp 725-729, 1993.
- [7] H. M. Sun and S. P. Shieh, "Construction of dynamic threshold schemes," *Electronics Letters*, vol. 30, no 24, pp. 2023-2024, 1994.
- [8] C. H. Cooke, "An integer Optimization Problem with mixed algebraic and number-theoretic constraints," *Applied Mathematics Letters*, vol. 16, pp 635-638, 2003.
- [9] Mark Hillery et al., "Quantum secret sharing," *Physical Review*, vol. 59, no. 3, pp 1829-1834, 1999.
- [10] Marcus Greferath and Stefan E. Schmidt, "Secret sharing on partially ordered sets," *1998 Proceedings. 1998 IEEE International Symposium on*, p 299, 16- 21 Aug., 1998.
- [11] Amos Beimel and Yuval Ishai, "On the power of nonlinear secret-sharing," *16th annual IEEE Conference on 2001-Computational Complexity*, pp 188-202, 2001.
- [12] C.-S Lai and Y.-C Lee, "V-fairness (t, n) secret sharing scheme," *IEE Proceedings-Computers and Digital Techniques*, vol. 144, issue 4, pp 245-248, July 1997.
- [13] Okamoto, E., "Cryptosystems based on polynomials over finite fields," *Information Theory Workshop, 2002. Proceedings of the 2002 IEEE*, pp 74-77, 20-25 Oct. 2002.
- [14] C. C. Chang, C. S. Tsai, and T. S. Chen, "A new scheme for sharing secret color images in computer network," *Proceedings of the Seventh International*

- Conference on Parallel and Distributed Systems*, pp 4-7, 2000.
- [15] Chih-Ching Thien, Ja-Chen Lin, "Secret image sharing," *Computers & Graphics*, vol. 26, pp 765-770, 2002.
- [16] Chin-Chan Chang and Tai-Xing Yu, "Sharing a secret gray image in multiple images," *Cyber World, 2002. Proceedings. First International Symposium on*, pp 230-237, Nov. 2002.
- [17] Thien, C.-C. and Ja-Chen Lin, "An image-sharing method with user-friendly shadow images," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 13, issue 12, pp 1161-1169, Dec 2003.
- [18] C. C. Lin and W. H. Tsai, "Secret image sharing with steganography and authentication," to appear in *Journal of Systems & Software*
- [19] C. C. Lin and W. H. Tsai, "Secret multimedia information sharing with data hiding capability by simple logic operations," to appear in *Pattern Recognition and Image Analysis*.
- [20] C. C. Lin, "A study on secret sharing with steganographic effects," *Doctor Thesis, Department of Computer and Information Science, National Chiao Tung University, Taiwan, Republic of China*, 2003.
- [21] P. Wayner, "Strong theoretical steganography," *Cryptologia*, vol. XIX/3, pp 285-299, 1995.
- [22] Low, S. H., N. F. Maxemchuk and A. M. Lapone, "Document identification for copyright protection using centroid detection," *IEEE Transactions on Communication*, vol. 46, no. 3, pp 372-383, 1998.
- [23] Brassil, J., N. F. Maxemchuk and L. O'Gorman, "Electronic marking and identification techniques to discourage document copying," in *Proceedings of INFOCOM'94*, Toronto, Ont., Canada, pp 1278-1287, 1994.
- [24] W. Bender, et al., "Techniques for data hiding," *IBM System Journal*, vol. 35, no 3&4, Feb 1996.
- [25] Y. H. Chang and W. H. Tsai, "A steganographic method for copyright protection of HTML documents," *Proc. of 2003 National Computer Symposium*, Taichung, Taiwan, Dec. 2003.