THESIS

Rights Protection of Digital Content in the DRM Environment

submitted by:

Antonius CAHYA PRIHANDOKO

under supervision:

Associate Professor Bruce LITOW

Dr Hossein GHODOSI

for the degree of Doctor of Philosophy

at the Discipline of Information Technology

College of Business, Law and Governance

JAMES COOK UNIVERSITY

2015

Keywords

Anonymous cash, Blind decryption, Chaotic maps, Content provider's security, Digital rights management, Digital watermarking, Oblivious content distribution, Traitor deterring mechanism, Traitor tracing schemes, Users' privacy, White-box implementation.

Statement of the Contribution of Others

1. Editorial assistance from a professional editor is used in preparing the final thesis. I have read and complied with the Guidelines for the Editing of Research Theses by Professional Editors. Professional editors, Alan and Jean Dartnall, provided copy editing and proofreading services, according to the guidelines laid out in the university-endorsed national Guidelines for editing research theses. This editorial work was funded by School of Business, James Cook University.

2. All published papers within my candidature were prepared under supervision Dr Hossein Ghodosi and Associate Professor Bruce Litow.

Antonius Cahya Prihandoko

March 2015.

Declaration

I declare that

1. I have stated clearly and fully in the thesis the extent of any collaboration with others. To the best of my knowledge and belief, the thesis contains no material previously published or written by any other person except where due reference and acknowledgement has been made.

2. Every reasonable effort has been made to gain permission and acknowledge the owners of copyright material. I would be pleased to hear from any copyright owner who has been omitted or incorrectly acknowledged.

3. This thesis is my own work and has not been submitted in any form for another degree or diploma at any university or other institution of tertiary education. Information derived from the published or unpublished work of others has been acknowledged in the text and a list of references is given.

Antonius Cahya Prihandoko

March 2015.

Acknowledgments

Praise to the Lord Jesus Christ for He has been a constant source of strength and help throughout my life, including study and this project. All I have and reach are because of His mercy.

The Directorate General of Higher Education of Republic of Indonesia, deserves much appreciation for their generous scholarship and other non-material assistance. Without them, I would not be here.

I gratefully acknowledge the assistance of my supervisors, Associate Professor Bruce Litow and Dr Hossein Ghodosi, and my research student monitor, Dr Steve Campbell, during the conduct of this research and for their careful reading of this thesis during preparation. Sincere thanks must go to all staffs of the Discipline of Information Technology, the College of Business, Law and Governance (School of Business), the Graduate Research Study, the Teaching and Learning Development, and the International Office of the James Cook University for their warmest treatment during my study at JCU. Special thanks to Dr Liz Tynan and Ms Kellie Johns who generously conducted supporting programs to improve my writing and research skills.

I would like to thank also some experts. Dr Renato Ianella from W3C ODRL Community Group, Dr Ghita Stefan and Professor Victor Patriciu from Military Technical Academy, Bucharest, Romania, for their explanation of Digital Rights Management. Dr Brecht Wyseur from Nagra, Belgium, for his valuable insight into white-box cryptography. Professor Ingemar J. Cox from University College London,

Dr Alessandro Piva from University of Florence, Italy, and Dr Jeffrey Bloom, for useful information about digital watermarking for copy protection. Dr Jae Park from University of Texas at San Antonio, for his explanation of the concept of usage control. Dr Jeff Lotspiech from IBM for his valuable clarification of his traitor tracing scheme.

Some images used in Chapter 5 are taken from the University of Southern California - Signal and Image Processing Institute (USC-SIPI) Image Database (sipi.usc.edu/database). The USC-SIPI provide this image database for free to support research in image processing, image analysis and machine vision.

Much appreciation goes to my parents, Bapak J.B. Mulyono and Ibu M. Sri Mulyati, my parents-in-law, Bapak J.B. Elias and Ibu M.V. Roesmi, and all relatives. Their great support has a special place in my heart. I owe them a debt that can never be repaid.

Finally, and most importantly, I dedicate this thesis to my dearest wife Juliana Ririet Ellyperwati, my lovely son, Stanislaus Jiwandana Pinasthika, and my lovely daughter, Beatrix Jayanimittakirana Anindyakencana, who always encouraged and inspired me while I was studying here. They gave me the spirit and support while they were away, and brought colorful days when they were beside me. It showed me how important they are in my life. Forever.

Antonius Cahya Prihandoko

March 2015.

Abstract

In electronic-business settings, content providers produce digital goods/services (such as, games, images, softwares, etc.) which consumers/users wish to purchase. In general, mass-production of digital goods/services is possible if an instance of the goods/services is made available. Digital Rights Management (DRM) mainly considers technological approaches to protect content providers' rights on their original products (also known as, intellectual property) against illegal reproduction of the goods/services (also known as, piracy). Due to the wide range and different types of digital goods, several DRM systems have been studied in the literature. Many existing DRM systems, however, focus on the security of the content provider and often neglect the users' privacy.

The problem statement of this thesis is devising DRM systems that protect users' privacy while content provider's security is maintained. To achieve this goal during the life-time of the digital content, we distinguish three phases/stages.

1. *At the time of purchase* – At this stage, user issues a request for purchasing, and payment is processed in accordance with the content provider's procedure. Protecting users' privacy is mainly achieved by minimising personal data acquisition. To construct a DRM system with this characteristic, I employ two mechanisms, namely *anonymous cash* and *blind decryption*. Anonymous cash allows anonymity of the user (similar to paying cash in daily shopping, which makes it impossible to trace the customer), while blind decryption is used to hide the identity of the purchased item (e.g., one may buy two items from

video-shop without revealing their names to the shop-keeper). In this thesis, both schemes are improved to preserve security and privacy when price variation and buyer authorisation are applied in the content transaction. The improvements are discussed in Chapter 3.

2. *At the time of content delivery* – In daily shopping, customers possess purchased items, i.e., delivery is done directly and there is no doubt that the customer has received the goods. In electronic-business settings, however, the goods are delivered via electronic devices/channels. To make sure that unauthorised users cannot intercept the line and obtain a copy (which is identical to the original) of the goods, a common practice is to encrypt the content. Note that an unauthorised user still can intercept the line and obtain a copy of the encrypted material, but utilisation of the goods/services implies decryption of the material which is believed to be impossible by unauthorised users, while legitimate users who receive the key can decrypt the content and use it. Securing content delivery relies on the secrecy of the cryptographic keys. In this thesis, I employ two approaches, namely *obfuscation* and *obliviousness*, and present two schemes for securing content delivery in DRM systems. Each of these schemes has been presented at, and appeared in, the Proceedings of the International Conference on Computer Science and Information Technology (CSIT) 2013 and the Information System International Conference (ISICO) 2013, respectively. Their details are presented in Chapter 4.

3. *At the time of usage* – After the goods/services are delivered, decrypted, and made available to the customer/user, a dishonest/malicious user may wish to make profit by reproducing (e.g., simply generating a copy of it) and selling to others. This phase/stage of the system is the longest and more complicated. DRM systems have introduced several techniques (e.g., copyrights, watermarking, traitor tracing, etc.) to protect content providers and prevent piracy. In

this project, I study these techniques and improve some of the existing techniques. First of all, I study digital watermarking that is viewed as a potential tool for preserving protection of digital content. I employ chaotic maps to construct some blind watermarking schemes and achieve perfect security of watermark. These schemes are presented in Chapter 5. One of the schemes has been presented at the IEEE International Conference on Information Technology and Applications (ICITA) 2014. With the aid of digital watermarking and motivated by some traitor tracing schemes, I construct a traitor deterring scheme. This mechanism prevents users from constructing a pirate decoder and pirate content. This scheme was presented at the IEEE International Conference on Communication, Networks and Satellite (COMNETSAT) 2013. The detail of this scheme is presented in Chapter 6.

Contents

List of Figures

List of Tables

Statement on Access to this Thesis

I, the undersigned, the author of this thesis, understand that James Cook University of North Queensland will make it available for use within the University Library and, by microfilm or other photographic means, allow access to users in other approved libraries.

All users consulting this thesis will have to sign the following statement:

In consulting this thesis, I agree not to copy or closely paraphrase it, in whole on in part, without written consent of the author; and to make proper written acknowledgment for any assistance I have obtained from it.

Beyond this, I do not wish to place any restrictions on this thesis.

Antonius Cahya Prihandoko

March 2015.

Previously Published Materials

The following papers have been presented at refereed conferences and published in the corresponding proceedings or journals, and contain material based on the content of this thesis.

1. A.C. Prihandoko, B. Litow, and H. Ghodosi, "DRM's Rights Protection Capability: A Review", in *Proceedings of the First International Conference on Computational Science and Information Management (ICOCSIM)*, B.B. Nasution, M. Zarlis, J.M. Zain, R.W. Sembiring, and T. Herawan, Eds., vol. 1. Medan, Indonesia: University Malaysia Pahang Press, December 2012, pp. 12-17. ISBN 978-967-0120-60-7.

2. A.C. Prihandoko, H. Ghodosi, and B. Litow, "Obfuscation and WBC: Endeavour for Securing Encryption in the DRM Context", in *Proceedings of the International Conference on Computer Science and Information Technology*, T. Herawan, N. Ahmad, and P. Vitasari, Eds., vol. CSIT 2013. Yogyakarta, Indonesia: Universitas Ahmad Dahlan Press, June 2013, pp. 150-155. ISBN 978-979-3812-20-5 2.

3. A.C. Prihandoko, H. Ghodosi, and B. Litow, "Secure and Private Content Distribution in the DRM Environment", in *Proceedings of the Information System International Conference*, A. Herdiyanti, M.P. Syafitri, and R.V. Margareta, Eds., vol. ISICO 2013. Bali, Indonesia: Department of Information System, ITS, Surabaya, December 2013, pp. 659-664. ISBN 978-979-18985-7-7.

Available online at Open Access Journal of Information Systems (http://is.its.ac.id/pubs/oajis/index.php/home/detail/1156/ Secure-and-Private-Content-Distribution-in-the-DRM-Environment)

4. A.C. Prihandoko, H. Ghodosi, and B. Litow, "Deterring Traitor Using Double Encryption Scheme", in *Proceedings of the IEEE International Conference on Communication, Network and Satellite*, vol. COMNETSAT 2013. Yogyakarta, Indonesia: IEEE, December 2013, pp. 100-104. ISBN 978-1-4673-6054-8. Available online at IEEE Xplore Digital Library. INSPEC Accession Number: 14515736; DOI: 10.1109/COMNETSAT.2013.6870869. (http://ieeexplore.ieee.org/xpl/articleDetails.jsp?arnumber=6870869)

5. A.C. Prihandoko, H. Ghodosi, and B. Litow, "Blind Image Watermarking Based on Chaotic Maps", presented at *the IEEE International Conference on Information Technology and Applications*. ICITA 2014. Sydney, Australia, July 2014. ISBN 978-0-9803267-6-5. Available online at *IT in Industry Journal*, vol. 2, no. 2, 2014, pp. 44-50. ISSN 2203-1731 (http://www.it-in-industry.org/index.php/itii/article/view/27)

Chapter 1

Introduction

We live in a digital world. All types of content — image, video, audio and text — can be represented in digital forms. This representation improves efficiency and accuracy in copying, storing and distributing content. Additionally, wide spread of broadband Internet has made digital content available worldwide and accessible in various platforms. This technology enables end-users to consume content much more easily. The end-users are able to watch movies, read electronic newsletters or books, listen to digital music, enjoy valuable photographs or learn history without physically going to cinemas, libraries, art galleries or museums. As a result, e-business on digital content has attracted great attention.

In e-business settings, electronic content distribution is used as a main tool for services. This content distribution technology provides a new prospective market in the Internet era. More digital content industries have been earning benefits through this technology from year to year. The 2011 International Federation of the Phonographic Industry (IFPI) digital music report, for example, indicated that the amount of music industry revenue from digital channels has increased from 2% in 2004 to 29% in 2010 [87]. These data indicate that electronic distribution has become a preferred method for disseminating multimedia content.

Table 1.1: Total piracy rates and amount of losses worldwide [6, 7, 109]

Year	Piracy rates (%)	Losses (in $\$M$)
2003	36	28,803
2004	35	32,778
2005	35	34,482
2006	35	39,698
2007	38	47,809
2008	41	52,998
2009	43	51,443
2010	42	58,754
2011	42	63,456
2013	43	62,700

1.1 Copyright Violation

The efficiency of communicating content through digital media is unquestionable, but there are challenges. Unexpected effects of this technology include increased opportunities for copyright violations. Downloading content from web sites without paying is easily done worldwide. While the growth of broadband Internet connection around the world is generating many profits, this phenomenon has also made the Internet an enormously efficient medium for distributing copyright-infringing content.

Digital piracy, in fact, has been growing globally and has caused a significant drop in industry revenue. For example, despite the rise of the digital music market from 2004 to 2010, global music revenue was down 30% due to piracy [87]. Significant lost revenues were also suffered by movie industries. The Motion Picture Association of America (MPAA) studios lost USD 6.1 billion to piracy in 2005 [175]. The same phenomenon also occurs in the software industry. Studies conducted by Business Software Alliance (BSA) in some countries in Asia Pacific, Middle East, Africa, Europe, Latin and North America in 2007 [6] and 2012 [7] revealed that while the software piracy rate increased about 15 %, losses caused by this piracy were doubled within less than a decade (see Table 1.1).

The advance of the Internet and digital content technologies have created chal-

lenges for developing and establishing the regulations that protect and enforce intellectual property rights (IPR) in many countries. In 1998, the US Senate released the Digital Millennium Copyright Act (DMCA) that criminalised the production and dissemination of technology that enables users to circumvent copy-restriction systems [133]. As with DMCA, the European Union Copyright Directive (EUCD) gives new protection to technological measures systems that restrict the use of literary and other digital works [27].

The progress of IPR protection and enforcement varies among the countries. Several countries outside the United States have made significant positive progress on IPR protection and enforcement including Czech Republic, Hungary, Poland, Fiji, Israel, Saudi Arabia, Sweden and European Union, whereas some other countries, such as Brazil, Canada, China, India, Russia, Spain and Ukraine still have significant concerns about the Internet based piracy [64, 90, 100, 193, 203]. Although Internet piracy is rapidly superseding physical piracy in many markets around the world, the production and trade of pirated optical disc remain a major problem in some countries, such as Malaysia, the Philippines, Romania and Indonesia [9, 100]. The courts of many countries have struggled with the novelty of digital media and the standardisation of the copyright material protection [107, 172]. Briefly, cybercrime violating digital content has become a great concern and major threat for e-business worldwide.

1.2 Digital Rights Management

Digital Rights Management (DRM) emerged in the early 1990s as a concrete response to increasing threat in the digital content industries. This system provides a technological approach that enables content providers to properly manage the use and distribution of their intellectual property [3]. The technical and managerial perspectives of DRM have been emphasized primarily on a preventive approach and

content protection [206]. Focusing on content protection, however, makes DRM fails in maintaining consumers' interest [148]. Consumer's privacy is often sacrificed when DRM is implemented. The decline of privacy caused by the DRM implementation becomes a critical problem and a main reason behind the suspicion of users towards DRM.

Foremost among technologies applied in the DRM systems are cryptography and digital watermarking. Cryptography is used to secure content distribution using encryption and signature mechanisms. An encryption algorithm could be symmetric or asymmetric. One big issue with a symmetric algorithm is the key exchange problem. This problem can be resolved by an asymmetric algorithm. However, in terms of complexity, an asymmetric algorithm is less efficient than a symmetric one [177]. In practice, a hybrid system would be suitable, whereby a symmetric algorithm is used to encrypt and decrypt messages, and an asymmetric algorithm is used to communicate the key. The asymmetric systems are also mainly used in the digital signature schemes. In these schemes, the private key is used for signing the message and the public key for verifying the signature.

Digital watermarking provides another security solution by embedding marks into content to preserve protection of it. In the electronic commerce applications, each type of media data requires a specific watermark according to watermark design patterns (WDPs). Watermark for audio and video data has to be imperceptible and robust; watermark for text data has to be imperceptible and fragile; and watermark for image data can be either perceptible and fragile, or imperceptible and robust [104]. These watermark specifications are provided to meet customers satisfaction and content security.

1.3 Aims and Objectives

This research aims at investigating the implementation of cryptography and watermarking technologies to improve DRM for content distribution systems. The objective of the investigation is to develop some protection schemes, including protecting users' privacy at the time of transaction, securing content delivery and preserving protection after content delivery. Each of these schemes is projected to equally achieve content provider's security and users' privacy, i.e., while the users' privacy is protected, the content provider's security is still maintained. Achieving the goals of the research, we will find answers to these questions:

1. What are the current state and major problems for the DRM system?
2. How can DRM provide balanced protection for content provider's security and users' privacy in a content distribution system?
   (a) How is it possible to protect users' privacy while it still maintains content provider's security?
   (b) How is it possible to secure content delivery while it still preserves users' privacy?
   (c) How is it possible to preserve protection after content has been delivered (at the time of content usage)?

1.4 Scope

The term "rights" in this research refers to the rights of both content provider and users in a content distribution system. On one side, the content providers need to securely manage the distribution and use of their protected content. On the other side, the users want their privacy to be protected by the system, so that they can privately access the content. In a content distribution system, these rights need to be balanced protected.

Researches in this thesis focus on constructing some cryptography and watermarking based protection schemes that can be adopted by a DRM system to provide balanced protection for content provider's security and users' privacy in a content distribution system. The content provider's security means that only authorised users can access the protected content and use it properly. The user's privacy protection refers to a condition when the user's identity is not connected to the content he purchased or accessed.

The protection schemes proposed in this thesis can be integrated or adopted independently. Since each proposed scheme is designed to achieve both security and privacy aspects, the implementation of the schemes supports DRM to provide balanced protection for both content provider and users.

1.5 Achievements and Contributions

The major outcome of this research is the development of some protection schemes that provide balanced consideration for content provider's security and users' privacy in a content distribution system. The schemes can be distinguished in three stages — at the time of content purchase, content delivery and content usage.

1. Protecting users' privacy at the time of content purchase;

   (a) Improved blind decryption scheme to overcome varying prices and buyer authorisation.

   Blind decryption allows content provider to identify a user, but not the items purchased by the user. The improved scheme is intended to preserve content provider's security and users' privacy in cases when varying price and buyer authorisation are implemented in the system.

   (b) Improved anonymous cash scheme to overcome buyer authorisation.

   Anonymous cash allows users to purchase content anonymously, while content provider is still able to identify which content is being purchased.

The improved scheme is intended to anticipate whether buyer authorisation is applied in the system.

2. Securing content delivery;

   (a) White-box implementation based content distribution scheme.

   In this scheme, the content key is replaced by a composition of internal and external keys. The external keys are stored inside the inaccessible area of the smart card's memory. The content can only be decrypted using the corresponding smart card that can be purchased anonymously. Thus, the content and its associated smart card will not be connected to the user's identity.

   (b) Oblivious content distribution scheme.

   This scheme allows content provider to deliver contents to user in such a way that at the end of the scheme the user cannot access contents beyond what he is supposed to access and the content provider will not know which contents are accessed by the user.

3. Preserving protection at the time of content usage (after content delivered);

   (a) Secure blind watermarking schemes based on chaotic maps.

   In this watermarking scheme, chaotic maps are used to secure the embedded watermark. However, the watermark's robustness is also considered in the constructed schemes.

   (b) Double encryption scheme to deter piracy.

   This scheme is intended to deter pirate decoder and pirate content. To deter a pirate decoder, content is encrypted and the system gives each user a unique traceable personal key to decrypt content. If a pirate decoder is discovered, the key used by the decoder can be identified. To deter pirate content, the system utilises watermarking and encryption to construct

and allocate content version for each user. If a pirate copy of content is found, its version can be identified from the watermarks extracted from its decrypted segments.

Some kinds of media for which the proposed schemes are applicable are explained in subsection 7.2. Some of these schemes have been published as conference papers and are listed in the *Previously Published Materials* preliminary section. There are five papers that have been presented in refereed International conferences and appeared in the corresponding proceedings or journals. The contents of these publications are discussed in subsequent chapters of this thesis.

1.6 Thesis Outline

The remaining parts of this thesis are organised as follows.

Chapter 2 contains a literature review on Digital Rights Management (DRM). Beginning with a brief description of the DRM systems and the usage control perspective, the review continues with the current state and major problems for the DRM system. The review is then concluded by a discussion on improving the DRM system in terms of its technological approaches.

Chapter 3 presents a study on protecting users' privacy at the content transaction time. At this stage, while the system allows users to maintain their privacy, it must prevent any malicious action that disadvantages content provider's security. The study focuses on two schemes — anonymous cash and blind decryption — and improves these schemes to overcome some conditions that might be applied in a content distribution system, such as price variation and user authorisation.

Chapter 4 provides a study on securing content delivery. The question of the study is how it is possible for the system to secure content delivery, while it still preserves users' privacy. Securing content delivery relies on the secrecy of the decryption keys. Two approaches for achieving the secrecy include obfuscation and

obliviousness. Each approach is used to construct a content distribution scheme. The schemes protect the decryption key in the different ways, but their implementations ultimately preserve both security and privacy.

Chapter 5 describes a study on digital watermarking that is viewed as a potential technology to preserve content protection. The study focuses on utilising chaotic maps to construct watermarking schemes. Experimental results exhibit that the constructed watermarking schemes achieve a perfect watermark security. Additionally, the watermarking scheme which is undertaken in the frequency domain is more robust than those in the spatial domain.

Chapter 6 provides another study on preserving protection. Watermarking alone might be less effective to preserve protection. It must be integrated with a protection system, such as traitor tracing schemes — a copyright violation detection system which works by tracing the source of leaked information. Motivated by some traitor tracing schemes, a traitor deterring mechanism — the mechanism that can be used to discourage users from constructing pirate decoder and pirate content — is constructed.

Chapter 7 concludes this thesis with a synopsis on how DRM system can provide balanced protection for content provider and users in a content distribution system. Additionally, some applications of the constructed schemes are also presented. This chapter is finally closed by a description of limitations and open problems.

Chapter 2

DRM Overview

2.1 Introduction

Digital Rights Management (DRM) refers to any type of technology that allows content providers to protect their intellectual property. The technology is used to identify the rights holder, assign permissions and control usage tracks. DRM has two basic tasks — *rights management* and *rights enforcement* [141]. The former enables authorised proprietors to recognise their content, declare the rights and provide distribution models, and the latter allows authorised proprietors to implement their rights and content usage rules.

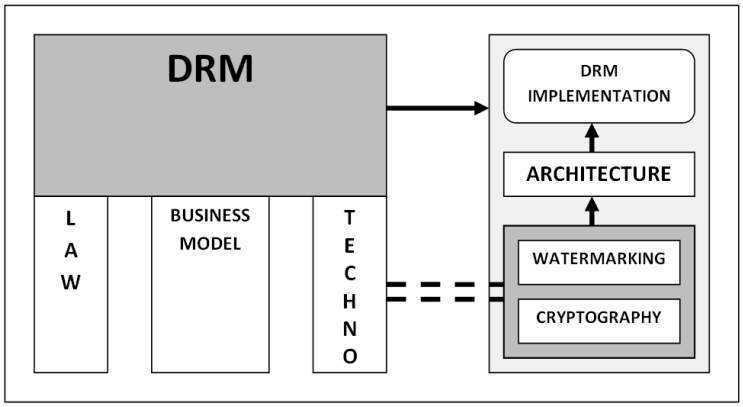


Figure 2.1: *DRM structure and implementation framework*

The DRM system is built upon three pillars — law, business models and tech-

nology [15, 164] (see Figure 2.1).

- Law is required as a legal foundation for assigning permissions, restrictions and obligations to access content.

- Business models are the basis for the establishment of protocols that are used to manage content usage.

- Technology that is mainly based on cryptography and digital watermarking is utilised to support the implementation of the protocols.

Because of these pillars, DRM becomes an umbrella term for research of multiple scientific disciplines, such as information security, copyright law and technological implementation [163], as well as business application of the digital content industry, including financial, business model and pricing policies [106].

To provide persistent protection on digital content, DRM must satisfy the following requirements [73].

- Interoperability — DRM must be able to deal with usage rights for different kinds of digital content, such as sound, video, image or text, and for different platforms, such as desktop, laptop, Personal Digital Assistances, mobile phone, and so on.

- Security — DRM is required to be sufficiently robust to withstand any attack that circumvents the protection rule and enables unauthorised access to content.

- Privacy — Rights holders demand that DRM also secures access to their data, providing a proper classification and protection of numerous kinds of data held on the system. Although DRM must work simultaneously with the e-commerce system that may store customers' data, the presence of DRM must preserve users' privacy.

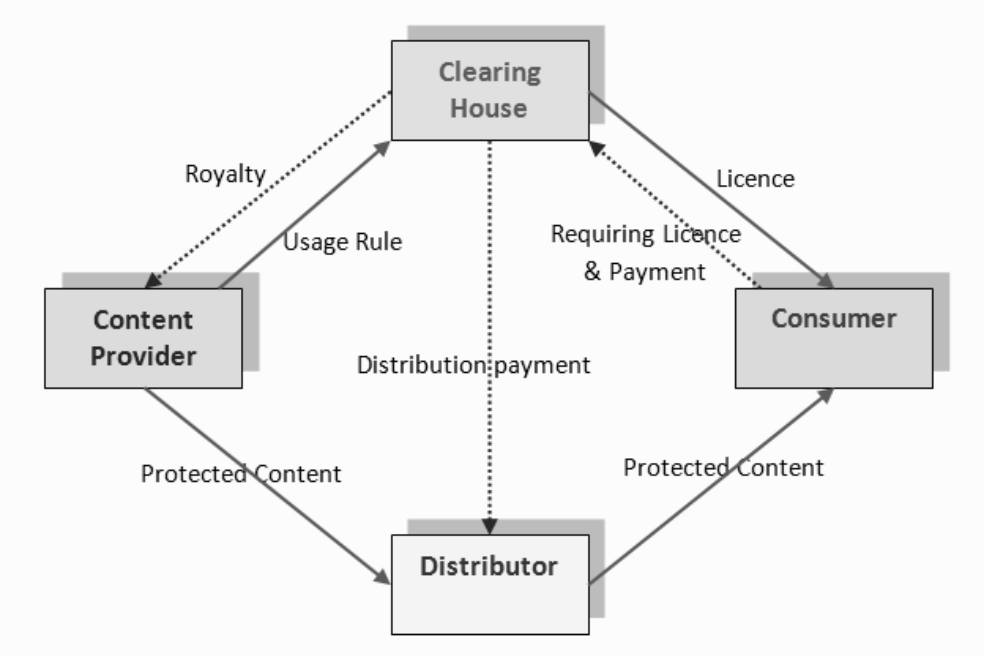


Figure 2.2: *Typical DRM system for content distribution [108]*

2.2 DRM Framework

A typical DRM framework for content distribution (see Figure 2.2) consists of four parties — content provider, distributor, clearinghouse and consumer (user) [108]. First of all, content provider encrypts content for security purposes and then passes the protected content to the distributor and corresponding usage rules to the clearinghouse. The distributor makes the protected content available on a web server that enables users to download it. The clearinghouse is responsible for the financial transaction, issuing license to the consumer and paying royalty to the content provider. A consumer then retrieves the content through the distribution channel and requests a licence from the clearinghouse. The consumer has to register his profile, provide detail of the purchased content, and then make a payment. After verifying the submitted data and charging the consumer's account, the clearinghouse releases a license and delivers it to the consumer. The consumer decrypts and uses the content according to the rights described in the license. The clearinghouse then pays royalty to the content provider and distribution fees to the distributor accordingly.

The overall DRM framework suited to construct digital rights-enabled systems

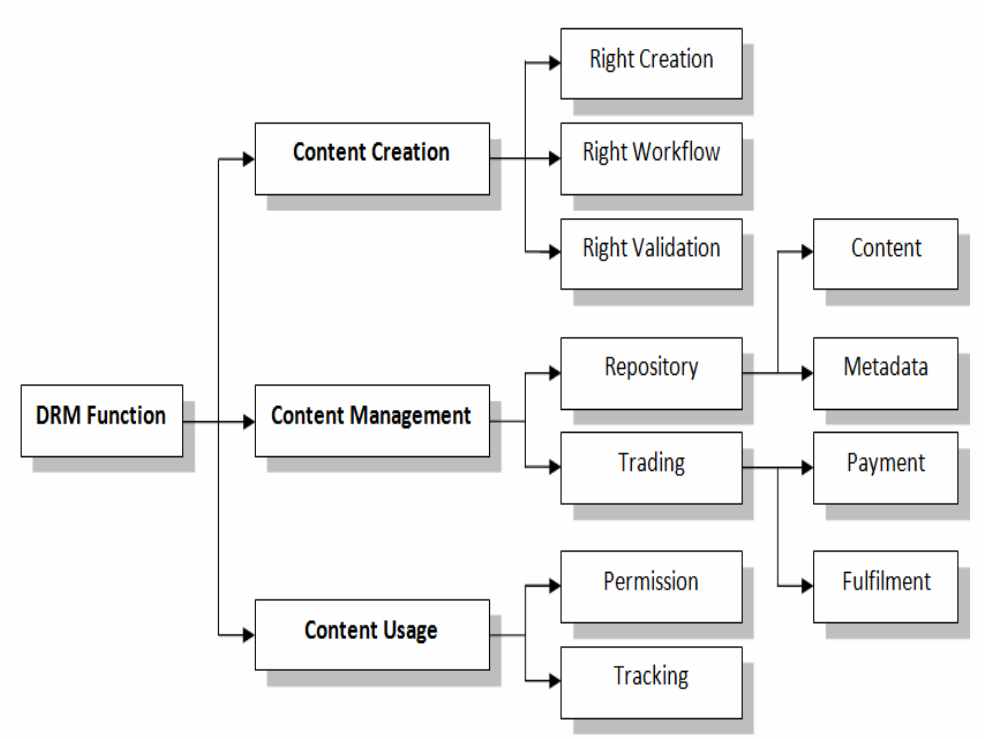


Figure 2.3: *DRM functional architecture [86]*

can be modelled in three stages of content's life cycle — content creation, content distribution management and content usage [86]. The model is complemented by a functional architecture providing the framework to implement DRM functionality (see Figure 2.3). First of all, when content is created, the rights that specify the rights' owners, constraints and usage permissions must be validated and assigned to the content. Subsequently, when the content is copied or distributed, it must be processed through a sequence of review and approval of the rights. In the distribution stage, the system needs to have proper access to the content and metadata. These factors are needed for managing licences assigned to participants involved in a trading agreement, and royalty payments from licensees to rights holders. In a particular condition, the content may also be encrypted or packaged to fulfil the agreement. Finally, after content has been traded, usage flow of content has to be monitored and controlled to enforce the license conditions.

DRM functionality can be viewed hierarchically in four layers [49] (see Figure 2.4).

1. Trust management — This layer is responsible for certificate validations to

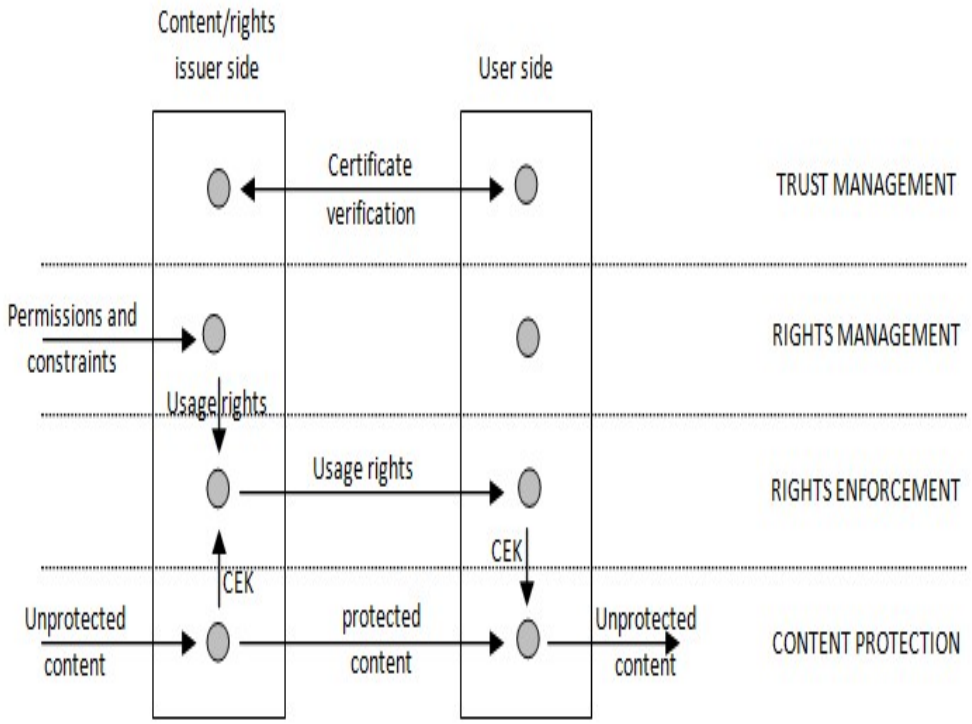


Figure 2.4: *DRM layer [49]*

ensure that only the certified parties interact in the system, i.e., only authenticated issuers are able to create licenses and only authenticated clients are able to retrieve the licenses.

2. Rights management — In this layer, a set of rights and constraints over content is defined by content providers, recorded on the server, and forwarded to the rights enforcement layer.

3. Rights enforcement — This layer has the responsibility to protect usage rights and assure a compliance by the clients.

4. Content protection — This layer is an essential component in a DRM system. In this layer, the content or rights issuer securely seals the content and passes the content encryption key (CEK) to the rights enforcement layer. The client can only access the protected content if the rights enforcement layer forwards the CEK.

This hierarchical view is an abstract DRM system model, however, it provides a

clear and simple view of the DRM system.

2.3 Closed Source DRM Systems

DRM has received great attention from both academia and industry, since it has a potential prospect for commercial sector. Many commercial entities, such as Microsoft, Adobe, Apple, Sony and so on, have developed various technical solutions for DRM implementations. Some existing DRM systems are distinguished into two groups, closed source and open source. Closed source DRM systems are constructed by commercial bodies to protect their own products, while open source systems are provided by a party to be used by another parties. In this section, examples of some closed source DRM systems are presented according to the type of content they protect.

2.3.1 DRM for Videos

- Content Scrambling System

The Content Scrambling System (CSS) is an early example of DRM system [19]. The CSS uses an encryption algorithm to encrypt content on the DVD disc to prevent byte-for-byte copying of a MPEG stream. The encryption key is split and separately stored in the restricted area of DVD and the DVD drive itself. The DVD players manufacturers have to obtain a licence for this technology and apply it to their devices so that they can decrypt and play CSS-protected content. Such devices are then called compliant devices [112]. The CSS was first introduced in 1996, but was compromised in 1999 by an application called DeCSS, which allowed a CSS-encrypted DVD to play on a computer running the Linux operating system [170].

To improve the CSS technology, the DVD copy protection mechanism employed watermarking to reduce the functionality of the copy. There are three possible types of disk — legal encrypted with CSS, illegal with CSS and illegal without CSS — and

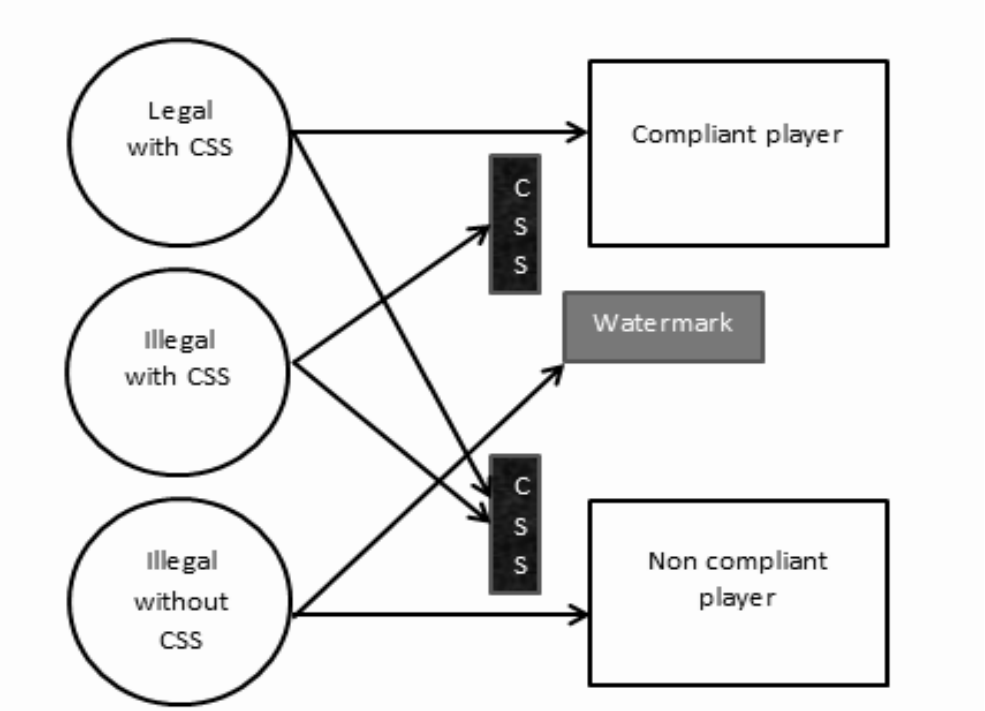


Figure 2.5: *Watermark's role in a DVD copy protection system [118]*

two types of device — compliant and non compliant devices [118] (see Figure 2.5). Both legal and illegal with CSS disks are not playable in a non compliant player as this player has no CSS decryption key. One part of the CSS decryption key is stored on a part of the legal with CSS disk that is readable, but not writable. One cannot get this key onto the copied disk, causing an illegal with CSS disk also cannot be played in a compliant player. Unlike previous types of disks, without watermark, an illegal without CSS disk is not blocked by the CSS protection and is playable in both compliant and non compliant devices. With the aid of watermarking, the watermark will be presented on all protected videos. The logic in the player then can determine that the disk should not be played if it is watermarked, but not encrypted. In this improved technology, however, watermarking does not prevent content from being copied, so that if there is any illegal copy without CSS, this copy cannot be prevented from being played in a non-compliant device.

- Microsoft

Microsoft released the Windows Media DRM (WMDRM) [115] in 1999. The system restricts and controls what the users can do with the content. The WMDRM is a platform to protect and securely deliver content for playback on Windows-based computers, portable devices and network devices [55]. This application is used to read instructions from media files in a rights management language. The instructions stated what the users can do with the media, such as how many times the media file can be played, whether the file can be copied or saved to the local disk, or whether the file can be transferred to other devices. However, the WMDRM system was cracked by Beale Screamer in 2001 [180].

In 2006, Microsoft added a DRM system called the Protected Media Path (PMP) in their Windows Vista [116]. PMP contains the protected video path (PVP) that is used to stop DRM-protected content when an unsigned software is running. This mechanism aims at preventing the unsigned software from accessing the content. PVP can also encrypt information during the transmission to monitor or graphics card, which makes it more difficult to make unauthorised recordings.

- Advanced Access Content System

The Advanced Access Content System (AACS) is a DRM system that was developed by a consortium consisting of Disney, Intel, Microsoft, Panasonic, Warner Brothers, IBM, Toshiba and Sony in 2004 [1]. This system was applied to HD DVD and Blue-ray disks to anticipate the occurrence of copyright violation done by legitimate users. The AACS scheme consists of three parts.

1. Copy protection — The AACS prevents copying by using the Volume ID (VID) and a special key to enable the drive to handle this VID.

2. Decryption protection — To decrypt the disc, a user needs to acquire the Volume Unique Keys (VUK) consisting of VID and the processed device key.

3. Revocation — The AACS provides each individual playback device with a unique set of decryption keys through a broadcast scheme, called a sequence keys scheme [91, 93]. If a device's keys are compromised and published, the licensor can simply revoke those keys in any future content.

Some traitor tracing schemes, such as scoring methods and set-cover schemes, were developed to support the revocation mechanism [92, 94, 93]. More detail of the sequence keys scheme used in the AACS is presented in subsection 6.4.3.

2.3.2 DRM for Music

- Apple FairPlay (iTunes)

FairPlay is a DRM system developed by Apple. The system was initially applied to music files sold in Apple's iTunes store [10]. FairPlay uses a master decryption key and a user key that is used to decrypt the master key. Both keys are stored together with the data in the file. With this scheme, iTunes music could only be played on Apple devices and Apple's QuickTime media player. In November 2003, FairPlay was cracked by Jon Lech Johansen [180].

In 2007 the Apple's CEO, Steve Jobs posted an open letter advising record companies to leave DRM technologies [95]. Although Apple was forced to create a DRM system, there were three alternatives — Apple could continue down a DRM path; Apple could license the FairPlay technology to others; or record companies could be persuaded to license music without DRM technology. Obviously, the third option was preferable. Finally, Apple started offering DRM-free music in 2009. In spite of the fact that the company has already removed DRM from music in the iTunes store or in other vendors work on Apple's devices, the company's annual revenue still increased significantly. In fiscal 2014, Apple generated USD 182.8 billion in sales [186].

- Sony BMG

The most extreme example of copy protection may go to the Sony rootkit which became a scandal. In 2005, Sony BMG secretly included Extended Copy Protection (XCP) and MediaMax CD-3 softwares on millions of music CDs to prevent users from copying CDs [26]. The softwares were installed on users' computers without notice and worked as a form of DRM that sent information about what the users do with the CD through their PC. The XCP and MediaMax CD-3 which were undetectable by anti-virus and anti-spyware programs, were in the form of rootkits. The rootkit created a severe security vulnerability that others could exploit and thus, enabled other malware to infiltrate users' computers. Of course, this application caused some class actions which were ultimately settled by agreements to provide affected consumers with a cash payout or album downloads free of DRM [113].

2.3.3 DRM for e-Books

- Adobe ADEPT DRM

Adobe Digital Experience Protection Technology (ADEPT) DRM is a DRM system from Adobe that is applied to ePubs and PDFs which can be read by many third-party e-book readers [2]. This DRM scheme uses a complex crypto system. Each e-book is encrypted using a per-book AES key in the Cipher-block chaining (CBC) mode, and this key is encrypted again using a per-user RSA key. This encryption scheme is used to ensure a strong DRM mechanism, but it has a problem as it hides the per-user key. As a result, the system was broken by reverse-engineering the application and retrieving the user key.

- Apple FairPlay

The FairPlay system was later also adopted in ePub files designed for Apple's e-book application on iOS devices. The system encrypts files using the Advanced

Encryption Standard (AES) algorithm in combination with MD5 hash function [53].

- Amazon Kindle e-book

Amazon [8] provides the Amazon Kindle e-book reader to read e-books bought from the Amazon's Kindle store. To be able to purchase an e-book from this store, a user is required to have an account in the Amazon web site. The user then needs to install the Amazon Kindle e-book reader into his computer and register the installed reader to the Amazon under his account. Any purchased e-book from the Amazon's Kindle store will be downloaded into the Kindle's library in the user's device. The library will always be synchronised to the user's account. Although the downloaded e-book file may be copied into another folder, the Kindle's reader will refuse to read files outside its library.

In addition, a Kindle's e-book cannot be converted to a different format and can only be read using the Kindle's reader. Typically, an e-book converter, such as Calibre [74], will read the meta data of an e-book file before converting it into another format. Even though a Kindle's e-book can be copied into the Calibre library, Calibre could not open it as it is locked by DRM. This mechanism discourages users to illegally duplicate their Kindle's e-book, but may not preserve their privacy, since the users must be providing their accounts every time they purchase an e-book.

2.4 Open Source DRM Systems

In this section, the open source systems are not categorised by the type of content they protect, because each system is provided to be used by other parties that might work on various types of content.

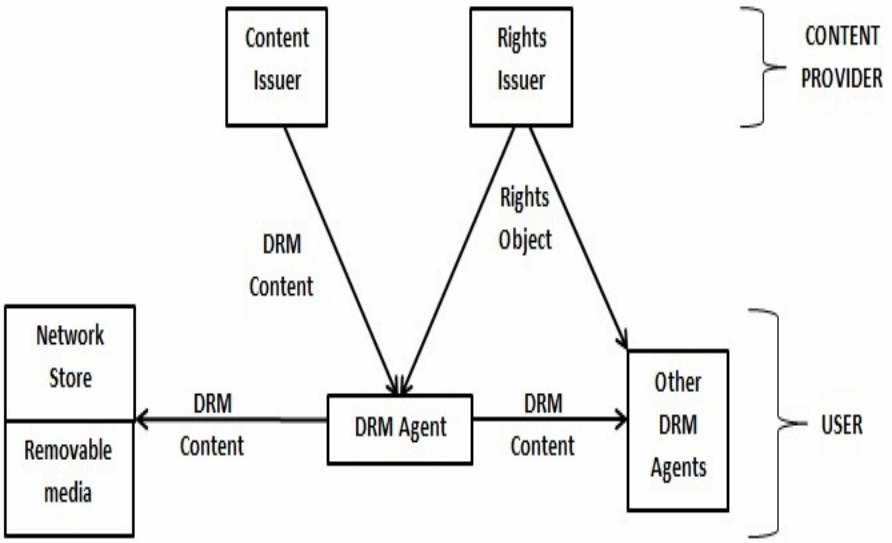


Figure 2.6: *Functional architecture of OMA DRM [137]*

2.4.1 OMA DRM

Open Mobile Alliance (OMA) DRM is an open source DRM system that aims to enable controlled consumption of digital media objects, super distribution of DRM content and content transfer between DRM agents [136, 137]. To achieve these goals, OMA DRM is complemented by a functional architecture that provides mechanisms to secure packaging and transferring usage rights and DRM content to trusted DRM agents and authenticating of those agents (see Figure 2.6).

Before content is distributed, it is packaged in a secure content container called DRM Content Format (DCF). DRM content is encrypted with a symmetric content encryption key (CEK). A content issuer then delivers the DRM content and rights issuer generates its associated rights object. A rights object, expressed using the Open Mobile Alliance Rights Expression Language (OMA-REL), is an XML document specifying permissions and constraints associated with a DRM content [138]. The document also contains the CEK to ensure that DRM content cannot be used without an associated rights object. OMA DRM makes a logical separation of DRM content from rights objects to ensure they can be requested and delivered separately or together.

To enforce rights objects at the point of content consumption, OMA DRM introduces a DRM agent. The DRM agent embodies a trusted component of a device that is responsible for managing permissions and constraints and controlling access to the DRM content on the device. Each DRM agent has a unique private-public keys pair and a certificate, enabling the content and rights issuers to securely authenticate the agent. However, this system does not specify a complete DRM infrastructure for mutual authentication between service providers and rights issuers.

A rights object is cryptographically bound to a specific DRM agent, so that only that DRM agent can access it. A rights object may optionally be bonded to a group of DRM agents (a.k.a. a domain), so that it can be accessed by all DRM agents that belong to the domain. As a DRM content can only be associated with a valid rights object, it can be freely distributed. However, to access the DRM content on a new device, a new rights object has to be requested and delivered to a DRM agent on that device.

The billing mechanism for obtaining rights, such as via mobile network operators (MNOs), is not considered in the OMA DRM system. Note that MNOs can play an important role to support users' privacy [176]. Thus, it might not be necessary to provide user's identity to the rights issuer. Instead, the MNO may act as a trustworthy party.

2.4.2 OpenIPMP

OpenIPMP is an open-source and open-standard based DRM system developed by Objectlab [131] that aims to provide the development community with interoperable DRM software that can be easily ported to any platform including Windows, Mac, Linux and any embedded platform. This system has been released to the software development and entertainment communities to promote an open-standards based Intellectual Property Management and Protection (IPMP) scheme for the Moving Picture Experts Group (MPEG)-4 audio and video content [117]. The openIPMP

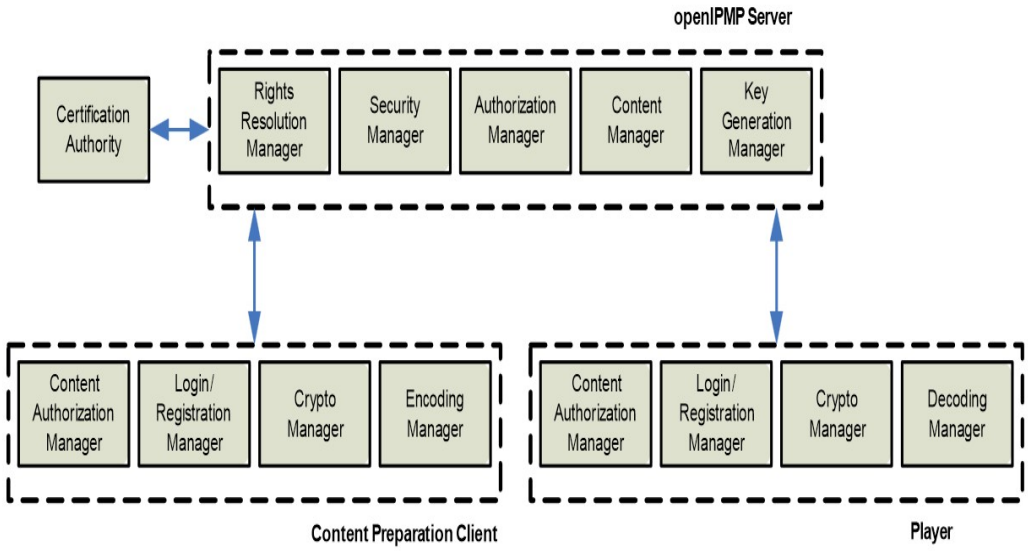


Figure 2.7: *The OpenIPMP architecture [173]*

system allows expression of licenses to use one of the leading rights expression languages, such as Open Digital Rights Language (ODRL) [88] and MPEG-21 Rights Expression Language [194].

The openIPMP framework (see Figure 2.7) is constructed by a set of tools and services capable to deliver a robust, scalable and adaptive infrastructure to support the management and secure distribution of content [173]. To manage rights and enforce the rules and permissions, the framework needs to be able to uniquely identify every user of the system. For this reason, each user is assigned with a digital certificate when they register with the system. The system also enables a secure and confidential communication with the openIPMP server components to ensure that sensitive data are not compromised during content distribution.

OpenIPMP's platform supports OMA DRM, Internet Streaming Media Alliance (ISMA) encryption and authentication specifications for MPEG-4 streaming (ISMAcryp) and MPEG's IPMP specifications for MPEG-2 and MPEG-4 [129]. Although openIPMP is designed to be integrated with any MPEG-4 or MPEG-2 based audio visual management system, the open source release comes fully integrated with MPEG4IP, the most popular open-source MPEG-4 encoding, decoding and play-

back software suite. Additionally, openIPMP plug-ins for Micosoft Windows Media Player and Apple Quicktime player are commercially available from a variety of MPEG codec vendors.

2.4.3 DReaM

DReaM (DRM everywhere available) is a Sun Microsystem Labs project to develop an open standards based DRM solution to support cross-service capabilities [59]. DReaM is flexible to deliver a mix of Sun-sourced technology interoperable with products from other vendors. The Microsoft Windows Media DRM (WMDRM) is an example of a solution for which the DReaM provides interoperability. The DReaM solution also supports DRM standards from Open Mobility Alliance (OMA) and Internet Streaming Media Alliance (ISMA).

DReaM architecture is independent of the content type, file and transport format. Whether the content is a time-based media (video, audio, music, games) or a document/data type, and whether it is digitally retrieved over a network (internet, service provider, peer-to-peer) or received on packaged media (DVD, CD-ROM, memory card, floppy, hard-disc), the content can be protected and managed using the same DRM mechanism since keys/licences are typically independent of the protected content itself. This architecture supports the separation between the rights management, user authentication, user identification, licensing, rights enforcement and protection system.

DReaM architecture is able to accommodate the inclusion of some DRM or Conditional Access System (CAS) components from other suppliers while avoiding the need to incorporate all their back-end components. The disintermediation system enables multiple instances of these components to exists in a DRM/CAS system. The process of disintermediation (see Figure 2.8) is as follows.

1. Client requests a licence.

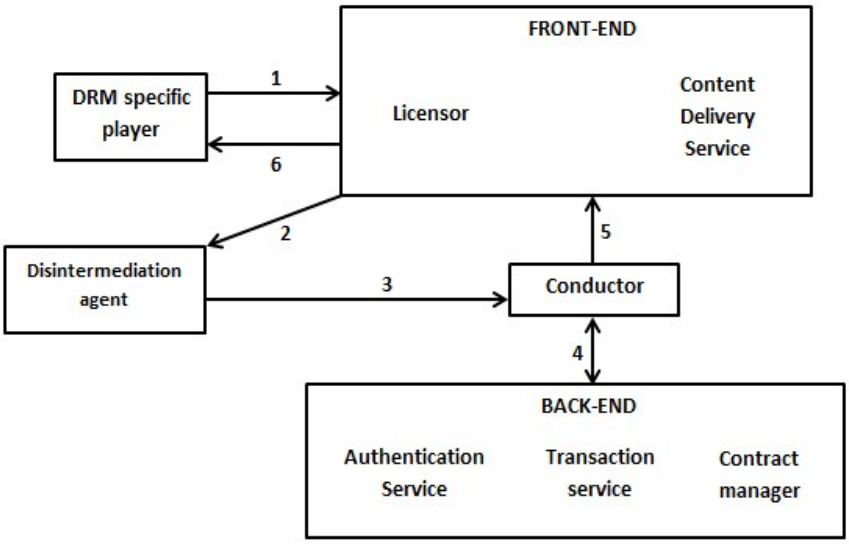


Figure 2.8: *DReaM Component Diagram [59]*

2. Front-end service redirects client to a client disintermediation agent.

3. Disintermediation agent contacts a conductor.

4. Conductor contacts back-end service for authentication and rights verification.

5. Conductor signals front-end service with instruction to deliver licence to client.

6. Front-end service delivers licence.

Sun's content protection technologies include DReaM-CAS (Conditional Access System) and DReam-MMI (Mother-May-I). The DReaM-CAS is intended to manage conditional access to content in the MPEG-2 format delivered through IP networking, using open standards such as SSL, Public Key Infrastructure, and AES or 3DES encryption. Asymmetric key technology is used to deliver individually protected keys to unlock the content keys.

The DReaM-MMI provides an approach to managing rights for a variety of client types that are directly or indirectly connected to content networks. The clients should be able to negotiate for rights through standardised protocols rather than downloading a licence with an inserted expression of rights. With the DReaM-MMI

method, no rights expression language (REL) is delivered to the client. Access is requested under certain conditions and the client software manages the use according to the guidelines. If a client wishes to access content under different usage terms, the client could renegotiate with the DReaM Licensor that is expected to let MMI create new permissions at any time.

The licensing process using DReaM-MMI is the same as the disintermediation process (see Figure 2.8). A protected content, its keys and the associated rights are stored in the content delivery server, the key and licence server (DReaM Licensor) and the rights repository (DReaM Contracts Manager), respectively. When a DReaM-MMI compliant client requests the rights to use the protected content, the DReaM Licensor communicates with the DReaM Contracts Manager to determine whether the client's request should be approved. If so, the content keys are delivered to the client. The DReaM-MMI compliant client is responsible for enforcing that the content is only used under the specified terms.

2.4.4 Marlin

Marlin is an open standard DRM initiatively developed by Marlin Developer Community (MDC) that aims at creating an inter-vendor interoperable platform. The MDC includes some companies such as Intertrust, Panasonic, Philips, Samsung and Sony. The Marlin architecture provides technological approaches to construct a user friendly DRM and copy protection. This technology enables users to purchase content through multiple distribution channels and to access it on any device that is part of their home domain [41]. Like the DReaM system, Marlin does not use Rights Expression Languages (REL), so that any patent issue regarding REL can be avoided.

To be flexible, most Marlin applications integrate two important platform technologies — the Octopus DRM architecture and the Networked Environment for Media Orchestration (NEMO) framework. An Octopus system has an Octopus DRM

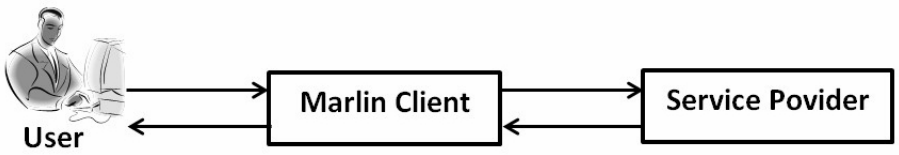


Figure 2.9: *Transaction model in the Marlin system [41].*

Engine component that controls and determines whether access should be granted under a given set of conditions. The system ignores the type of content it protects and does not depend on a particular set of semantics. This system is also able to issue content access rights separately from information that directs where or when it can be used. The NEMO framework provides the trusted connector between the various functional components in a Marlin system. Using the NEMO framework, Marlin components have a consistent mechanism to ensure that message is distributed with a suitable protection and is shared between entities that are properly authenticated and authorised.

Marlin's content acquisition model consists of three parties (see Figure 2.9).

1. Service provider — This party is an entity that provides services, such as a licence service and a registration service. In most cases, the service provider also serves as a content provider and collects content from content owners and distributes it to users.

2. Marlin client — A client is the functional components responsible for requesting licenses and links, and may be implemented in a hardware device or as a client application. When the host device requests access to content, the Marlin client executes the control program in the licence and checks the availability of any required links. If permitted by the licence, the Martin client allows the content key to be decrypted and used to access the protected content.

3. User — This party is an individual or a group of people that interacts with service providers to acquire licenses for digital content, and interacts with Marlin clients to access or manage the use of that content.

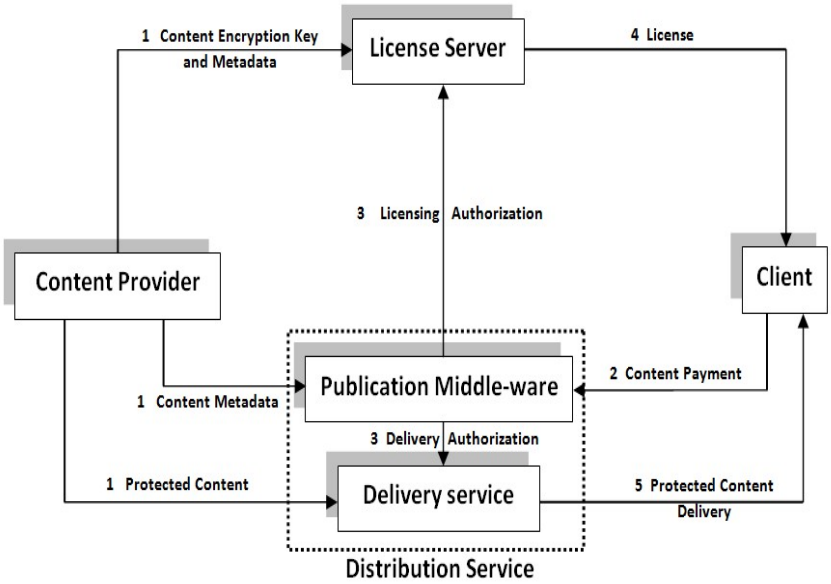


Figure 2.10: *DRM functional workflow [141].*

In the Marlin system, instead of being bound to the device that is used to obtain the rights, a licence is typically bound to a user. The relationship between users and device is maintained separately. In a simple purchase or rental model, users purchase rights to play a single content item that has been downloaded to their device. In a subscription model, users purchase rights to access a large collection of content for a limited period of time.

The Marlin DRM architecture also provides comprehensive support for broadcast and mobile delivery options. For example, the OMArlin specification governs how to enable interoperable downloading, streaming, sharing and consuming content between the OMA and Marlin systems. OMArlin content protection uses either OMA DRM 2.0 or Marlin Broadband to enable the distribution and consumption of digital content in a controlled manner. This specification enables a Marlin client to be considered as an OMA DRM agent, so that OMA content can be received, processed and consumed in the Marlin device without any modification on the OMA Rights issuers component.

The architecture of the open source DRM systems described previously (OMA DRM, OpenIPMP, DReaM and Marlin systems) underlies the construction of a ge-

neralized DRM functional workflow [141](see Figure 2.10). The workflow is started with content creation. In this stage, content provider also generates content identifier, content encryption key and content metadata. Content provider then sends the protected content to the delivery service, metadata to the publication middle-ware and content key to the licence server. A client can acquire content and make a payment to the publication middle-ware. After verifying the request and payment, the publication middle-ware signals the delivery service and the licence server to deliver the protected content and issue a licence, respectively, to the client.

Most of the open source DRM systems described above were focused on providing a secure interoperable system. The constructed architectures were flexible to be applied to various types of content and various platforms. Some systems were also capable of being integrated with another system. Moreover, to achieve security, most of the systems bind the content keys to the licence. Users can acquire a licence by providing their identity to the licence server. This means that the users' identity will be connected to the items they access by the licence server. However, none of the DRM systems provides adequate information as to whether they have a mechanism to protect users' privacy.

2.5 DRM and Privacy

DRM technologies usually utilise client-side DRM agents that work for content distributor or rights holder to control the usage of the distributed data. However, in general, this approach cannot provide high security as clients can do whatever they want with their own machine and can circumvent the protection scheme. The term of data usage control can be interpreted in two perspectives — DRM context and users' privacy. In the DRM context, this term bears the connotation that digital content is made available to use on the end-users' system, but the provider would like to restrict what the users can do with the content. In the privacy perspective,

the condition is reversed. The users who often supply personal data to content provider, would like to control how the provider uses the data.

Attempting data usage control is often beyond the data providers' capability. This is the fundamental issue of distributed usage control [151, 152]. Data providers would like mechanisms on the consumer's side to enforce their requirements and control how consumers' devices manage data. However, there are circumstances in which enforcing such control may not be applicable. Companies, for instance, are unwilling to permit other parties to control parts of their infrastructure. This situation limits content users as data providers, to control the use of their data by the companies. More description on current usage control (UCON) perspective is provided in Appendix A. Though the term of data usage control can be interpreted in the DRM context and users' privacy perspective, yet the UCON concept emphasises the term to the former more than to the latter. This means that the UCON concept considers content provider's security more than users' privacy.

The users' privacy has been increasingly connected to copyright enforcement. To deter unauthorised copies and to maximise benefit of the distributed digital content, content provider develop DRM technologies that enable more perfect control to access and use of digital content. The general functions that a DRM technology might perform include constraint, monitoring and self-help [40]. Constraint refers to the function for automatically enforcing restrictions on what users can do with the content. The monitoring function enables DRM technologies to report back to the content provider on the activities of users. The self-help function means that the direct restriction protocols can be designed to encode penalties as well as disabilities. These capabilities, however, often affect the content users' privacy because they create a potential for increased acquisition of information about users' intellectual habits and preferences.

Different DRM strategies have different effects on users' privacy. Feigenbaum *et al.* [57] presented some DRM strategies and how they affect the users' privacy.

- *Distributing persistent and complete DRM metadata with digital content.* Each digital content available on the content provider's website is formatted for use only by compliant applications. Using such applications, a user could only access the content as specified by the rights metadata. Under this strategy, rights metadata is added to the collectible information about the user.

- *Bonding downloaded content to a particular device or set of devices.* Before downloading digital content, a user is required to provide the ID of his device. This information would become another type of user-specific data that can be collected. This strategy also requires ongoing and periodic contact between user and content provider.

- *Bonding the downloaded content to the user.* After downloading content from a legitimate distributor, a user would have the right to use the content on any device at any time by proving that he is an authorised user. This strategy has a greater user-tracking than the other strategies.

A research that examined DRM-based online content distribution systems revealed that the most part of the assessed services do not accord with the users' expectations [121]. The examined DRM systems require a detailed investigation of content usage by users including the content accessed, the time of use, the frequency of use and the location of use. These services limit what users can do in the confines of their own home and create detailed reports about use of digital content. Obviously, by collecting information from users related to DRM implementations, businesses interfere with the privacy norms and expectations regarding the post-purchase use of content. Therefore, the following considerations need to be taken into account when a DRM system is developed [121].

- Data collection that is not absolutely necessary to protect content should be avoided.

- Data should be fully anonymised soon after collection as much as possible.

- When the justification for collecting information no longer holds, it should be destroyed.

- Additional data collection by the service and any data collection by third parties should be minimised.

- In all cases users should be notified that data collection is taking place.

Many privacy-preserving DRM schemes have been proposed to take the users' privacy aspect into consideration when a DRM system is developed. Minimising personal data acquisition in the most schemes is intended to disconnect the users' identity from the accessed content.

- An early privacy-preserving DRM scheme, presented by Conrado *et al.*, make use temporary pseudonyms which are managed by users' smart cards [42]. The temporary pseudonyms enable users to anonymously buy content from a provider and prevents the tracking of users while the content is accessed. User' identities are disassociated from their content. This scheme works by assumption that the smart cards do not reveal any personally identifiable information (PII).

- Another privacy-preserving DRM schemes were developed based on anonymous cash and blind decryption concepts [144]. The anonymous cash based scheme allows users to purchase content anonymously so that the content provider will know which content is being purchased but will not know who is purchasing. Conversely, the blind decryption based scheme requires users to purchase content non-anonymously so that the content provider can debit the users' account for payment. In this scheme, however, the content provider will be unable to figure out which content is being purchase.

- A recent privacy-preserving DRM scheme employs a software re-encryption scheme and combines secret sharing and homomorphic encryption [146]. This

scheme allows users who purchase software stay anonymous while the software licenses are bound to users and their validity is checked before execution. This mechanism prevent malicious users from relaying software to others.

These privacy-preserving DRM schemes, however, need to be improved to overcome more issues in their implementations. In this thesis, we proposed to improve the anonymous cash and blind decryption based schemes. The improved schemes are provided in Chapter 3.

2.6 Current State of DRM Research

To justify DRM capability, a literature study was conducted over the range of scientific papers and technical reports on DRM [157]. Evaluating DRM's rights protection capability yields contrary judgements. On one hand, the DRM systems are considered to be the best solution for combating digital piracy since the systems enable content providers to securely distribute content and prevent unauthorised distribution [81, 132, 165]. On the other hand, however, the persistent protection level in most DRM systems is still criticised, as the systems are easily attacked [178]. If even a small fraction of users are able to transform protected content into an unprotected form, then illegitimate distribution networks are likely to make that content available ubiquitously [78].

A recent study on existing DRM technologies identified some vulnerabilities including [205]:

- The analog hole — This factor is inevitable as all digital content must be decrypted in an analog form to be perceptible to humans [196]. Once content is in analog form, it is relatively simple to digitally recapture content in an unrestricted way.

- Key distribution — This factor is a hard task for a DRM system that utilises a

cryptographic mechanism as attackers have been able to obtain the encryption key in a particular way.

- Incompatibility — As media standards and formats evolve over time, old DRM-enabling media may be difficult to migrate to the new systems. Therefore, DRM must adopt advancing technology to avoid obsolescence.

The success or failure of DRM is often taken to be a technological question: has a particular scheme already been cracked? How broadly is protected content being redistributed? Can any scheme provide absolute security for content? By these measures, DRM, at least in its most visible applications, has failed [23]. Most widely available schemes are cracked within a few years of release. And due to the nature of the Internet, breaking a scheme once means it is broken everywhere. Under these conditions, absolute security is both required and inapplicable.

Between those two contrary judgements, there is also the argument that current DRM systems partially succeed. Although the DRM system successfully maintains copyright law and enhances owners' control over digital content, it fails to protect users' privacy and maintain *fair use* [148]. Therefore, user's need should also be considered in improving DRM's capability. The success of a DRM system does not merely depend on its persistent protection, but also on users' satisfaction on its implementation.

2.7 Major Problems for DRM Systems

Major problems for current DRM system include digital piracy, fair use and users' privacy. Digital piracy is considered a serious problem by content providers as it causes a significant drop in industry revenues and represents a threat to digital content marketers [182]. Meanwhile, fair use is problematical because content providers and users have conflicting interpretations of it. Another aspect, users' privacy, is recently mostly invoked by users due to its degradation caused by DRM implementation [28].

2.7.1 Combating Digital Piracy

Digital piracy can be seen as a result of the ease of copying and sharing content on the internet [80]. At least 1 in 4 bits of traffic on the internet is related to infringing content [30]. Some factors that potentially threaten intellectual property protection thereby increasing the need for DRM include [197]:

- the rising quantity of digitised content — Effective and efficient software allows rapid content digitisation; once digitised, it is much easier and simpler to copy and distribute.

- advancement of computer networking technology — New content distribution channels have been generated due to the advancement of computer networking technology. Such channels have the capacity to rapidly distribute significant amount of content.

- the sophistication of software functionalities — Sophisticated software has a simple mechanism and empowers end-users to manipulate digital content easily.

DRM's core technologies for combating digital piracy can be categorised as cryptographic-based and watermarking-based technologies. A cryptography mechanism accomplishes protection by securing content distribution using digital signature and encryption. Digital signatures are mainly used to protect the authenticity and integrity of content, rights holder and user, whereas encryption is practically used by most content providers to control access of content and prevent piracy [60]. Digital content is always distributed in an encrypted form. However, once the content has been decrypted, it becomes ordinary content and could be duplicated and redistributed in an unprotected form. Without copyright information, content provider is unable to trace the subsequent duplication and distribution of the content. This is an aspect of copy control that cryptographic-based DRM technology cannot afford [145].

Digital watermarking is viewed as an ideal technology to overcome the copy control problem [119, 51]. Recently improved, this technology effectively deters any unauthorised copying and plays a very important role in e-commerce [102, 150]. In practice, an identification code or copy control information can be attached to a digital watermark which is then embedded into the host content. Digital watermarking potentially preserves the content protection as the inserted watermark will remain associated with the content in its subsequent duplication and distribution. Together with digital signature, watermark is used to authenticate content, rights holder and user. In collaboration with the copy detection systems, watermarking technology is also employed to identify any illegal copy of content. Though watermarking helps to track the source of violation, however, it cannot prevent authorised users from retransmitting the content [34]. Therefore, integrating watermarking with a cryptography mechanism may improve the system's security.

The integration of cryptography and watermarking was recently proposed to achieve better protection in the DRM system. This collaboration is used to secure content distribution and protect the rights of both content providers and users [81, 183, 184, 185]. Cryptography handles network security issues; it ensures confidentiality, authenticity and integrity of content when it is transmitted through a public channel, while digital watermarking effectively protects copyright of content even after its transmission [102, 147]. At this point, the integration of watermarking and cryptography provides better security and higher efficiency of protection protocol. However, the implementation of this integration in a DRM system needs further investigation to measure its effectiveness.

2.7.2 Fair Use Issue

Fair use is a doctrine in U.S. copyright law and has been adopted by many countries worldwide. According to this doctrine, the use of copyrighted products for particular works such as criticism, comment, news reporting, teaching (including multiple

copies for classroom use), scholarship and research, without the the creators' permission, is not a copyright violation [135]. However, the distinction between fair use and violation is unclear and not easily justified, as there is no specified number of parts of content that may safely be taken without permission. In addition, acknowledging the source of the copyrighted content does not substitute for obtaining permission [134]. Under such a doctrine, the creators have no power to restrict the usage of their intellectual property.

Content providers and users have contrary interests in relation to fair use; while the providers attempt to control piracy by eliminating fair use, the users consider fair use to be their rights [148]. In fact, fair use is often invoked by users to prevent copyright holders from having exclusive control over their intellectual property more than copyright law intends [28]. In addition, the higher education community expects that the educational purposed DRM circumvention can be expanded to more educational body members [99]. These contrary interests need to be bridged primarily by establishing a strong legal foundation. However, without technological support, the rules are likely to be circumvented.

Technically, it is hard to implement fair use as required by copyright legislation. No DRM system is able to take such doctrine into account, because a computer program cannot make subjective decisions [105]. Many DRM solutions approach this issue by enforcing content usage control. This control not only affects those who may access content, but also affects how the content may be used or distributed afterwards. Nevertheless, no solution is able to cover all needs.

2.7.3 Users' Privacy Protection

Users' satisfaction needs to be considered when content providers endeavour to counter piracy. Although content providers urge the enhancement of the DRM's protection capabilities to effectively control piracy, this effort might not succeed without users' acceptance of the protection implementation. The protection typically con-

sists of some restrictions. For example, to limit content usage, a DRM application may make content to be accessed only on a particular device [166]. Or, the application may come with hidden files copied automatically into user's equipment to monitor what the user does with the content [69, 105]. These mechanisms are obviously not accepted by users as they cannot flexibly and privately access the content. In addition, the cost of DRM deployment will also be charged to the users. In the economical perspective, the anti piracy function in DRM becomes less effective if many restrictions are applied in using content. In this situation, DRM-free content becomes a more profitable option for the seller [4]. It is reasonable as the more restriction applied to the content, the more inconvenient for the consumers.

Users' privacy is often sacrificed when a DRM protection mechanism is implemented. Typically, to access protected content, user needs to acquire a licence by submitting his or her personal data for authentication process, assigning and controlling usage rights. However, content providers often use the data for additional purposes, such as customised goods, services and marketing information without the user's permission. In this case, users can be affected by unsolicited marketing and when the data is not adequately protected, it can be misused. The loss of privacy caused by DRM implementation becomes a critical problem because users' adoption of DRM technology is slow [159]. When purchasing digital content, users often have insufficient information on when, how and where their data would be collected, used or stored. Thus, users are often apprehensive about subsequent use of their personal data. Users often invoke privacy protection to preserve the ownership and distribution of confidential personal information [28]. Intensive anti-piracy measures cause concern among users about the protection of their privacy within the DRM system. This phenomenon indicates the growing attention of users to their personal information and the rising demand for regulations that are intended to protect privacy [76]. Moreover, the lack of privacy protection is certainly one of the main reasons behind the users' suspicion about DRM. Therefore, it is essential to develop privacy-aware

DRM systems.

Technically, users' privacy protection is approached by minimising personal data acquisition. The anonymous cash scheme, for instance, allows users to purchase content anonymously. The content provider will know which content is being purchased but will not know who is purchasing it. Conversely, the blind decryption mechanism allows content provider to know the user's identity, so that it can debit the user's account, without knowing which content is being purchased [144]. Another scheme approaches users' privacy protection by applying a partially blind signature in the licence acquisition protocol [58]. In this scheme, the licence server can generate a decryption key and deliver it to the user without knowing the corresponding key ID. Basically, in all schemes, user's identity will not be connected to the purchased item. Further, the schemes need improvements to satisfy various content distribution models and requirements. Detailed discussion and improvement on the anonymous cash and blind decryption mechanism are presented in Chapter 3.

DRM is required to provide balanced protection for content provider and users. This consideration is important to establish trust relationship and benefit balance among participants involved in a DRM-enabling contents distribution system [207]. However, in current DRM technological implementation, users' privacy is still insufficiently taken into account as the systems are mainly focused on combating piracy [82]. Therefore, to improve DRM capability, the needs of both users and content provider should be well understood. The former would like a DRM system that can handle most fair use scenarios, keeps the confidential data collected from users, does not monitor the usage of DRM data, allows for the transfer of rights, and is flexible depending on the media or situation. The latter would like a DRM system that can keep track of illegal use of DRM-enabled media, corrects collection of revenue from the usage of their works, creates a secure distribution channel, and prevents the illegal use of their works [11]. This description devolves to a statement: while content providers demand security, users need their privacy to be protected in

a DRM system.

2.8 Discussion for Improvement

Improving DRM means resolving major issues on piracy, fair use and users' privacy. In practice, effective DRM protection can be achieved if piracy is controlled, fair use is maintained and users' privacy is protected. Though there is no single solution that can overcome all problems, at least some approaches can be considered to minimise the impact of each problem. To be effective, an approach should be directly focused on content in its life cycle.

Digital piracy could be controlled firstly by securing content delivery. Encryption is typically used to prevent any illegal access on the protected content; enabling only authorized users to unlock content and use it properly. When content reaches end-users, there should be a mechanism to control how users use the content. Usage control is implemented by enforcing restrictions to ensure that users obey the usage rights and obligations assigned to them. However, in a particular situation, malicious users are likely to deny copyrights — making illegal copy and redistributing it for their own benefit. At this case, a mechanism to trace the source of violation is required.

Fair use might be problematic, but it could be maintained by creating strict content usage rules. At the time of requesting content, users must have a good understanding of the rights and obligations regarding the use of content. Again, an appropriate usage control scheme is needed to enforce the restrictions in this case.

Users' privacy protection could be approached by managing personal information properly. DRM systems should gather data no more than necessary and store it for no longer than necessary for executing rights enforcement. Additionally, to enhance users' acceptance of the DRM technology, there should be adequate information for the users about when their consumption patterns are gathered, to whom such

information is given, and to what level of security the information is ensured.

Based on the description above, technologically, the DRM improvement should include three main schemes — digital piracy control, content usage control and users' privacy protection. Piracy control aims at preventing any illegal access, copying and distribution of protected content, and tracing the source of violation if it occurs. Protection methods that are likely to be used to control piracy include encryption, watermarking and traitor tracing. Usage control schemes aim at enforcing content usage restriction. However, this enforcement should not sacrifice users' privacy. Users' privacy protection, technically, is done by minimising personal data acquisition. This mechanism is set in such a way that user' identity would not be connected to the purchased content. This means that one of them (either user's identity or the content ID) must remain unrevealed to the content provider.

Balanced consideration of piracy control, content usage control and users' privacy protection schemes lead to balanced protection for content provider and users in a content distribution system [157]. The implementation of these schemes, however, may not be handled separately. They need to be integrated in any content protection mechanism at any stage of the content life-time.

1. At the time of purchase — The user's personal data might not be connected to the purchased item, so that the user's privacy can be protected. The problem is how is it possible to protect users' privacy while still maintaining content provider's security?

2. At the time of content delivery — The protection scenario is focused on securing content delivery. The question is how is it possible to secure content delivery, while the system still preserves users' privacy?

3. After content is delivered — The protection scenario is intended to deter users from violating copyright of content. However, if a user commits a violation, a traitor tracing scheme can be utilised to identify the violator. In this case,

security and privacy requirements are also applied — how it is possible to accurately trace the real traitor without framing innocent users?

This thesis is intended to investigate some protection schemes at three stages:

- protecting users' privacy at the time of content purchasing;

- securing content delivery;

- preserving protection after content is delivered (at the time of content usage).

These protection schemes do not belong to a continuous protection system. A constructed protection scheme in a particular stage is independent of the schemes in other stages. The implementation of the schemes will depend on the business model applied in a content distribution system. A model may integrate the schemes in a particular way or implement any scheme independently.

Chapter 3

Protecting Privacy at the Transaction Time

3.1 Introduction

Most protection scenarios in the DRM systems focus on achieving security for content providers and often pay less attention to users' privacy. This chapter provides protection mechanism with a different perspective — protecting users' privacy while still preserving content provider's security. At the time of content purchasing, users' privacy protection can be approached by minimising users' personal data acquisition. This approach allows users to purchase an item in such a way that their identity would not be connected to the purchased item. The implementation of this approach, however, must be controlled so that any malicious action that disadvantages content providers' security can be prevented. The investigation is focused on two schemes — anonymous cash and blind decryption. These schemes are constructed using the notion of the blind signature concept [31].

A blind signature mechanism allows a user to have another party signs something without the signer knowing what is being signed [31]. Suppose a user (Bob) wants to have a signer blindly signs his document, x. This mechanism works if it satisfies:

i. Bob can prove to the signer that he is who he says he is. This condition needs an authentication scheme.

ii. No one can duplicate the signer's signature, but anyone can verify that the signer's signature is valid. These conditions require that the signing function S' known only to the signer has a corresponding function S. The function S is the inverse of S', but gives no hint about S'.

iii. Bob can obtain the signer's signature on his document, x, in such a way that the signer is incapable of knowing it. This condition requires Bob to have an encryption function E_B and its inverse, a decryption function D_B. Both are known only to Bob. Additionally, E_B and S' must be compatible, in the sense that $D_B(S'(E_B(x))) = S'(x)$ for any document x.

If these requirements are fulfilled, a blind signature protocol can be undertaken (see protocol 3.1).

Protocol 3.1: Blind Signature

1. Bob applies E_B to his document x, and sends $E_B(x)$ to the signer, using a sort of authentication scheme.
2. The signer verifies Bob's authentication, applies S' to $E_B(x)$, and sends $S'(E_B(x))$ back to Bob.
3. Bob applies D_B to $S'(E_B(x))$ to obtain $S'(x)$, and verifies that $S(S'(x)) = x$.

3.2 Anonymous Cash

Anonymous cash is a digital analogue of cash. The basic property of cash is anonymity — when Bob withdraws money from a bank, the bank gives him the cash without knowing what Bob buys, and when Bob spends money, the merchant has no idea who Bob is. In contrast, if Bob purchases something using a credit card online, he has to provide his identity to the merchant, and inform the credit card

provider who he is making a purchase from. This mechanism has the potential to invade Bob's privacy.

Anonymous cash allows users to purchase content in such a way that content provider will know which content is being purchased but will not know who is purchasing it. Although not knowing the buyer's identity, the content provider is allowed to determine the royalties to the copyright holder based on the sold items. The implementation of anonymous cash for DRM consists of two stages — obtaining tokens and purchasing content.

Obtaining Tokens. Tokens act as digital cash. A user can purchase tokens, either from content provider (merchant) or from a third party (such as bank). All tokens are assumed to be worth the same amount, such as a dollar. Of course, purchasing tokens must be done non-anonymously so that the token provider can charge the user for payment.

How user obtains token is based on blind signature concept [31]. First of all, Bob chooses x at random, applies a public hash function f, and blinds the result by encrypting it with E_B. Bob then sends $E_B(f(x))$ together with his signature to the token provider. The token provider verifies Bob's signature, signs $E_B(f(x))$ using its secret signature function, S', and debits Bob's account. Bob then computes $D_B(S'(E_B(f(x))))$ to strip away his encryption and obtain $S'(f(x))$. Note that E_B and D_B are only known by Bob. Using token provider's public key, S, Bob can verify whether $S(S'(f(x))) = f(x)$.

In the communication for purchasing tokens, the token provider needs two private key operations [144]. One private key operation is used to blindly sign each token. Another private key operation is required to set up the encrypted conversation in which consumer purchases tokens. Thus, the token provider will execute two computations when a consumer requesting tokens. Note that the communication for purchasing tokens could be done with a long-term shared secret key between the token provider and the consumer and many tokens can be purchased in the same

conversation.

Purchasing Content. In the communication of requesting content key, a private key operation is needed to establish the server-side-authenticated encrypted channel. When purchasing content key, Bob presents the token together with the metadata of the requested content, and the content provider returns the content key. More precisely, Bob gives the pair of $(S'(f(x)), x)$ to a merchant. The merchant verifies whether $S(S'(f(x))) = f(x)$ to make sure the token was actually signed by the token provider. The merchant then sends $S'(f(x))$ and x to the token provider. The token provider checks the validity of the signature, pays the merchant, and puts x on a list of tokens that have been spent.

3.2.1 Avoiding Double-Spending

Due to anonymity, the transactions using either real money or anonymous cash, provide mechanism to prevent double-spending. In the use of real money, for instance, double-spending may occur when a user spends fake money. To avoid this double-spending, authorised money is only manufactured by an official authority such as the central bank. If an illegal party attempts to duplicate the authorised money, the fake one can easily be identified. Similarly, in the anonymous cash scheme, double-spending occurs when a same valid unit of token is spent more than once. In this scheme, the key used by token provider to sign the tokens is assumed to be a perfectly kept secret, so that another parties are unable to duplicate an authorised token. If a user spends his token more than once, the token provider will be able to identify this double-spending and reveal who did it. In a basic anonymous cash scheme, however, a user could be immediately caught trying to double-spend when the merchant is online and has the token provider checks the user's token in real time.

To make double-spending harder to do, Chaum *et al.* constructed an electronic cash protocol [32]. In this protocol, the bank as token provider fixes a security

parameter k that determines how likely double-spenders are to be caught; the higher the parameter k, the higher the probability that a cheater will be caught. Bob, as a user, has an account number u with an associated counter v; both Bob and the bank know u and v. The protocol employs two functions $f, g : \mathbb{Z}_n \times \mathbb{Z}_n \rightarrow \mathbb{Z}_n$ which are hard to invert. f and g are two-argument collision-free functions; that is, for any particular function, it is infeasible to find two inputs that map to the same point. f is required to be similar to a random oracle. For unconditional untraceability, g is also required to have the property the first argument gives a one-to-one map from the second argument onto the range [32].

Assume that a token is worth one dollar. Each of Bob's token consists of k 4-tuples $(a_i, c_i, d_i, r_i) \in \mathbb{Z}_n^4$ which are randomly chosen. To have a token signed, Bob proceeds Protocol 3.2 (all arithmetic is done mod$n$) and then to pay a merchant (say, Alice) one dollar, Bob and Alice proceed the Protocol 3.3.

Protocol 3.2: Electronic cash

1. Bob computes and sends to the bank k blinded values $B_i = r_i^3.f(x_i, y_i)$, for $1 \leq i \leq k$, where $x_i = g(a_i, c_i)$ and $y_i = g(a_i \oplus (u||(v+i)), d_i)$.
2. The bank chooses a random subset R of $k/2$ indices and sends it to Bob.
3. Bob reveals (a_i, c_i, d_i, r_i) for $i \in R$, and the bank checks that these 4-tuples yield the valid values of B_i. If Bob tried to cheat, it is likely that the bank catches him in this step.
4. If the 4-tuples Bob reveals are valid, the bank gives him $\prod_{i \notin R} B_i^{1/3}$, charges his account one dollar and increments his counter v by k.
5. Bob multiplies $\Pi_{i \notin R} B_i^{1/3}$ with $\prod_{i \notin R} r_i^{-1}$ to extract his token

$$C = \prod_{i \notin R} f(x_i, y_i)^{1/3}.$$

He also increments his copy of the counter v by k.

Protocol 3.3: Purchase item using e-Cash

1. Bob sends C to Alice.
2. Alice chooses a random binary string of length $k/2$: $z_1, z_2, ..., z_{k/2}$, and sends it to Bob.
3. For $1 \leq i \leq k/2$, if $z_i = 1$, then Bob sends Alice a_i, c_i and y_i; if $z_i = 0$, then Bob sends Alice x_i, $a_i \oplus (u||(v+i))$ and d_i. Alice then verifies that these data fit the value C Bob provided. Since f and g are assumed to be impossible to invert, and the bank's signing scheme is assumed to be impossible to duplicate, if Bob is cheating then he is likely to be caught.
4. Alice sends C and the transcript of the conversation to the bank, which verifies their correctness, pays her and keeps the transcript on file.

The electronic cash protocol allows Bob to maintain his anonymity as long as he does not cheat. The consequence of cheating is having his identity revealed. The only identifying information Bob gives to Alice is his account number u, but it is XOR-ed with a random string a_i, so Alice cannot see it. Even if Alice can invert g, she still has an enormous number of possible pairs (a_i, c_i) and $(a_i \oplus (u||(v_i)), d_i)$ which g maps to x_i and y_i. However, if Bob attempts to double-spend the same token, the bank will compare the transcripts from two purchases where he used the token. Due to the randomized binary strings chosen by merchants, it is likely that there is some i such that $z_i = 0$ in one transaction and $z_i = 1$ in the other. The bank can XOR Bob's two responses to obtain $u||(v+i)$, which identifies Bob's account number and deduces a double-spending.

Double-spending might perfectly be prevented in the electronic cash scheme, but framing may emerge in this scheme. While the bank can discover double-spenders, it can also forge transcripts to frame innocent users [32]. To prevent a frame-up, a user should have a digital signature scheme and a certified copy of his public key. First of all, the user chooses k pairs of random integers z_i' and z_i''. When computing $f(x_i, y_i)$, he replaces his account number u with $u||z_i'||z_i''$. Along with the blinded values B_i, the user sends the signed concatenation $g(z_1', z_1'')||g(z_2', z_2'')||...||g(z_k', z_k'')$.

The bank can prove that the user cheated if it can provide $k/2 + 1$ correct pairs (z_i', z_i''). Since the bank cannot break the user's signature, it cannot change the given values of $g(z_i', z_i'')$. Even if the bank can invert g, the user only has to provide his own pairs (z_i', z_i'') to show that g has actually been broken.

3.2.2 Advantage and Disadvantage

One advantage of the anonymous cash scheme is that the content provider is allowed to know how many items have been purchased. This benefit might be important in some applications to determine the royalty amount for each contributor. However, many current content distribution schemes, such as premium TV channels, apply a flat rate payment regardless of how much or which content is accessed within the channel.

A disadvantage of the implementation of anonymous cash (and also electronic cash) is that it is less efficient and costly [144]. First of all, the anonymous cash scheme needs two conversations — a non-anonymous conversation to purchase tokens and an anonymous encrypted conversation to purchase content. Additionally, the scheme requires at least three private-key operations — one for the token provider to blindly sign each token, one for establishing the server-side-authenticated encrypted channel for content (key) requests and one for setting up the encrypted conversation when a user purchases tokens. Moreover, the scheme also requires content providers to keep a large database of keys for all the content items.

3.3 Blind Decryption

Blind decryption is more efficient and less expensive than anonymous cash. A blind decryption scheme only needs one conversation, that is, when a user requests a content key.

Blind decryption has a similar notion as blind signature — this scheme allows

a user to have a party called blind decrypter to decrypt a ciphertext without the decrypter knowing what is being decrypted [143]. In this mechanism, the user has a *blinding* function to make the ciphertext unintelligible to the decrypter before decryption and the associated *unblinding* function to make the text clear to the user after decryption. In the context of DRM for content distribution system, the blind decrypter could be the content provider who holds the encryption and decryption key for the distributed content.

Typically, a blind decryption scheme is a public-key encryption (PKE) scheme, but in a particular computation, the scheme can also be done without public keys. Blind decryption provides an efficient protocol for obliviously decrypting ciphertexts [75]. In such a protocol, a user of a ciphertext interacts with the secret key holder. At the end of the protocol, the user obtains the plaintext while the key holder cannot figure out what it decrypted. With this property, the blind decryption scheme can efficiently protect users' privacy in online marketing [5]. In this scheme, user's identity is still revealed to the content provider so the provider can debit the user's account, but the information about which content the user purchased is hidden. Thus, the user's identity will not be connected to the purchased content.

Blind decryption can be performed using some types of mathematics [143]. In all scenarios, content provider encrypts the content key and then makes it available for users. The encryption key could be public or secret. A user who wishes to access the content key blinds the encrypted content key and then sends it to the content provider who will blindly decrypt the content key.

3.3.1 Blind Decryption with an RSA Key

Blind decryption can be applied using RSA cryptosystem [161]. This form of blind decryption is almost identical to blind signatures with an RSA key. Suppose content provider has public keys (e, n) and secret keys (d, p, q). Content provider encrypts the content key, m, to be m^e. All arithmetic is done mod$n$. To obtain m, a user

performs protocol 3.4.

Protocol 3.4: Blind decryption with an RSA key

1. The user blinds m^e by choosing a random number r and computes $r^e m^e$.
2. The user then sends $r^e m^e$ to the content provider.
3. The content provider computes $(r^e m^e)^d$, and sends back the result, rm, to the user.
4. The user divides rm by r and obtains m.

3.3.2 Blind Decryption with a Diffie-Hellman Key

Blind decryption using a Diffie-Hellman key [50] has no similar blind signature scheme. This means that this scheme works for blind decryption, but would not work as a blind signature.

Suppose content provider's public key is g^x mod p, where g and p are known, and the private key is x. Assume that all operations are being done mod$p$. Each item of content is encrypted with a different key (y), but all items use the same secret x. Thus, for an item, the content key is of the form $m = g^{xy}$, for a particular y. The content key is used as a symmetric key for any symmetric algorithm such as AES. The metadata associated with the item that is encrypted with key g^{xy} includes g^y.

To obtain a content key g^{xy}, a user can simply submit g^y to content provider. However, for blind decryption purpose, the user have to blind g^y first. The detail of blind decryption with a Diffie-Helman key is presented by protocol 3.5.

Protocol 3.5: Blind decryption with a D-H key

1. The user chooses a number z and computes z^{-1} mod q, where $q = | < g > |$.
2. The user computes g^{yz} and sends it to the content provider.
3. The content provider computes g^{xyz} and sends it back to the user.
4. The user computes $(g^{xyz})^{z^{-1}}$ and obtains g^{xy}.

3.3.3 Blind Decryption without Public Keys

This form of blind decryption could not have a similar blind signature scheme as there is no public key with which to validate a signature. Instead, the content provider has two secret numbers, x and x^{-1}, which are exponential inverses mod$p$, to encrypt and decrypt the content key, respectively.

First of all, the content provider encrypts the content key, m, and then makes m^x available for users. To have the content provider blindly decrypt m^x, a user proceeds protocol 3.6.

Protocol 3.6: Blind decryption without public keys

1. The user chooses random y, and its exponential inverse y^{-1}.
2. The user computes m^{xy} and sends it to content provider with the request to "decrypt".
3. The content provider applies x^{-1} and returns m^y.
4. The user applies y^{-1} to obtain m.

3.3.4 Acquiring Blind Decryption

The blind decryption scheme does not need an anonymising network. Instead this scheme only needs a non-anonymous conversation for user to purchase content key (see Figure 3.1). It is not necessary for this conversation to be encrypted, but must

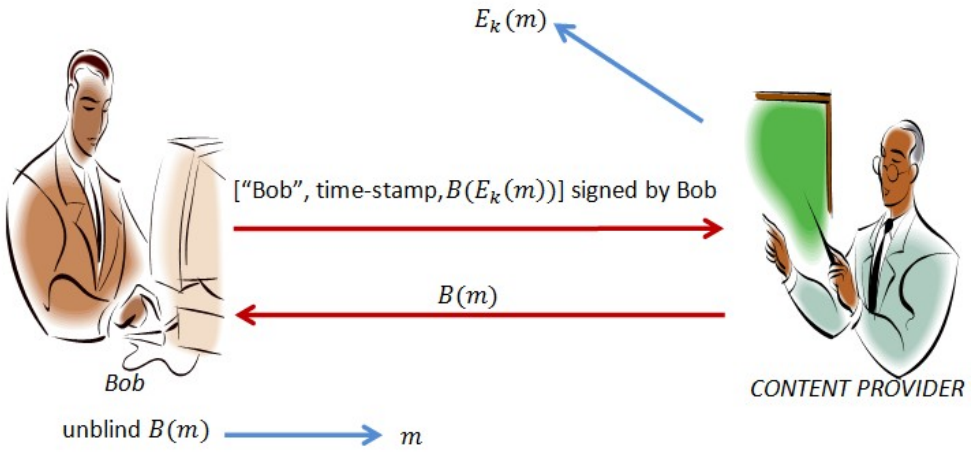


Figure 3.1: *A typical blind decryption scheme*

be signed by the user.

Suppose content provider encrypts the content key m with the encryption key K to be $E_K(m)$, and makes the encrypted content key available to the public. To obtain m, Bob blinds the encrypted key with his blinding function B, to be $B(E_K(m))$, and send it to the content provider for a blind decryption. The decryption request must be signed by Bob, so that content provider will be able to authenticate Bob and debit his account. The request Bob sends must also be resistant to repetition to avoid multiple debiting for the same decryption. To prevent such a repetition, Bob adds a time-stamp in his request and has the content provider store it:

["Bob",time-stamp,$B(E_K(m))$] signed by Bob.

After verifying Bob's signature, the content provider debits Bob's account, decrypts $B(E_K(m))$, and sends $B(m)$ back to Bob. Bob unblinds $B(m)$ and obtains m.

3.4 Varying Prices and Privacy Issue

Items' pricing is not an issue in the anonymous cash scheme. Knowing which item is being purchased, content provider can ask users to provide tokens according to the price of the purchased item. For example, a user has to submit n blindly signed

tokens to purchase an item worth n units. For any item's price, user's privacy is kept protected due to anonymous content purchasing. Even though user non-anonymously purchases tokens, his identity cannot be connected to the items he is going to buy. A large denomination token is not always intended to buy a single expensive item, as user can use this token for purchasing multiple cheap items in the same transaction.

The pricing also causes no issue for the blind decryption scheme if the prices of all items are flat. Content provider can apply the same cost for each requested blind decryption. Even though user non-anonymously purchases an item, the identity of the item remains unrevealed to content provider. However, a different situation occurs when the prices vary. Content provider cannot debit user's account without information about the price of the purchased item. Therefore, user has to submit the item's price along with the request of a blind decryption. As a consequence, the purchased item could be identified from its price.

3.4.1 Single Decryption per Unit Price

A trivial solution of varying prices and privacy issue is decrypting content key per unit price of item. Suppose an item is worth n units and m is the content key to access the item. First of all, content provider splits m into n shares of m_i, where $1 \leq i \leq n$, such that

$$m = m_1 \oplus m_2 \oplus ... \oplus m_n,$$

and encrypts each share using the encryption key K to be $E_K(m_i)$, for $1 \leq i \leq n$.

To obtain the content key m, a user blinds each encrypted share using his blinding function B, and then asks content provider to do n blind decryptions, each for $B(E_K(m_i))$, where $1 \leq i \leq n$. Content provider then charges the user's account for 1 unit per decryption. The user unblinds each $B(m_i)$ to be m_i, and xors all these shares to obtain m.

Counting all required decryptions, however, enables content provider to identify the price of the purchased item. To avoid this reveal, the user could spread decryption requests over time [144]. This strategy will obfuscate the exact amount of any item, or at least reduce the probability of the item's price being identified.

3.4.2 Decryption Using Multiple Valued Keys

Acquiring n blind decryptions to buy an item valued of n units causes an overload of $n-1$ additional private key operations for content provider. The same overload is also applied when user requests n single tokens to purchase such an item in the anonymous cash scheme. To reduce this burden, content provider needs to have different pairs of encryption-decryption keys for different denomination of units [144]. In the anonymous cash scheme, for example, the token provider could have keys: K_1, K_2 and K_3, to sign 1-unit, 10-unit and 100-unit tokens, respectively. To buy an item valued at 112 units, a user could present 112 1-unit tokens, or one 100-unit token and 12 1-unit tokens, or one 100-unit token, one 10-unit token and two 1-unit tokens.

In the blind decryption scheme, the content key can be split in such a way as to achieve the most efficient decryption. Suppose content provider's keys are K_1, K_2 and K_3 with denominations 1, 10 and 100, respectively. The content key of an item worth 112, for instance, could be split into four shares and the metadata would contain four wrapped keys:

$$(unit = 100, E_{K_3}(m_1)),$$
$$(unit = 10, E_{K_2}(m_2)),$$
$$(unit = 1, E_{K_1}(m_3)),$$
$$(unit = 1, E_{K_1}(m_4)).$$

In this case, content keys with the same denomination have to be encrypted with the same key. This means that the content provider should have only one key

for a particular denomination, because the price submitted by the user is the only clue enabling the content providers to determine which key should be used for a decryption.

To obtain the content key, Bob would request the content provider for four blind decryptions:

["Bob",time-stamp,$B(E_{K_3}(m_1))$,unit=100] signed by Bob

["Bob",time-stamp,$B(E_{K_2}(m_2))$,unit=10] signed by Bob

["Bob",time-stamp,$B(E_{K_1}(m_3))$,unit=1] signed by Bob

["Bob",time-stamp,$B(E_{K_1}(m_4))$,unit=1] signed by Bob

Again, Bob has to spread these decryption requests over time to keep the price of the purchased item from being disclosed.

To improve the privacy protection in the blind decryption scheme, content provider can use various ways to split the content's key m. For an item worth 34 units, for instance, m is split into $m_1, m_2, ..., m_{34}$, worth of 1 unit each; and m is also split into $m_1, m_2, ..., m_7$, where $m_1, ..., m_3$ worth of 10 units each and $m_4, ..., m_7$ worth of 1 unit each. With these splits, user can opt for which way he wants content provider to decrypt m. More decryption options gives better protection of user's privacy.

3.4.3 Can Blind Decryption be Requested Once?

Though spreading the decryption requests over time can minimise the opportunity of revealing content ID, it is time consuming. Additionally, separate decryption requires an extra operation for content provider. In this case, the option with the smallest number of blind decryptions is the best choice from the content provider's point of view. Is it possible for user, in the blind decryption scheme, to request blind decryption once without a risk of revealing his purchased item?

Requesting all units decryption at once is possible only when content provider uses a single decryption key. However, if the items' prices vary, this condition causes a payment charging problem. For example, suppose content provider uses RSA cryptosystem with the pair of encryption and decryption keys, (e, d). For an item worth t units, the provider splits the content's key m into k shares, $m_1, m_2, ..., m_t$ so that

$$m = m_1.m_2...m_t.$$

Each share is worth one unit and is encrypted separately. To obtain m, instead of requesting decryption per unit price, a user multiplies all encrypted shares and blinds the multiplication result by choosing a random number r and computing

$$b = r^e.(m_1^e.m_2^e...m_t^e).$$

Regardless of the value of t, content provider can apply d to b so that the user will successfully obtain m. However, having no information about the value t, content provider cannot determine how much amount has to be charged to the user.

The same problem also occurs in a symmetric cryptosystem. Suppose E_K is an exclusive OR with the key K and the content's key m is split into t shares $m_1, m_2, ..., m_t$ so that

$$m = m_1 \oplus m_2 \oplus ... \oplus m_t.$$

Each share is then encrypted separately. To obtain m, a user computes

$$b = r \oplus (\oplus_{i=1}^{t} E_K(m_i)),$$

where r is a random binary number, and asks content provider to blindly decrypt b. The content provider then applies E_K to b, and the user can obtain m by computing

$$m = r \oplus E_K(b).$$

Here, t is required to be an odd number. Otherwise, m will be revealed when the user computes $\oplus_{i=1}^{t} E_K(m_i)$. Again, in this case, content provider is unable to charge the user's account.

Based on the analysis described previously, we suggest that a blind decryption can be requested once without experiencing varying price and privacy issues, if the scheme satisfies the following:

- The content keys of all items having the same price are encrypted using the same key; knowing the item's price enables content provider to determine which key has to be used to decrypt the content key.

- There should be many options in a group of items with the same price. With this condition, content provider will know to which group an item belongs, but will not know what it is.

- The item's price should not immediately indicate the type of the item. For example, X-rated items may be more expensive than G-rated items. However, content provider should also provide a package containing some G-rated items having the same price as an X-rated item. With this condition, a user who quotes an expensive unit for his blind decryption request does not always intend to purchase an X-rated item.

3.5 Buyer Authorisation

Buyer authorisation is another aspect that needs to be considered in a content transaction. For example, an item may only be permitted to the buyers of a certain age group, or the member of an organisation, or the citizen of a particular country, and so on. For this reason, content provider should also be able to verify whether a buyer satisfies the content access condition.

To be eligible when purchasing a restricted item, a user firstly has to obtain an appropriate authorisation certificate (Ac). An Ac is a triple $(A_{ID}, A_{ATR}, A_{EXP})$.

A_{ID} is the registration number of the certificate. A_{ATR} is the authorisation attribute, such as "more than 18 years old", "university student", "member of IEEE", "citizen of Australia", and so on. A_{EXP} may indicate how many times the certificate can be used to purchase items, or when the certificate expires. For example, the certificate $(A_{ID}, "university\ \ student", 10)$ can be used to purchase 10 items permitted only to the university's students; the certificate $(A_{ID}, "member\ \ of\ \ IEEE", "31-12-2014")$ can be used to purchase items specified for the members of IEEE by 31 December 2014. Of course, obtaining an authorisation certificate must be done non anonymously.

Once user's data has been verified, content provider releases 2 copies of a signed authorisation certificate ($Sign_K(Ac)$). The first copy is stored in the content provider's server and updated every time the user uses the certificate. The second copy is given to the user. The user can verify the signature using the content provider's public key. Each time the user uses his certificate, it is confirmed to its copy stored in the server. If its A_{EXP} refers to a number of purchases, content provider reduces the A_{EXP} by 1 in every transaction.

3.5.1 In the Blind Decryption Scheme

An authorisation certificate used in a blind decryption scheme has a specific A_{ID} format. The A_{ID} is a concatenation of the registration number and the user's account number. Each time the certificate is used, the associated user's account will be charged for a portion of the item's price. This mechanism aims at discouraging the authorised users from sharing their certificate to unauthorised persons.

Consider the following illustration. Suppose an item is worth 50 units and is permitted to any university's student who is a member of the IEEE. The item's key, m, may be split into 3 shares m_0, m_1, m_2, such that

$$m = m_0 \oplus m_1 \oplus m_2.$$

The share m_0 is then encrypted using key K_0 worth 30 units; m_1 and m_2 are encrypted using keys K_1 and K_2, respectively, each worth 10 units. The meta data of the item consists of 3 encrypted shares and 2 authorisation attributes:

$$(unit = 30, E_{K_0}(m_0)), (("uni\ \ student", E_{K_1}(m_1)).\text{AND}.("IEEE", E_{K_2}(m_2))).$$

Assume a user (say Bob) has certificate Ac_1 and Ac_2 with "university student" and "IEEE member" attributes, respectively. Bob is eligible to purchase the item. He then blinds the encrypted keys and requests following 3 blind decryptions separately:

["Bob",time-stamp,$B(E_{K_0}(m_0))$,unit=30] signed by Bob;

["Bob",time-stamp,$B(E_{K_1}(m_1))$,$Sign_K(Ac_1)$] signed by Bob;

["Bob",time-stamp,$B(E_{K_2}(m_2))$,$Sign_K(Ac_2)$] signed by Bob.

If the operator for the authorisation attributes is "OR", content provider needs to split m into two shares

$$m = m_0 \oplus m_1.$$

The item's meta data would be

$$(unit = 40, E_{K_0}(m_0)), (("uni\ \ student", E_{K_1}(m_1)).\text{OR}.("IEEE", E_{K_2}(m_1))).$$

In this case, content provider can keep the price of K_1 or K_2 at 10 units each and raise the price of K_0 into 40 units; or increase the price of K_1 or K_2 into 20 units each and keeps the price of K_0 at 30 units. A user who want to buy the item needs to satisfy one of the authorisation attributes and requests only 2 blind decryptions according to his certificate.

If the items' pricing is managed as the suggestion described in subsection 3.4.3 and there are many options for each authorisation attribute, content provider cannot

figure out what item is being purchased. Thus, this mechanism perfectly preserves users' privacy. Additionally, authorised users are discouraged to share their certificate with other users as using a certificate means debiting the associated account. Thus, this mechanism deters unauthorised buyers.

3.5.2 In the Anonymous Cash Scheme

Preserving privacy is much easier in the anonymous cash scheme as users can purchase an item anonymously. However, if purchasing an item needs a buyer authorisation, how can content provider verify the buyer? To purchase a restricted item, a user can submit his anonymous cash along with an appropriate authorisation certificate, but how to guarantee that the user is the certificate holder?

An approach to prevent authorised users from sharing their certificate with an unauthorised person is by combining anonymous cash with the scenario in the blind decryption scheme. Recall the illustration in subsection 3.5.1. Content provider splits the content key m into three shares m_0, m_1 and m_2, where the last two shares associate with authorisation attributes. Users can purchase m_0 using anonymous cash, while requesting m_1 and m_2 in the same way as those in blind decryption schemes. This mechanism works as the encrypted key's shares can be decrypted (purchased) separately.

3.6 Comparative Evaluation

This section provides a comparison between our proposed schemes and some existing literatures.

An early privacy-preserving DRM concept that was proposed by Conrado *et al.* allows a user to anonymously purchase content and prevents the tracking of user when content is accessed [42]. The user's identity is not connected to the accessed content. This condition can be achieved using some temporary pseudonyms which

are managed by user' smart card. The main assumptions of this concept are that the user communicate to the content provider through anonymous channels, that the smart card does not disclose any personally identifiable information (PII), and that content can only be accessed by compliant devices.

Another privacy-preserving DRM schemes that were presented by Perlman *et al.* enable a user purchasing content anonymously or purchasing content without revealing which content it is about [144]. Perlman *et al.* proposed two schemes. The first scheme is based on anonymous cash. The distributor blindly signs cash that is requested by a user. The user then spends the cash to purchase content from the same distributor. The second scheme is based on blind decryption. A user ask the distributor to decrypt an encrypted content key. To prevent the distributor from learning which key is being requested, the user blinds the encrypted key before asking the decryption. Perlman's schemes do not have models that limit the number of executions of the content as the license cannot be checked each time the content is executed.

A recent privacy-preserving DRM that was applied for cloud computing allows for a license checking before every software execution [146]. In this scenario, a user who purchases software from the provider stay anonymous. At the same time, the software license is bound to the user and the validity is checked before execution. A software re-encryption scheme is implemented to make computing centers which execute the user's software unable to build the user's profile. Moreover, a combination of secret sharing and homomorphic encryption makes any malicious user unable to relay software to others.

Our proposed schemes improve Perlman's privacy-preserving DRM schemes. In the anonymous cash based scheme, a user firstly purchases tokens from the token provider through a non-anonymous channel. By this mechanism, the token provider can debit the user's account for the token he request. The user then spends the tokens to purchase content anonymously from the content provider. The token pro-

vider and the content provider can be a same party. At the end of transaction, the content provider will know which item is being purchased but will not know who is purchasing. In the blind decryption based scheme, a user firstly blinds an encrypted content key and then ask the content provider to decrypt the blinded encrypted key. By this mechanism, the content provider can identify who is purchasing but cannot figure out which item is being requested. Furthermore, we develop mechanisms that enable both schemes can be implemented to overcome varying price problem and buyer authorisation. In the case of restricted authorisation, we introduce authorisation certificates to prevent unauthorised buyers and to limit the number of content execution.

3.7 Chapter Remarks

The main goal of the users' privacy protection schemes is to enable users to purchase content so that the users' ID would not be connected to the purchased content's ID. To achieve this goal, either user's ID or content's ID should not be revealed to the content provider.

Anonymous cash scheme is likely to achieve the goal perfectly as users can purchase content anonymously; the content provider knows which content is being purchased, but cannot determine who is purchasing it. A problem that potentially threatens content provider's security in this scheme is double-spending. A user may spend a valid token for multiple purchasing. A trivial solution for this problem is by listing each token that has been spent, so that any double-spent token could be identified from the list. However, this mechanism needs the content provider to be online and the token provider to check the token in real time. Electronic cash scheme gives a better solution for double-spending problem. The scheme keeps the user anonymous, but once the user double spends his token, his ID will be revealed. However, while token provider can discover double-spenders, it can also frame inno-

cent users. To avoid the framing, user should have a digital signature scheme and a certified public key. The implementation of anonymous cash and electronic cash is less efficient as it needs two separate conversations: for purchasing token and for purchasing content.

Blind decryption scheme has more efficient implementation as it needs only one conversation for purchasing content. This scheme, however, potentially has privacy issues. As content provider does not know which content is being purchased, the only clue to what to charge user's account is the item's price. Consequently, varying prices can emerge as privacy issues. We propose to improve the existing scheme by providing a guideline to overcome this issue. Furthermore, in the case of restricted authorisation, we design authorisation certificates that make blind decryption scheme appropriate to prevent unauthorised buyers.

Chapter 4

Securing Content Delivery

4.1 Introduction

Secure delivery is urgently required in a content distribution system to guarantee that only authorised users can access the protected content and use it properly. Digital Rights Management (DRM) is a popular approach for this security requirement. Under this system, content is typically sent in an encrypted form along with the licence associated with it. At the users' side, an application processes the licence by means of a rights expression manager (REM), authenticates the users and decrypts the content using the corresponding decryption routine. The application can be implemented in hardware, such as in a set-top box for typical pay TV systems, or in software on the users' PC.

Trusted media players in most DRM applications contain the decryption key. The key must be kept secret and inaccessible to users, as finding the key would allow someone to decrypt and access content without restriction, thus defeating DRM protection. Unfortunately, trusted media players are often running on an untrusted platform. Although encryption algorithms used by most DRM application, such as the data encryption standard (DES) and the advanced encryption standard (AES), are believed to be secure, executing them in an insecure environment may allow

adversaries to compromise the system and obtain information about the decryption key [171, 201, 202]. Thus, keeping the key from being accessible to users is a major challenge for the DRM systems.

Problem Statement

Consider the typical DRM system for content distribution, illustrated in Figure 2.2 and described in section 2.2. Most DRM systems make protected digital content available on their distributor servers. Users can obtain the protected content from the distributor channel and then request a licence containing the decryption key from the clearinghouse. Downloading content from the distributor's channel does not seriously threaten either content provider's security or users' privacy. Users may download content anonymously (and even freely), so that their identity would not be connected to the content they choose. However, without obtaining an appropriate decryption key, users cannot unlock the protected content.

In contrast, acquiring a licence from the clearinghouse causes serious concerns over security and privacy. If an eavesdropper steals a licence when a user requests it from the clearinghouse, revenue will be lost. This obviously threatens content provider's security. Additionally, users' personal information collected by the clearing house is not guaranteed to be kept secret, as the clearinghouse may send the users' data and viewing detail to marketing agencies. Thus, it threatens users' privacy.

To overcome the security and privacy issue in the DRM system, we consider two approaches — obfuscation and obliviousness. The obfuscation approach protects the secret key by making the implementation of the encryption algorithm unintelligible. This unintelligibility is expected to prevent any adversary from reverse-engineering or compromising the system. The obliviousness approach protects the key by splitting it into several shares. The key can only be reconstructed using a predetermined threshold. This condition makes users unable to access content beyond what they are supposed to access. Discussion on the obfuscation and obliviousness approaches

are provided in section 4.2 and 4.3, respectively. The main question in this study is how it is possible to secure content delivery while the system still preserves user's privacy?

4.2 Obfuscation Approach

Two common techniques for the obfuscation approach include code obfuscation and white-box cryptography. Code obfuscation modifies the program that is used to implement the encryption algorithm, while white-box cryptography reforms the encryption function as a composition of some functions. These mechanisms are intended to make the implementation of the encryption algorithm unintelligible. In this section, we describe the theory and practice of code obfuscation and white-box cryptography, examine their feasibility in DRM applications, and construct a white-box implementation protocol to improve DRM for content distribution systems.

4.2.1 Code Obfuscation and Its Feasibility

Code obfuscation is an obfuscation technique intended to protect software implementation. In this technique, the program (code) used to implement the algorithm is rewritten in such a way that certain characteristics of the original program are hidden and unintelligible. This mechanism is expected to prevent the program from being reverse engineered.

In practice, obfuscation of a program (code) is applied to the variables used in the program. Variable names are scrambled, and data that were stored in a single variable are split into multiple variables and recombined at the execution time. This mechanism makes a code difficult for humans to understand, which is effective for hiding an algorithm and protecting the code, but not for any encryption key used by the code. Additionally, code obfuscation is often integrated with code flattening. In code flattening, extra paths are introduced in the program structure. This technique

makes the program difficult to analyse. However, it can be reverse engineered [171] and, thus, fails to achieve the main goal of code obfuscation.

Theoretical researches on code obfuscation are many, but there are fewer on implementations. Most of the researches produce conceptual decisions whether or not the obfuscator of a particular program exists according to a certain definition. However, none provides a real example on how to obfuscate a program if such an obfuscator provable exists. Practical obfuscation, on the other hand, has less theoretic foundation. Because of the fact of the practical obfuscation, the result of the theoretical aspect, and the lack of a bridge that connects theoretical and practical aspects, code obfuscation is less applicable in the DRM implementation.

4.2.2 White-box Cryptography

White-box cryptography (WBC) is an obfuscation technique intended to protect secret keys from being disclosed in a software implementation. The term "white-box" relates to the attack model that is applied to examine the security of this protection mechanism. Unlike the traditional cryptographic threat model, black-box, which assumes that attackers can only observe the input and output of the algorithm, the white-box model assumes that the attackers have full control over the whole operation and can freely observe dynamic code execution. Despite providing a full transparent methodology, WBC integrates the cipher in such a way that it does not reveal the secret key. This mechanism is more appropriate for DRM applications which are often executed in an insecure environment.

The basic notion of white-box implementations (see Figure 4.1) is to rewrite a key so that all information related to the key is hidden. The key is obfuscated using external encoding so that the encryption and decryption software requires encoded inputs and produces encoded outputs. This encoding mechanism can be done by replacing the encryption key E_k with the composition $E'_k = G \circ E_k \circ F^{-1}$. Input encoding key F and output decoding key G^{-1} must not be computed in the

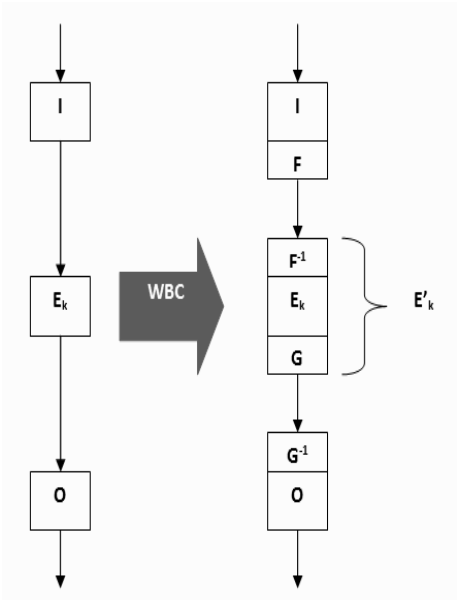


Figure 4.1: *Basic notion of white box implementation [202]*

same platform that computes E'_k, so that the white-box implementation cannot be used to compute E_k. This means that encoding input and decoding output have to be kept secret. At this point, white-box implementation cannot stand alone; it should be used in conjunction with other techniques to provide protection against key recovery attacks [96]. The white-box scenario is not standard, but useful for many DRM implementations.

Security and Feasibility of White-box Cryptography

Security of white-box implementation is relative; there is no system that is absolutely secure. A system is secure relative to a security model which may depend on an adversary's goal and the resources that can be accessed by the adversary [96]. In the white-box scenario, it is much more difficult to determine the resources of an attacker as they are endless. The best effort for achieving security is to prevent all known relevant threats in an effective way. A secure protection, for example, can be achieved by combining the effect of the secret key with some implementation specific data using a mathematical operation that is extremely hard to invert [169]. This mechanism allows constructing a system that operates similarly to the asymmetric encryption

algorithm, with a performance level close to the symmetric algorithm [168]. The security also depends on the implementation — a strong cryptographic algorithm is not necessary for a poor implementation.

WBC reflects more the reality than code obfuscation. Of the protection techniques applied in the DRM implementations, white-box cryptography is suggested to be the most effective protection in the DRM applications [171]. Currently, WBC is being used in real-world applications. Several commercial companies such as Microsoft, Apple and Sony have announced or deployed white-box techniques. Although many cryptanalysis techniques have been published, so far in real-world products, there has been no white-box implementation that has suffered from a key extraction attack. Instead, attackers focus on other parts of the system and exploit the cryptographic functionality. According to the result of some public challenges on weak white-box implementations, breaking white-box implementations in practice is difficult and time consuming [202]. The attacks depend on the construction of the white-box implementation and the properties of the underlying cipher. Therefore, broadly applicable attacks are difficult to deploy.

Despite the robustness of practical white-box implementations, performance, memory size and security are still the main concerns for current applications. Low performance and high-consumed memory size limit the application of WBC, especially for mobile devices. Although no attack on commercial white-box implementations has been found, it does not exclude the possibility of successful attacks in future. Additionally, effectiveness of the white-box implementation is limited when attackers can observe the execution of the DRM program. Therefore, it is not enough to only protect an application against key extraction, the application must also be hard to invert.

4.2.3 White-box Implementation Scheme to Improve DRM

The white-box implementation scheme presented in this subsection improves DRM with the aid of smart card's advantages [154]. A smart card contains an embedded microprocessor so that it can be used not only to store data, but also to process the data [33]. Since a smart card carries both processing power and information, it does not need access to a remote database at the time of a transaction. A smart card may contain programs and mobile databases that can be modified, updated or deleted through embedded program functions. The microprocessor is also used for security purposes. Data are never directly available to the external applications as the microprocessor controls data handling and memory access according to a given set of conditions.

White-box implementation works by obfuscating the original encryption and decryption keys. Suppose E is the encryption key. Two random keys F and G are generated to obfuscate E. Instead of using E, cipher text X' is obtained by encrypting content X using diffused key $G^{-1}(GEF^{-1})F$. Composition (GEF^{-1}) is called the internal key, while G^{-1} and F are the external keys. The provider then passes the protected content along with the internal key to the distributor and corresponding external keys to the smart cards manufacturer. The smart card is used to store the external keys and the original encryption key, compute their inverse and do external encoding-decoding. The manufacturer produces smart cards and delivers them to the distributor that will make them available to purchase.

A model of the smart card needs to be defined to make the scheme works. In this model, a smart card memory has two parts — an accessible area and an inaccessible area (see Figure 4.2). User's device can only communicate to the former part but cannot approach the latter. Data structure and mobile database are stored in the inaccessible area. This area is used to keep the external keys (G^{-1} and F) and the original encryption key (E) inaccessible. These items can only be accessed by encoding and decoding mechanisms defined in the accessible area. The inverse of

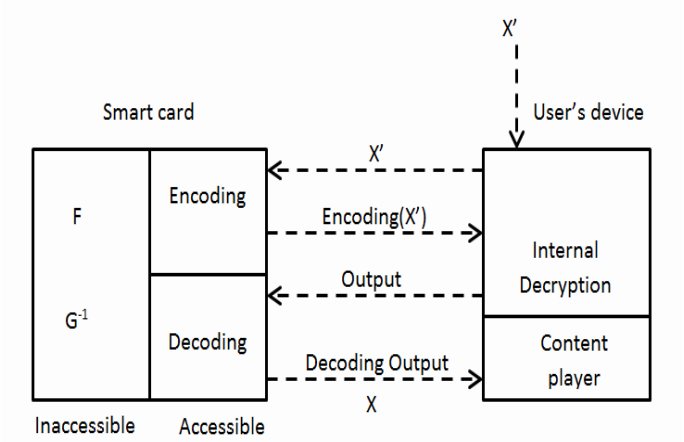


Figure 4.2: *Smart card model and decryption protocol*

G^{-1} is computed prior to encoding the input of the internal decryption, while the inverse of F and E are computed prior to decoding the internal decryption output.

A user can purchase content and its corresponding smart card from the distributor. To unlock the protected content, the user's device must be connected to a compatible card reader. User's device has two functions: it runs the internal decryption and plays the decrypted content. The decryption protocol involves communication between user's device and the smart card (see protocol 4.1 and the illustration at Figure 4.2).

Protocol 4.1: Decryption content in a WB implementation

1. Upon receiving the encrypted content X', user's device records its internal key and passes X' to the smart card.
2. The smart card encodes the cipher text X' using the key G, and sends $G(X')$ as an input for the internal decryption in the user' device.
3. The device uses the inverse of the internal key to decrypt the input and sends the output $FE^{-1}G^{-1}(G(X'))$ back to the smart card.
4. The smart card decodes this output using the key F^{-1} and then applies E^{-1} to obtain the content X. The content X is now playable to the user's device.

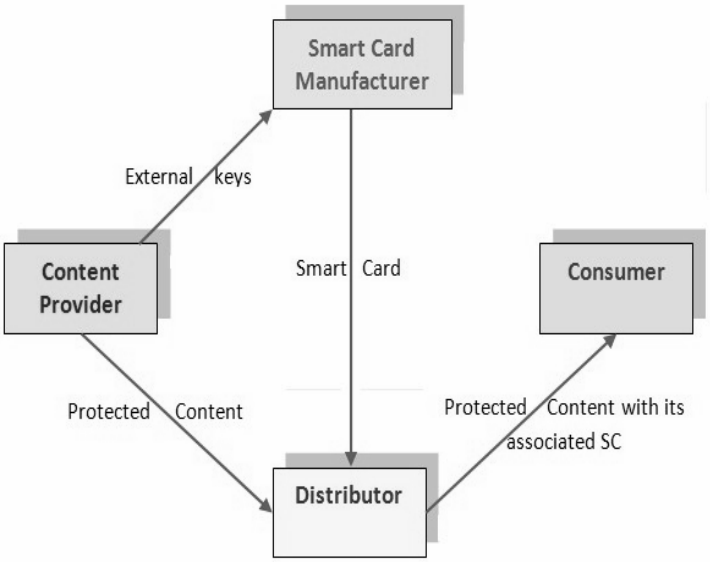


Figure 4.3: *Improved DRM system*

4.2.4 Security and Privacy Analysis

The model of improved DRM for content distribution is illustrated in Figure 4.3. Instead of a clearing house, the system employs a smart card manufacturer. This mechanism makes the system more efficient. Users can obtain content and its corresponding smart card from one party, i.e. the distributor. Assuming the smart card is a tamper-proof device, security and privacy of the scheme can be analysed as follow.

Security

In this mechanism, digital content is protected and the decryption key is hidden behind the obfuscated encryption key. The essential keys (external keys) to reveal the secret key are stored inside the inaccessible area of the smart card's memory. Knowing only the internal key is not adequate to unlock the content, as the internal decryption mechanism has to collaborate with external encoding-decoding operations which are undertaken inside the smart card. As a result, the user can only access the content that corresponds to the smart card he purchased.

Privacy

To unlock the protected content, a user does not need to provide his personal data for the licence. Instead, he can purchase the corresponding smart card anonymously. The content and its associated smart card will not be connected to the user's identity. Therefore, the user's privacy is protected.

4.2.5 Applications of the White-box Implementation Scheme

The advantage of a smart card helps white-box implementation to achieve security for content provider and preserve privacy for the users in a content distribution system. The white-box implementation scheme can be applied in both off-line and online business scenarios.

Off-line Content Distribution Scenario

Content provider encrypts digital content and passes the protected content to the distributor and all usage rules to the smart card manufacturer. The usage rules contain the external keys (G^{-1} and F), the original encryption key (E), all mechanisms on how and when these function can be retrieved, and external encoding-decoding function that has to be performed by the corresponding smart card. Once smart cards are completed, the manufacturer sends them to the distributor.

The distributor is typically an off-line retailer. The distributed content could be digital movies or songs. The retailer wraps the protected content (stored in mass storage devices such as CD's or DVDs) along with its corresponding smart card, and then makes them available to purchase. Users can purchase this package anonymously from the shop. This means that the distributor will not record the users' identity nor connect it to the purchased item. Thus, the users can privately play back the content on their smart card equipped players or computers connected with a smart card reader.

Online Content Distribution Scenario

In the context of an online content distribution scenario, the role of smart cards could be filled by a secure distributor server. The external and the original encryption keys are stored securely in the server. The distributor provides the protected content online and available to download. Users could download the protected content anonymously (and also maybe freely), but they cannot unlock the content unless they purchase *passing codes*. Two pass codes have to be used to unlock protected content. These codes are the outputs of the external encoding and decoding mechanisms which are undertaken inside the distributor server.

To keep anonymity, the purchasing *passing codes* can be done using an *electronic cash* scheme [32]. Before requesting *passing codes*, a user has to purchase adequate unit tokens from the distributor. The tokens can be used to purchase multiple items. Assuming that a user has downloaded protected content, the online content decryption is then performed through protocol 4.2. With this scenario, users can do an online transaction anonymously. Thus, while the content provider can securely distribute the protected assets, the users' privacy is also preserved.

The term of privacy protection in these applications, especially in the off-line scenario, is confined to the fact that the users' identity is not officially recorded nor connected to the purchased item. However, in real practice, an off-line retailer will know who is purchasing which item, as the customer directly comes to the shop. The privacy can be perfectly protected if the content provider has a package containing N protected items and the user is allowed to opt K out of these N items. In this case, the user must not be able to access more than K items and the content provider must not be able to determine which items are selected by the user. This scenario is known as the *oblivious transfer* concept.

Protocol 4.2: Online decryption

1. At the time of acquiring the passing codes, the user has to submit the downloaded content's ID (it could be the serial number) and an adequate amount of tokens.
2. After verifying the payment, the distributor server performs external encoding according to content's ID submitted by the user. This process outputs the first passing code.
3. This passing code enables user's player to partially decrypt the content. The user then sends the output of this decryption to the server.
4. The server performs external decoding based on the partial decryption's output submitted by the user. This decoding process yields the second passing code.
5. Finally, the second passing code allows the user's player to fully decrypt and play the content.

4.2.6 Comparative Evaluation

This subsection compares our proposed White-box implementation scheme to existing obfuscation approach literatures. The literatures include two common techniques — code obfuscation and white-box cryptography.

Code obfuscation is intended to protect software implementation. Theoretical study of the software protection was initiated by Goldreich and Ostrovsky [70], who provided a hardware-based theoretical treatment. This study motivated the emergence of code obfuscation ideas. The first contribution for a formalization of code obfuscation was provided by Hada [79], who presented a notion of obfuscation based on the simulation paradigm for zero knowledge. However, the formal definition of obfuscation was initiated by Barak *et al.* [14]. According to their definition, a probabilistic algorithm $\mathcal{O}$ is an obfuscator of a program P if it satisfies:

- **Functionality**. $\mathcal{O}(P)$ is a program that computes the same function as P.
- **Virtual Black Box Property (VBBP)**. Anything that can be efficiently computed from $\mathcal{O}(P)$, can be efficiently computed given oracle access to P.

Obfuscation, however, is hard to achieve. Barak *et al.* [14] showed there exist some predicates $\pi : \mathcal{F} \rightarrow \{0,1\}$ that can be computed efficiently when having access to an obfuscated implementation $\mathcal{O}(f)$ of $f \in \mathcal{F}$, but, given oracle access to f, no efficient algorithm can compute $\pi(f)$ much better than by random guessing. As a result, a generic obfuscator, i.e. an obfuscator that protect any given program, does not exist.

The first positive results in code obfuscation referred to the set of point functions as the obfuscatable family [110]. A point function, $\{f_\alpha\}$, is defined as $f_\alpha(x) = 1$ if $x = \alpha$ and $f_\alpha(x) = 0$ otherwise. Point functions can be obfuscated by random oracles $\mathcal{R}$ because the output of a random oracle hides all information about the input that produced it. The obfuscation of the point function f_α is defined as follows: $\mathcal{O}(f_\alpha)(x) = 1$ if and only if $\mathcal{R}(x) = \alpha'$ and 0 otherwise, where $\alpha' = \mathcal{R}(\alpha)$. The use of random oracles for obfuscation was motivated by the expectation that given access to an idealized building block, it would be feasible to obfuscate some functions. However, the existence of the idealized block allows the construction of a natural class of functions that are impossible to obfuscate and programable random oracles, in practice, are difficult to realize [72]. Moreover, under cryptographic assumptions, obfuscators of point functions with multi bit output can be constructed without a random oracle [29]. A best-possible obfuscation may not hide all information [72]. An obfuscated code may leak as little information as any other code, meaning that any information that is not hidden by the obfuscated code is also not hidden by another program with the same size and functionality.

Other positive results of code obfuscation were also applied to cryptographic primitives. First of all, the simulation-based obfuscation [83], which allows obfuscating point function, converting secret-key cryptography into public-key cryptography and transforming message authentication codes (MAC). The obfuscation of an indistinguishability under chosen plaintext attack (IND-CPA) secure symmetric encryption scheme results an IND-CPA secure asymmetric scheme. Similar results hold for

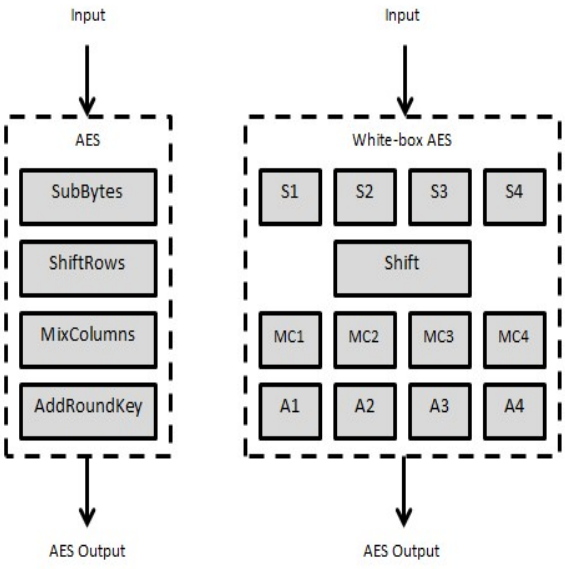


Figure 4.4: *Comparison of Standard AES and White-box AES [171]*

the obfuscation of MAC algorithms into digital signature schemes. However, these results do not apply to indistinguishability under chosen ciphertext attack (IND-CCA) secure schemes. Another positive obfuscation for traditional cryptography was applied to widely used re-encryption functionality [84]. This functionality takes a ciphertext for message m encrypted under a party's public key, and transforms it into a ciphertext for the same message encrypted under the other party's public key.

Because of the lack of implementation, code obfuscation is less applicable in the DRM applications than white-box cryptography. The first introduced white-box implementations were applied to the DES and AES [37, 38]. The Advance Encryption Standard (AES) consists of N_r rounds; where $N_r = 10$ for $AES - 128$. A basic round has four parts: `SubBytes`, `ShiftRows`, `MixColumns` and `AddRoundKey`. An `AddRoundKey` operation occurs before the first round and the `MixColumns` operation is omitted in the final round.

The general notion of the white-box AES implementation is to merge several steps of the cipher into a network of lookup tables and to obfuscate the results using random input-output encoding [38]. Thus, instead of AES_k, the white-box AES implements $N_O \circ M_G \circ AES_K \circ M_F \circ N_I$. M_F and M_G are the affine input and output

Table 4.1: Size and performance comparison [38, 204]

Description	standard AES	white-box AES
Memory size	4352 bytes	770048 bytes
Operation	300 lookups and xors	3104 lookups
Encrypt 1 MB data	< 0.5 seconds	> 3 seconds

respectively, and N_I and N_O are the input and output non-linear nibble encodings respectively. Comparison of standard AES and white-box AES (see Figure 4.4) shows that to obfuscate the AES algorithm, each of its parts is diffused into several lookup tables. The sequences of lookup tables S1 to S4, MC1 to MC4 and A1 to A4 are functionally equivalent to SubBytes, MixColumns and AddRoundKey, respectively. The Shift operation, however, is not undertaken as a lookup table due to its nature.

White-box AES implementation provides secure protection of a secret key in cryptographic module, but has a slow performance. The comparison of the size and performance between standard AES and white-box AES implementation is visualized in Table 4.1.

White-box AES implementation has been broken using an algebraic cryptanalysis technique, with the worst time complexity of 2^{30} [18]. Nevertheless, the implementation can still provide effective key protection by increasing the attacks' time complexity. One technique for increasing this complexity is Medusa [114], a software tamper resistance technique which makes a binary program code tamper resistant by incorporating the code into the key of a white-box implementation. This technique increases the integrity of the white-box implementation: if an adversary modified the implementation, the results will be wrong. As a result, the attack time complexity can be much higher. In response to the algebraic cryptanalysis, a new construction beyond the lookup table strategy, called perturbations strategy [25], was introduced. This strategy is an improvement of the traceable block cipher scheme [17]. However, this construction was proven to be insecure by De Mulder *et al* [120], and there has

been no improvement made for the perturbations strategy up to now.

The most recent improved scheme is proposed by Yoo *et al* [204]. To address the performance problems, Yoo *et al* combine white-box AES and standard AES; they used a white-box AES only once at the beginning, and then applied standard AES to the rest of the scheme. They claimed that their scheme has the same performance compared to standard AES and is robust enough to withstand white-box attack. This claim, however, needs to be criticized. If white-box is only implemented in the first round and the subsequent rounds remain in standard AES, the scheme will be exposed to white-box attacks just as the standard one because additions with round key information can easily be distinguished. The scheme may be secure to withstand black-box attacks, but may fail to preserve security against software attacks.

Our proposed white-box implementation scheme directly adopts the basic notion presented in Figure 4.1. For security purposes the content key is obfuscated by a composition of internal and external keys which are stored separately. The external keys are essential — without these keys, the content decryption key is hard to reconstruct. To keep the external keys secret, they are stored inside an inaccessible area of a smart card's memory. With this mechanism, our proposed scheme requires a simpler computation than many white-box implementation proposed in the literatures. The security of our scheme relies on the physical security of smart cards. To preserve users' privacy, the smart card can be purchased anonymously. This mechanism makes the scheme achieve content provider's security and users' privacy simultaneously.

The white-box implementation is likely to be integrated with the oblivious transfer scenario to achieve perfect security and privacy in a content distribution system. The white-box implementation is utilized to secure encryption to withstand reverse engineering, while oblivious transfer concept is employed to construct a content distribution scheme.

4.3 Obliviousness Approach

Oblivious Transfer (OT) is a cryptographic protocol that allows two parties to privately exchange one or more secret messages. An OT protocol has to be set up in such a way that it will achieve security for the sender and privacy for the receiver [66, 68]. The former means that the receiver will not be able to learn more than he was supposed to learn. The latter means that the sender will not know what the receiver has learned.

The first OT protocol, introduced by Rabin [158], was intended to overcome the exchange of secrets (EOS) problem. This protocol enables a sender to deliver a message to a receiver in such a way that the receiver can access the message with probability 1/2 and the sender will not know whether the message was received. Rabin's protocol was then generalized to the OT_1^2 [54]. In the OT_1^2 protocol, the sender has two secret messages and the receiver wishes to learn one of them. At the end of the protocol, the sender does not know which message was chosen while the receiver knows nothing of the unselected message. This scheme has been studied extensively and generalised to a wide variety of models including OT_1^N [125, 189, 190] and OT_K^N [126, 39]. In the OT_1^N and OT_K^N protocols, the sender has N secret messages and the receiver would like to learn one or K out of them, respectively.

The oblivious content distribution scheme constructed in this section makes use the concepts of a secret sharing scheme and one-time program.

4.3.1 Secret Sharing Scheme

Secret sharing scheme refers to a mechanism for distributing a *secret* amongst a group of participants. The actors in this scheme include one *dealer* and N *participants*. The dealer has a secret and splits it into N shares and distributes the shares to all participants. Each participant receives one particular share. Any group of t participants or more can together reconstruct the secret but no group of fewer

than t players can do so. Such a system is called a (t, N)-threshold scheme, with parameter t.

In the Shamir (t, N)-threshold scheme [174], initially the dealer D chooses N distinct non-zero elements of $\mathbb{Z}_p = \{0, 1, 2, 3, ..., p-1\}$, denoted x_i, where $1 \leq i \leq N$. This scheme requires that $p \geq N + 1$. D then gives the value x_i to the participant P_i, for $1 \leq i \leq N$. The values x_i are public. When D wants to share a secret key $S \in \mathbb{Z}_p$, D secretly chooses, independently at random, $t-1$ elements of $\mathbb{Z}_p$, denoted $a_1, a_2, ..., a_{t-1}$. D then computes $y_i = a(x_i)$, where $1 \leq i \leq N$ and

$$a(x) = S + \sum_{j=1}^{t-1} a_j x^j \bmod p \tag{4.1}$$

D finally gives the shares y_i to participant P_i, for $1 \leq i \leq N$. Thus, each participant P_i receives a pairs of (x_i, y_i).

To be able to reconstruct the secret, S, all possible x_i, that is all non-zero elements of $\mathbb{Z}_p$, are required to have each multiplication inverse in $\mathbb{Z}_p$. Thus, p must be a prime number. An approach to compute S is by means of the Lagrange interpolation [179]:

$$S = \sum_{j=1}^{t} y_{i_j} \prod_{1 \leq k \leq t, k \neq j} \frac{x_{i_k}}{x_{i_k} - x_{i_j}} \tag{4.2}$$

Suppose

$$b_j = \prod_{1 \leq k \leq t, k \neq j} \frac{x_{i_k}}{x_{i_k} - x_{i_j}}$$

for $1 \leq j \leq t$, then

$$S = \sum_{j=1}^{t} b_j y_{i_j}.$$

Hence, the key S is a linear combination of the t shares.

In the case when $N = t$, the protocol of (t, t)-threshold scheme in $\mathbb{Z}_m$ is as follows. Suppose the dealer D wants to share the secret, S, to t participants. Initially, D secretly and independently determines at random $t - 1$ elements of $\mathbb{Z}_m$, denoted

$y_1, y_2, ..., y_{N-1}$. D then computes the t-th y using the equation:

$$y_t = S - \sum_{i=1}^{t-1} y_i \bmod m \tag{4.3}$$

and finally gives the share y_i to participant P_i, for $1 \leq i \leq t$.

To reveal the secret, S, all those t participants have to combine their shares together and reconstruct S using the formula [179]:

$$S = \sum_{i=1}^{t} y_i \bmod m \tag{4.4}$$

It is clear that any group of $t-1$ participants or fewer cannot reconstruct S as they receive $t-1$ independent random numbers as their shares. Those $t-1$ participants may sum their shares, i.e

$$y_1 + y_2 + y_3 + ... + y_{t-1}$$

and yield

$$S - y_t.$$

However, they have no information about the random value y_t, and hence, unsuccessfully reveal the secret, S. Unlike the (t, N)-threshold scheme, the (t, t)-threshold scheme does not require m to be a prime number as all elements of Z_m have each addition inverse necessary for reconstructing S.

4.3.2 One-time Program

The one-time program (OTP) is a new computational paradigm for supporting security applications [71]. An OTP can be executed once on a single input and then self destructs. Such a program can be extended to a k-time program which can be evaluated for k times and then self destructs. The program acts as a black-box as there is nothing about the program that can be leaked except the computation result. To achieve this functionality, however, an OTP cannot be solely software ba-

sed, as any software, in principle, can be copied and run more than once. One-time functionality can be achieved by means of secure hardware devices, namely one-time memory (OTM).

An OTM is a memory device which is initialed with two keys (k_0, k_1) and a tamper-proof bit (TPB) set to 0. Upon receiving a single bit input $b \in \{0, 1\}$, OTM verifies whether TPB = 0. If so, OTM sets TPB to 1 and outputs k_b, otherwise it outputs an error symbol. OTM outputs one of its initial keys and the other key is irretrievably lost. The security assumptions from the OTM device include: (1) memory locations that are not accessed by the device, will not leak via a side channel, whereas a memory cell that is accessed may be immediately leaked; (2) the single bit b is tamper-proof, but is readable. OTM enables OTP to be executed for a limited number of times. The OTP is guaranteed not to be reverse-engineered and the components that restrict the number of execution cannot be removed [71]. This characteristic ensures that the OTP mechanism deals effectively with copy protection and software protection problems.

4.3.3 Oblivious Content Distribution Scheme

The oblivious content distribution scheme presented in this subsection is more flexible and appropriate for DRM implementation [155]. This scheme utilises tamper-proof devices. A tamper-proof device means any device that can be used only in a particular way, otherwise the device will be corrupted and its content will no longer be accessible. Utilising tamper-proof devices in this scheme is less expensive. The device contains only two types of functions, `GetKey` and `GetContent`. `GetKey` allows the user to ask for a shared key, while `GetContent` requires an authorized key to reveal the message stored in it. With these characteristics the device can be mass produced at a low cost. Creating a single device containing all pairs of functions (`GetKey`, `GetContent`) may be reasonable and more efficient. However, for the sake of clarity in this subsection, one device is assumed to contain a pair of functions

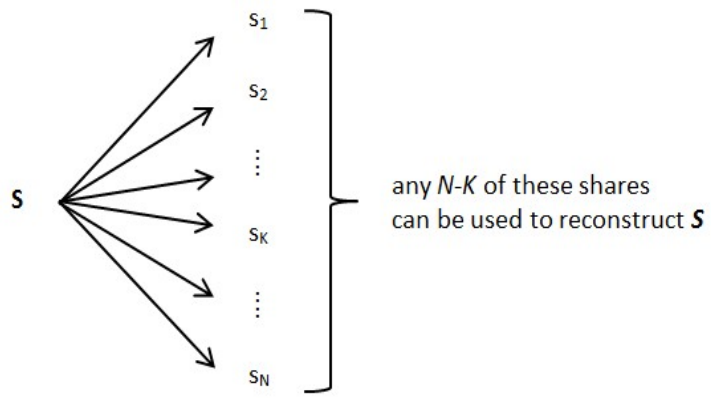


Figure 4.5: $N - K$ *secret sharing schemes*

(`GetKey`, `GetContent`).

The scheme allows content provider to deliver contents to user in such a way that at the end of the scheme the user cannot access contents more than he is supposed to access and the content provider will not know which contents are accessed by the user. Suppose the content provider (say, Alice) provides N content items (e.g. movies), $M_1, ..., M_N$, and the customer (say, Bob) wishes to access K out of N items. Alice has a secret key S to access the items, and utilises Shamir's secret sharing schemes [174], with the threshold parameter $N - K$, to share the secret. That is, she splits the secret into N pieces such that any set of at least $N - K$ shares can reconstruct the secret (see Figure 4.5). To share the secret and send the contents, Alice performs protocol 4.3.

Protocol 4.3: Sharing the secret key and sending the content

1. Alice secretly chooses random $N-K-1$ elements of $\mathbb{Z}_p$, denoted $a_1, ..., a_{N-K-1}$ and forms the polynomial $f(x) = S + a_1x^1 + ... + a_{N-K-1}x^{N-K-1}$. Note that p is a prime and $p > N$.
2. For $i = 1, ..., N$, she computes s_i, where $s_i = f(i) \bmod p$.
3. Alice loads device d_i with s_i as the key value, and M_i as the content value.
4. Alice gives all devices to Bob.

After delivering the devices there is no subsequent communication between Alice

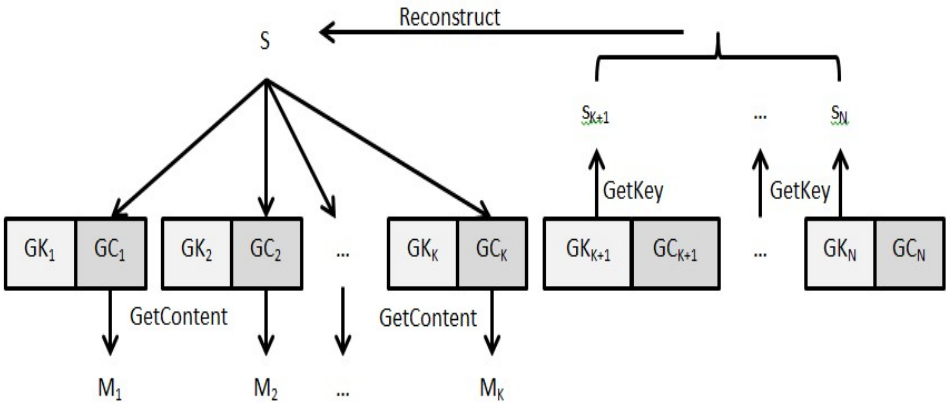


Figure 4.6: *Process of obtaining K out of N contents.*

and Bob. Bob can access K content items if he accepts sacrificing $N-K$ items that are not supposed to be accessed. This condition applies with the assumption that once a device is executed, it will be corrupted or will destroy itself. This assumption is motivated by the notion of one-time program. To obtain K content items, Bob performs protocol 4.4 (see also Figure 4.6 for the illustration).

Protocol 4.4: Getting K out of N items

1. For simplicity, assume that K items Bob wants to access are $M_1, ..., M_K$. Bob then performs the *GetKey* function on the devices $d_{K+1}, ..., d_N$ (namely $GK_{K+1}, ..., GK_N$), to obtain $N-K$ shares.
2. With the $N-K$ shares, $s_{K+1}, ..., s_N$, Bob can reconstruct the polynomial, e.g. using the Lagrange interpolation over field $\mathbb{Z}_p$, and learn the secret S.
3. Using the access code S, Bob performs the *GetContent* functions on devices $d_1, ..., d_K$ (namely $GC_1, ..., GC_K$) to obtain the items $M_1, ..., M_K$.

The protocol 4.4 can be modified to cover another need. For instance, instead of focusing on the *number-of-items* variable, the protocol can be applied to the *number-of-plays* variable (e.g. a customer wants to watch a movie for K times). The movie provider then sends the customer a package containing N pairs (`GetKey`, `GetContent`) with all `GetContent` functions associated with a single movie.

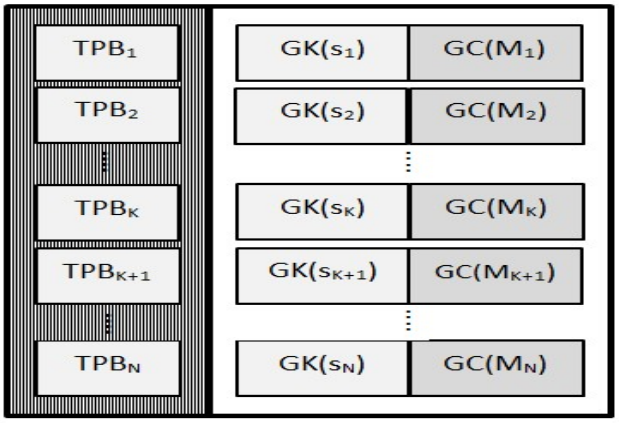


Figure 4.7: *Smart card model; GK and GC stand for GetKey and GetContent, respectively.*

4.3.4 Implementation of Oblivious Distribution to Improve DRM

Smart cards are employed to implement the developed scheme in the DRM applications. Content usage rules are properly concealed inside smart cards. A smart card is self-protective and self-updating. Access to usage rules stored in the smart card are restricted to functions available to the user. The content provider can design these functions in such a way to protect the digital assets accordingly. Additionally, a smart card can update usage statistics in its mobile database for every single content use to avoid disproportionate use.

To begin with, suppose the content provider provides N content items, $M_1, ..., M_N$ and encrypts all items using a secret key S. For a particular value K, $1 \leq K \leq N-1$, S is split into N shares, $s_1, , s_N$, using Shamir's scheme [174] with the threshold parameter $N-K$. The content provider then bundles all encrypted items into an item package and passes the package to the distributor and the key's shares to the smart card (SC) manufacturer.

The SC manufacturer creates smart cards and sends them to the distributor. A user can obtain the item package from the distributor's channel and purchase an appropriate smart card. To access the protected items, the user's player must be connected to a compatible smart card reader. A K-valued smart card can be used

to unlock K selected items and deny access to $N - K$ unselected items.

The smart card model (see Figure 4.7) has the following characteristics:

1. For a particular K, a smart card contains N pairs of functions (`GetKey`(s_i), `GetContent`(M_i)), where $i = 1, 2, , N$.

2. Only one function can be executed from each pair. That is, executing the function `GetKey`(s_i) will disable the associated function `GetContent`(M_i) and vice versa.

3. In concrete terms, the smart card executes $N-K$ `GetKey` functions associated with $N-K$ unselected items. The shares revealed by these functions are then combined to reconstruct the key S that be used to unlock K selected items.

Consider the 2nd characteristic: for each pair (`GetKey`(s_i), `GetContent`(M_i)), only one function can be executed. This characteristic is supported by one-time program (OTP) mechanism as follows. Each pair (`GetKey`(s_i), `GetContent`(M_i)) has an associated tamper-proof bit, TPB_i, assuming that the smart card memory has an inaccessible area to store all TPBs. This means that the TPB can only be changed using the internal smart card mechanism and remains inaccessible to any external device. The value of each TPB is set at most twice — at the time of smart card manufacture and when the associated function is executed. The smart card manufacturer sets the initial value of all TPBs to be 0. When a function is executed, the associated TPB is then set to be 1 (see algorithm 4.1). After one execution,

Algorithm 4.1 Run *GetKey* or *GetContent*

```
Input: a chosen function, GetKey or GetContent
Output: Value of executed function or error
BEGIN
if TPB = 0 then
    set TPB = 1 and execute the chosen function
else
    return error
end if
END
```

the pair ($\texttt{GetKey}(s_i), \texttt{GetContent}(M_i)$) is no longer executable as $TPB_i \neq 0$. Thus, only one function can be executed from this pair.

The first task of the smart card is to reconstruct the decryption key S. This task requires user to select K items. The decryption key is then reconstructed by combining shares associated with unselected items. Reconstructing the key is processed using algorithm 4.2.

Algorithm 4.2 Reconstruct the key S

```
Input: C = {i|M_i is selected item }
Output: the secret key S
BEGIN
if |C| = K then
    record C
    for i ∉ C do
        run GetKey(s_i)
    end for
    compute S from revealed shares s_i
    record S
else
    return "the number of selected items does not match"
end if
END
```

The indices of all selected items and the secret S are recorded in the inaccessible area and used as the input parameters of the `GetContent` functions. At this stage, the smart card is ready to retrieve the selected items using algorithm 4.3.

Algorithm 4.3 Unlock the selected items

```
Input: i
Output: M_i
BEGIN
while executable GetContent is available do
    if i ∈ C then
        run GetContent(M_i)
    else
        return "requested content is unaccessible"
    end if
end while
END
```

The user can obtain all selected items after which the smart card will expire.

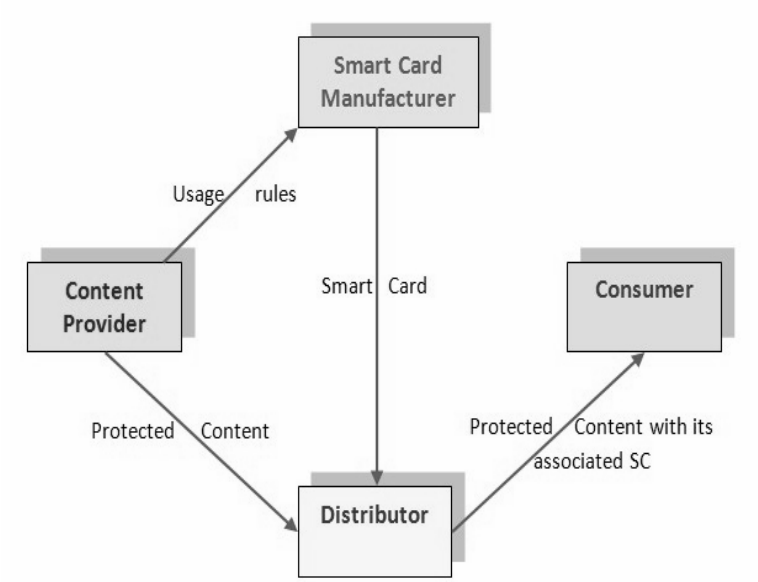


Figure 4.8: *Oblivious distribution implementation in the DRM system*

This scheme enables user to access content no more than he was supposed to access and at the end of the scheme, content provider cannot figure out which items have been retrieved. After decrypting items, user can play them for an unlimited time. Indeed, how many times user can play each item is beyond this scheme. If the number of plays is included in the restriction, however, the scheme can be advanced to cover more variables in one scheme.

4.3.5 Security and Privacy Analysis

The improved DRM model for content distribution (see Figure 4.8) provides an efficient mechanism. Instead of a clearing house, the system employs a smart card manufacturer. Users obtain the content and the corresponding licence (provided by an appropriate smart card) from one party, that is, the distributor. This mechanism makes the process more efficient. Furthermore, the improved system also achieves security and privacy for the content provider and the users, respectively. An analysis of both achievements follows.

Security

In the proposed scheme, the shares of the secret key and the function for accessing content are stored in tamper-proof devices. The user cannot access content without obtaining the secret key. The key, however, is split into several pieces of shares and distributed among the pairs of functions (`GetKey`,`GetContent`) inside the device, using Shamir's secret sharing scheme [174]. This scheme is secure because knowing less than a predetermined number of shares gives the user no way to reconstruct the secret. As a result, the user can only obtain the secret key if (and only if) he sacrifices all items that he is not supposed to access. Additionally, the reconstructed secret key and the identity of all selected items are recorded within the inaccessible area, meaning they are kept from disclosure to user. This mechanism ensures the secret key can only be accessed by `GetContent` functions associated with the selected items. This means that the user is not able to access anything other than the items that are supposed to be accessed. Therefore, the proposed protocol achieves perfect security for the content provider.

Privacy

In the proposed scheme, there is no interaction between content provider and user after the content provider gives all devices to the user. In the implementation context, the content provider, including the distributor, will not interact with the user after purchasing the item package and an appropriate smart card. Additionally, the smart card will expire after all selected items are retrieved. This means that all pairs of functions (`GetKey`,`GetContent`) are disabled at the end of the scheme. Therefore, the content provider cannot determine which items the user has accessed. Moreover, to unlock the protected items, user does not need to provide his personal data for the licence. Instead, he can purchase an appropriate smart card anonymously. The selected items and the smart card will not be connected to the user's identity. Therefore, the user's privacy is protected.

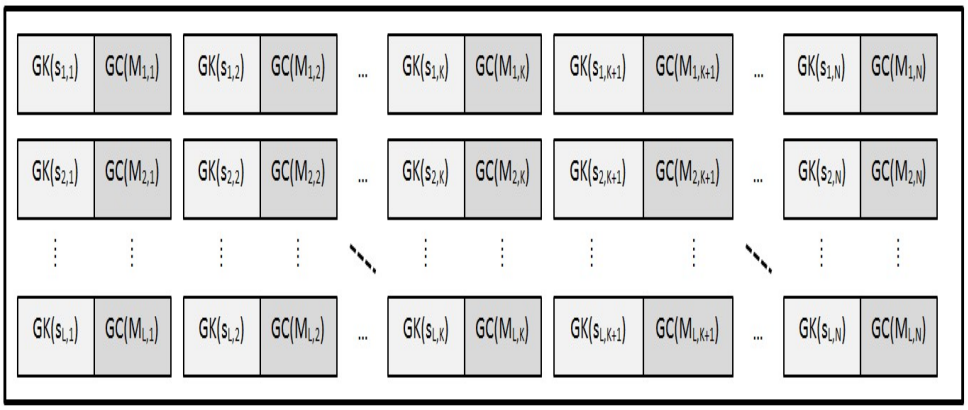


Figure 4.9: *Extended smart card model: GK and GC stand for GetKey and GetContent, respectively.*

4.3.6 Advanced Implementation

In the oblivious content distribution scheme described above, a user can decrypt a set of items no more than he was supposed to access. However, once a content item has been decrypted, the user can play it without limit. If the restriction of the number of plays is also considered in a business scheme, then an extra variable must be added to the content distribution scheme.

The proposed scheme can be enlarged to cover more variables of the usage rules. That is, the variable *number-of-items* and *number-of-plays* can be integrated in one scheme. For example, a user may purchase 5 items, namely content M_1, M_2, M_3, M_4, M_5, and 20 plays. In this case, the user can play all items, but no more than 20 times overall. He may play M_1 for 3 times, M_2 for 4 times, M_3 for 7 times, M_4 for 4 times and M_5 twice. However, he cannot play M_2 for 10 times and M_5 for 11 times.

The smart card model for the extended content distribution scheme is depicted at Figure 4.9. Suppose Alice has N items and Bob purchases K items and L plays. Smart card used to fulfill this need is called (L, K)-smart card. An (L, K)-smart card contains $L \times N$ pairs of functions (`GetKey`$(s_{i,j})$,`GetContent`$(M_{i,j})$) in a $L \times N$ matrix. For a particular j and $1 \leq i \leq L$, all $s_{i,j}$ and $M_{i,j}$ is associated with the key share s_j and the item M_j, respectively. The smart card has the following

characteristics:

1. As in the previous model, only one function can be executed from each pair.

2. The secret decryption key S can be obtained by executing some `GetKey` functions at the first play (i.e at the first row of the matrix). Once S is reconstructed, it can be used to decrypt other selected items at subsequent plays.

3. Executing `GetKey`$(s_{1,j})$ functions will disable associated `GetContent`$(M_{i,j})$ functions for all $1 \leq i \leq L$, and thus, disable user to access item M_j in all plays.

4. For each i, executing a `GetContent`$(M_{i,j})$ function, will disable all `GetContent`$(M_{i,h})$ functions, for $h \neq j$. This means that for each play user can only access one item.

To access K items and L plays, Bob has to perform protocol 4.5.

Protocol 4.5: Accessing K items in L plays

1. Bob determines K items he wants to access. For simplicity, without lost of generalisation, assume that K items Bob chooses are $M_1, ..., M_K$.
2. Smart card then executes $GetKey(s_{1,K+1})$, ..., $GetKey(s_{1,N})$ functions to obtain shares $s_{K+1}, , s_N$ and reconstruct the secret key S. The key is then used for all plays. These executions disable all $GetContent(M_{i,j})$ functions, for $1 \leq i \leq L$ and $K+1 \leq j \leq N$.
3. For each i, where $1 \leq i \leq L$, smart card can only execute one of K $GetContent(M_{i,j})$ functions, for $1 \leq j \leq K$

The characteristics of (L, K)-smart card and the extended scheme guarantee that the user can play all K items, but no more than L times overall. This advanced scheme provides flexible content distribution that still preserves security and privacy.

4.3.7 Applications of the OCD Scheme

The aid of the smart card is mainly intended to implement the oblivious content distribution (OCD) scheme in an off-line business model. However, the mechanism

of this implementation can also be adopted for an online content distribution system.

Off-line Scenario.

An off-line scenario means that there is no online interaction between content provider and user at the transaction time. Suppose content provider has N content items. All these items are firstly encrypted for security purposes. Content provider then computes the shares of the decryption key S according to the value K, where $1 \leq K \leq N-1$. For a particular S, there will be $N-1$ sets of N shares. Each set corresponds to a particular value of K. Content provider stores all protected items in mass storage devices such as CDs or DVDs. The provider then passes the storage devices to the distributor and all usage rules to the smart card manufacturer. The usage rules contain sets of shares used to reconstruct the secret decryption key.

The manufacturer creates smart cards with $N-1$ different values for a package of N items. A smart card of the value K, where $1 \leq K \leq N-1$, can be used to unlock K out of N items. Note that the value K can also be set to N, but in this case, the content provider can figure out that all items are likely to be chosen. Therefore, $K < N$ is more appropriate to preserve users' privacy. Once smart cards are completed, the manufacturer sends them to the distributor.

The distributor is typically an off-line retailer. Users can purchase a package of items and a corresponding smart card of desired value. The users can privately play back the items on their smart card equipped players or computers connected with a smart card reader. The items package can be purchased once for multiple smart cards purchasing. A business scenario could be set follows.

- **Without the number of play restriction.** At the basic OCD protocol, a smart card is used to unlock a particular number of items. There is no restriction on how many times user can play a selected item. For example, assume that a user has purchased a package of 5 movies, say $\{M_1, M_2, M_3, M_4, M_5\}$. If the user wants to play 2 of these movies, he needs to buy a smart card

of value 2. Once the selected movies are unlocked, they will be saved in the user's device and can be played unlimited times. If the user wants to unlock the other movies in the package, he needs to purchase a new appropriate smart card.

- **With the number of play restriction.** If the number of plays is taken into account, the scenario needs to use an (L, K)-extended smart card. This card can be use to play K items for no more that L times overall. In this case, the protocol must be set so that the access to the items package can only be undertaken through the smart card and any unlocked items cannot be saved in the user's device. Recall the previous example, if a user wants to play 3 movies for 10 times overall, he needs to buy a $(10, 3)$-smart card. All decryption and play will be executed through the smart card. If the package has been played 10 times, the user needs to purchase another smart card even for playing the same selected items.

Online Scenario.

The mechanism of the OCD scheme can be adopted in an online content distribution scenario. In this scenario, the smart card is substituted by a secure distributor server. All sets of the key's shares, `GetKey` and `GetContent` functions are stored and executed in the server. The distributor then makes all protected items available online.

To be able to access a 'clear' item, user has to firstly purchase a voucher from the distributor. Purchasing a voucher can be done online using a credit card and the voucher ID will be sent securely to the user's email address. The voucher's values can vary. A voucher of the value K can be used to purchase up to K items. If there is no restriction on how many times a selected item can be played, the item can be downloaded and saved in the user's device. In contrast, if the restriction is applied, the user cannot download and save the item, instead he has to play the item online.

With this scenario, while users can access content anonymously, the mechanism also preserves security for content provider.

4.3.8 Comparative Evaluation

Our proposed oblivious content distribution scheme is constructed based on the oblivious transfer (OT) concept. Approaches applied to any OT protocol are intended to achieve security and privacy simultaneously.

To achieve unconditional secrecy, Rivest proposed a protocol that utilises a trusted initializer [162]. At the initialisation step of this protocol, the trusted initialiser sends some information to both the sender and the receiver. The receiver then sends a request to the sender and the sender replies with some information. Utilising a trusted party, however, is unacceptable in the privacy preserving applications [127]. In these applications, the third party is not a trusted party, but is assumed not to collude with the sender. The privacy of the receiver is guaranteed as long as there is no collusion between the sender and the third party.

To omit the trusted party, Naor and Pinkas proposed a distributed oblivious transfer (DOT) in which the task of the sender is distributed among several servers [127]. That is, the two strings m_0 and m_1 are distributed among N servers in such a way that the receiver must interact with a predetermined number (k) of servers to obtain the chosen message m_σ, where $\sigma \in \{0, 1\}$. After receiving information from these k servers, the receiver cannot learn more than a single linear combination of m_0 and m_1, while a coalition of $k-1$ servers does not learn any information about σ. The security of the DOT protocols has been intensively studied [20, 43, 44, 67]. The DOT protocols are also aimed at overcoming the restriction in the availability of the secret message. In the previous OT protocols, if the unique sender is unavailable, the receiver cannot execute the protocols. Therefore, to increase the availability of messages, the sender needs to distribute them to several servers.

The efficiency of the system is also an important issue in the implementation

of an OT protocol. OT is unlikely to be based on more efficient one-way functions or other private-key cryptographic primitives [128]. As a result, all known OT protocols need public-key operations that are typically implemented using modular exponentiations, which are computationally intensive tasks.

Our oblivious content distribution scheme, described in the subsection 4.3.3, requires an efficient computation. In our approach, to achieve security the content key is split into several shares and can only be reconstructed using a predetermined number of shares. This mechanism ensures that users cannot access content they are not supposed to access. However, for privacy purposes, content providers will not know which content is selected by a user.

4.4 Chapter Remarks

White-box implementation and oblivious content distribution schemes have answered the main question of this study — how is it possible to secure content distribution and preserve users' privacy simultaneously? White-box cryptography is initially intended to secure content delivery by making the encryption implementation unintelligible. However, its implementation can be set such that it also preserves users' privacy. The oblivious content distribution scheme has a different story. This scheme is indeed constructed based on the oblivious transfer concept, which originally aims to achieve security and privacy in a communication system.

Both schemes employ a tamper-resistance device, such as smart card, to implement their protection mechanisms. White-box implementation is appropriate when the content provider has only one item to be protected. In this case, however, the users' privacy protection relies on the anonymity of purchasing a smart card. The users' privacy protection can be improved when the content provider provides several items the users can opt to use. Using the oblivious content distribution scheme, the users can access the items no more than they are supposed to access, while the

content provider will not know which items the users have chosen.

Chapter 5

Watermarking for Preserving Protection

5.1 Introduction

Once content reaches end-users, it will be decrypted and cryptography no longer protects it. There is an aspect of copy control that the encryption based DRM technology cannot afford, namely analog hole. This aspect is inevitable because to be perceptible to humans, all digital content must be converted into analog form. Analog content can always be duplicated and, thus, it is vulnerable to copyright infringement. Digital watermarking is viewed as a potential tool for preserving protection. A watermark, which is embedded into host content will remain associated with the content throughout its subsequent duplication and distribution. The watermark can later be detected or extracted to provide ownership evidence or copyright notification.

Watermarking algorithms have been implemented in all types of content — text, image, video and audio. Text watermarking might be undertaken by hiding watermark information in the document's layout [24, 85]; by substituting particular words with their synonyms [187]; or by zero-watermarking, that is, instead of di-

rectly altering the text document, the scheme uses the characteristic of the text to generate document's identification [89]. Watermarking for the last three content types has similar characteristics — audio watermarking has similar requirements as image watermarking and, for watermarking purposes, video is best considered as a sequence of still images [149]. Watermarking for these contents can be done in spatial and frequency domains. In spatial domain, watermark signal is directly embedded into host content signal. This watermarking process is relatively easy, but alters the host content during embedding process and has the lowest resistance to signal compression [181]. Frequency domain watermarking is carried out by altering the frequency value in the host content spectrum and is more robust and compatible with common signal compression standards. The discussion in this chapter mainly focuses on the digital image watermarking.

Research on digital watermarking was initiated in the early 1990's and the interest in this topic has started to increase since 1995 [47]. The increasing amount of watermarking research has been principally motivated by its essential applications in digital copyrights management and protection. Foremost among the applications are electronic distribution, transaction tracking, broadcast monitoring, authentication, tamper detection, copy protection and owner identification.

- Electronic distribution may apply imperceptible watermark for recipient validation and digital authentication, or perceptible watermark for a quick identification.

- Transaction tracking application inserts watermark into a host content to record transactions of a copy of the content.

- Broadcast monitoring applies watermarking to enable an authorised party to track when a specific program is being broadcast by a TV station.

- The authentication application needs either robust or fragile watermarks — information extracted from a robust watermark can be compared with the con-

tent features to evaluate its integrity, while the absence of a fragile watermark is an indication that some unauthorised changes have been applied over the content.

- Tamper detection enables further investigation by identifying the tampered parts within content.

- Copy protection mechanism employs watermarking to bear copy control information (CCI) which indicates the copying status of content — copy freely, copy once and copy never [46].

- Owner identification application uses watermarking to insert the copyright notification inside content.

5.2 Properties and Some Approaches

Properties of watermarking systems are mainly associated with two main algorithms — watermark insertion and detection. The significance of a property depends on the goals of the application and the task of the watermark in such application. The interpretation of the property, therefore, also varies with the application. Typically, research on digital watermarking attempts to achieve three main features — imperceptibility, robustness and security — whereas the evaluation of the last two features relies on the watermark's detectability.

5.2.1 Detectability

Detectability is the capability of watermark to be detected or extracted. Based on the method used to detect watermark, a watermarking scheme can be categorised as non-blind or blind watermarking [13]. Non-blind watermarking uses an *informed detector* which requires the original content. As the availability of the original content is limited, this system allows only a selected group of people to run the detector.

Conversely, a blind watermarking utilises a *blind detector* and allows more parties to detect the watermark without the original content.

The basic method for robust watermark detection is brute force search. This method systematically checks all possible components for the watermark signal and can guarantee successful detection in most cases. However, it is costly when the search space is large. Reducing the search area by guessing the likely distortions can make detection efficient, but it is impractical due to the huge amount of possible distortions. Synchronisation process before detection can be useful, enabling watermark to withstand geometrical transforms, such as rotation, scaling and translation (RST). Synchronisation is a process to realign a distorted image by overlying it with an undistorted image.

Early watermark synchronisation was applied on the global scale with the assumption that geometrical transforms are applied to an image homogeneously. However, local geometrical distortions, such as random bending attack (RBA), can make the watermark undetectable. To overcome the local distortion problem, a differential affine motion estimation can be utilised to estimate the direction and distance of object movements by comparing two images [52]. However, this method is costly due to algorithm complexity. To minimise the complexity, a flow line curvature can be applied on the host image before watermark embedding and detecting [198]. Flow line curvature is used to compute robust corners which are selected to resynchronise image. The invariant properties of flow line curvature are exploited to minimise computational cost. Instead of searching through hundreds of corners, this method only needs to work with 4 robust corners: 2 corners are recorded during watermark embedding and 2 corners are computed in watermark detection.

Synchronisation can be used to improve the watermark detectability and is applicable in both non-blind and blind detections. However, the synchronisation itself needs the reference image, which may not be readily available during watermark detection. Therefore, blind watermarking could be more appropriate for many real

applications.

5.2.2 Imperceptibility

Imperceptibility (a.k.a fidelity or perceptual transparency) of watermark aims at preserving the quality of the host content. While the watermark should act as a distinctive identifier, it must not degrade the aesthetic value of the content. The watermarked content is desired to have the same performance as the original one. Imperceptibility in the digital image watermarking, for instance, can be achieved by implementing Human Visual System (HVS) model [63]. Human's eyes are less sensitive to change made in highly textured regions, which have complex patterns, compared to flat regions, which are monotonous. Therefore, a bigger watermarking weight can be applied for complex textured regions compared to simple textured regions. This is a key for achieving a strong invisible watermark.

The imperceptibility of a watermarking method is usually measured by how similar the watermarked image is to the original image. A typical test that is used to evaluate the similarity of a watermarked image and its host image is *Peak Signal to Noise Ratio (PSNR)*:

$$PSNR = 10loq_{10}\left(\frac{(I_{MAX})^2}{MSE}\right) \tag{5.1}$$

where I_{MAX} is the maximum gray levels of the image (i.e. 255) and *Mean Squared Error (MSE)*:

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(X_i - X_i^*)^2 \tag{5.2}$$

with X and X^* are two compared images. The MSE is the average value of the square of the difference between the two images. Having large error in few pixels and no error in the rest gives an acceptable result according to the MSE. This situation is not acceptable in a watermarking application that aims at hiding the existence of a watermark in the image. As the PSNR is the logarithm of the reciprocal of the

MSE, it exhibits the same drawback for watermarking applications. A better test, the *Structural Similarity (SSIM)* index, is more localized as it is the average of a similarity index computed on local neighborhoods in the image [195]. The SSIM index is formulated as

$$SSIM = \frac{(2\mu_x\mu_y + c_1)(2\sigma_{xy} + c_2)}{(\mu_x^2 + \mu_y^2 + c_1)(\sigma_x^2 + \sigma_y^2 + c_2)} \tag{5.3}$$

where μ,σ, and σ_{xy} are mean, variance and covariance of the images, respectively, and c_1, c_2 is the stabilising constants. $SSIM$ has a value between 0 to 1. Similar images have $SSIM$ close to 1.

The SSIM takes advantage of known characteristics of the HVS. This method was developed based on the hypothesis that the HVS is highly adapted for extracting structural information. The SSIM assessment is also applicable to colour images. To assess a colour image, we firstly need to extract the first layer of the image which is a gray-scale image. The SSIM is then applied to the layer to measure the quality of the image.

5.2.3 Security and Robustness

Security of a watermark refers to its resistance to unauthorized detecting and decoding. A secure watermark may benefit content copyright protection as the significant information safely remains associated with the content wherever it is copied or distributed. Watermark is also desired to withstand multiple watermarking and be statistically unremovable [142]. In contrast, some other applications, such as tamper detection, tamper localisation and content recovery, may only need a low level of security. In such applications, small modification of the watermark can be valuable for the owner to identify the change.

Robustness of a watermark refers to its resistance against common processing, such as filtering, geometrical transforms and compression. A watermark that re-

sists all possible alterations is desirable, however in the real implementations, it is not necessary for a watermark to cover everything. For example, as a low quality compression will degrade image value, image watermarking needs to survive only to high quality image compression. A survey of watermarking research shows that many watermarking schemes give consideration to robustness more than security [130]. However, a robust watermark is not enough to accomplish protection because the range of hostile attacks is not limited to common processing and distortions. Therefore, robustness and security should be proportionally considered in a watermarking system.

A popular approach to achieve the robust and secure watermarking for multimedia content is the spread-spectrum technique. A general spread-spectrum system encodes data in a chosen binary sequence that appears like noise to an outsider but can be recognized by a legitimate receiver with the aid of an appropriate key [145]. Spread-spectrum watermarking takes advantage of the large bandwidth of the host content by matching the narrow bandwidth of the embedded watermark to it. In terms of perceptibility and robustness, high and low frequency have contrary characteristics. High frequencies support imperceptibility of the watermark but are less robust. Conversely, low frequencies support robustness but cause unacceptable impact on the content quality. Spread-spectrum watermarking resolves this conflict by embedding a low-energy watermark in each frequency band.

The most cited secure spread-spectrum watermarking method was one presented by Cox *et al* [45]. In this method, the watermark is placed into the perceptually most significant components of content spectrum, since many common signal and geometrical processes affect the insignificant regions of content. Practically, to place a length n watermark into a $N \times N$ image, the $N \times N$ discrete cosine transform (DCT) of the image is computed and the watermark is placed in the n highest magnitude coefficients of the transform matrix, excluding the DC components. This watermarking scheme yields a watermark which is invisible, difficult to remove and

robust in spite of common signal and geometrical distortions. However, this method is a non-blind watermarking and uses a sequence of meaningless random numbers as the watermark. A non-blind watermarking might be less applicable, because the original image may not be available when watermark detection is required.

In the proposed schemes presented in this Chapter, a meaningful watermark, i.e a binary image, is used in constructing a blind watermarking scheme. To achieve security, the watermarking scheme is constructed based on chaotic maps.

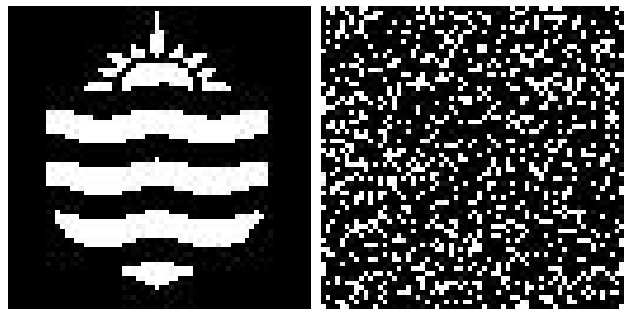
5.3 Chaotic Map and Watermarking

A chaotic map is an evolution function that exhibits some sorts of chaotic behavior. This map can be parameterised by discrete-time or continuous-time. Discrete maps usually take the form of iterated functions. The most attractive features of chaos in information hiding are its extreme sensitivity to initial conditions and the out-spreading of orbits over the entire space. These special characteristics make chaotic maps excellent candidates for securing watermarks.

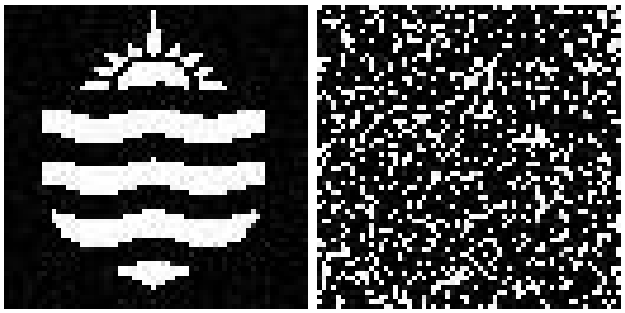
Choosing chaotic map to develop watermarking schemes in this thesis is basically intended to increase the security of the schemes. This choice is based on following considerations.

- Chaotic maps are very sensitive to initial conditions so that the security of the chaotic map based watermarking relies on the security of the secret keys. This condition is in line with the *Kerckhoffs' principle* [145] as in cryptography.

- Chaotic sequences can be constructed using a simple equation and thus, the watermarking computation complexity can be minimised.



(a) Binary JCU Logo (b) Permuted Logo



(c) Correctly reformed (d) Wrongly reformed

Figure 5.1: Example of image permutation using logistic map and two reconstructed images

5.3.1 Logistic Map

One of the simplest chaotic maps is the logistic map. The logistic map is discrete and mathematically written as

$$z_{n+1} = r z_n (1 - z_n) \tag{5.4}$$

where $z_n \in (0, 1]$ and $0 \leq r \leq 4$. When $r > 3.57$, the map is in the chaotic state [48]. The sequences generated by the logistic map are sensitive to initial value, meaning that two logistic sequences generated from different initial values are statistically uncorrelated.

Figure 5.1 illustrates how a logistic map can be used to achieve security. A binary image (Figure 5.1a) is permuted according to a generated logistic sequence with initial value $z_0 = 0.345$ (Figure 5.1b). The binary image is successfully reconstructed using the exact initial value, $z_0 = 0.345$ (Figure 5.1c), but it fails to be reconstructed

even using a close value, such as $z_0 = 0.346$ (Figure 5.1d).

5.3.2 Some Chaos-based Watermarking Schemes

Chaotic maps have been proposed to undertake various tasks in watermarking schemes. For example, a watermarking scheme employs logistic functions to modify Cox's watermarking, by adding a watermark encryption feature [123]. Instead of using a sequence of meaningless random numbers, this scheme uses a binary image as the watermark. The binary watermark is firstly encrypted using two generated logistic sequences and then spread into the host image spectrum using Cox's technique [45]. To extract the watermark, the modified Cox's scheme also requires the original host image. Thus, it is a non-blind watermarking. Though the scheme is robust against JPEG compression, it is susceptible to cropping, resizing and adding noise.

Chaotic maps are often used to modify a watermark before the embedding process. The modification could be mutation, permutation or a mixture. Watermark mutation is a process to randomly change the value of watermark pixels according to a chaotic sequence [12]. The mutated watermark pixels are then randomly embedded on the host image spectrum according to another generated chaotic sequence. This mechanism can preserve the hidden information against geometric and non geometric attacks. Next, watermark permutation is a process to randomly change the watermark bits' positions. In a scheme proposed by Wang *et al.* [192], after the watermark is permuted using the first chaotic sequence, a small number of reference points are randomly selected in the middle frequency bands of the host image spectrum according to the second chaotic sequence. The permuted watermark bits are then embedded into the neighborhood of each reference point according to the third sequence. This mechanism was proposed to achieve watermark imperceptibility and robustness. Another watermarking scheme used chaotic map iterations to accomplish watermark mixture [77]. The initial state of the iteration is constituted

by the watermark, which is considered as a Boolean vector. The subsequent chaotic Boolean vectors are generated by a number of iterations of a logistic map. The mixed watermark is the last Boolean vector generated by the chaotic iterations.

Some chaotic-based watermarking schemes rely on securing the selection of embedding locations. To firmly insert a watermark signal, some schemes randomly selected only several local spectrums as the embedding locations [48, 65, 111, 122]. In these schemes, a sub image is constructed from the host image according to a logistic sequence. The spectrum of the sub image is then used as the embedding location. Another scheme uses a 2-D invertible chaotic map to determine watermark insertion location in the spatial domain [200, 199]. Basically, the 2-D chaotic map is a one to one map. Utilising this map is intended to improve the success of watermark extraction.

Generally speaking, a chaotic map is employed in watermarking schemes to achieve watermark security. The map is typically used to accomplish three tasks — generating watermark, encrypting watermark and selecting insertion location. The initial value that is used to generate the chaotic sequence can be used as a component of the secret keys for detecting the watermark. Without appropriate keys, the watermark will be wrongly extracted or detected.

5.4 Proposed Schemes

In this section, three chaotic-based blind watermarking schemes are proposed. The first and second schemes are undertaken in the spatial domain, while the third is in the frequency domain. The simulations of all proposed schemes use square images as the hosts and a binary image as watermark (see Figure 5.2).

Figure 5.2: *Watermark: binary JCU logo*

5.4.1 The First Scheme

The first scheme employs two 1-D logistic maps (equation 5.4) and one 2-D chaotic map (equation 5.5). The first 1-D logistic map is used to encrypt the watermark. The encrypted watermark is then inserted into the host image according to the 2-D chaotic map and the second 1-D logistic map.

Watermark Insertion

The watermark insertion process begins with the watermark encryption. To encrypt a watermark W, a logistic sequence L_1 is firstly generated and then converted into a binary sequence L_{bin}. Finally, the watermark W is bitxor-ed with L_{bin} to obtain the encrypted watermark W_{enc}. The detail of this encryption process is presented in Algorithm 5.1.

The encrypted watermark W_{enc} is then inserted into the host image. The insertion process of each bit of the encrypted watermark is undertaken in two steps — determining the insertion location and then performing the pixel modification. The insertion location is determined according to the 2-D chaotic map. This map is a generalised form of the 2-D Arnold cat map [101] and is described by:

$$\begin{bmatrix} x_{t+1} \\ y_{t+1} \end{bmatrix} = \begin{bmatrix} a & b \\ c & d \end{bmatrix} \begin{bmatrix} x_t \\ y_t \end{bmatrix} = A \begin{bmatrix} x_t \\ y_t \end{bmatrix} \bmod N \tag{5.5}$$

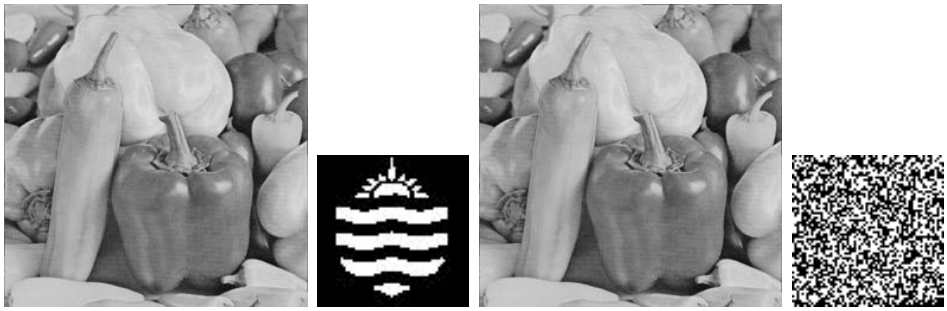
where a, b, c and d are positive integers such that $|A| = ad - bc = 1$, thus, three of these four parameters are independent; N in the first scheme is the number of rows or columns of the host image.

Algorithm 5.1 Encrypt a binary watermark

Input: watermark W which is an $M \times M$ binary image, and initial value of logistic map z_{0_1}
Output: encrypted watermark W_{enc}
BEGIN
- Generate a logistic sequence L_1 of length M^2 using equation 5.4 with the initial value z_{0_1}
- Compute m = mean of L_1
- Convert L_1 into a binary sequence L_{bin} using:
for $i = 1$ to M^2 **do**
  if $L_{1_i} \geq m$ **then**
    $L_{bin_i} = 1$
  else
    $L_{bin_i} = 0$
  end if
end for
- Reshape L_{bin} to an $M \times M$ binary matrix
- Bitwise exclusive OR (bitxor) the reshaped L_{bin} and W to obtain W_{enc}
END

The mechanism to select the insertion position for each bit of W_{enc} adopts the technique proposed by Wu *et al.* [199, 200] with some minor modifications. To determine the insertion location of a watermark bit, Wu *et al.* chose an initial value (x_0, y_0) and iterated it using equation 5.5 for n times. The output of this iteration round, (x_n, y_n), is determined as the insertion location of the watermark bit and used as the initial value of the next iteration round for the next watermark bit. The chain of these iteration rounds will end after all watermark bits are placed. However, according to the simulation using the same initial values as theirs (a=1, b=2, c=3, $(x_0, y_0) = (2, 3)$, n=20), this mechanism results in failure to extract watermark (see Figure 5.3). This failure is caused by placement conflicts. Some watermark bits are mapped into the same insertion location (see Table 5.1), so that the inverse function fails to reveal some original watermark bits.

In this scheme, therefore, instead of using a chain of dependent iteration rounds, the insertion location of each bit of W_{enc} is computed independently. Note that each pixel of W_{enc} has only one bit. This mechanism is intended to keep the natural property of the 2-D chaotic map, which is a one to one map, so that the watermark



(a) (b) (c) (d)

Figure 5.3: Failure of extraction due to placement conflicts: 5.3a: Original "chilli" image; 5.3b: Watermark: 64×64 binary JCU logo; 5.3c: Watermarked image; 5.3d: Wrongly extracted watermark

Table 5.1: Examples of the placement conflicts

Original coordinate	Insertion location	Original coordinate	Insertion location
(60,62)	(2,99)	(60,64)	(2,3)
(60,46)	(2,99)	(60,48)	(2,3)
(60,14)	(2,99)	(60,39)	(2,179)
(59,62)	(2,99)	(60,55)	(2,179)

bits of different coordinates will be mapped into different insertion positions. For each encrypted watermark bit $W_{enc_{i,j}}$, its coordinate (i, j) is served as the initial value and iterated n times using equation 5.5 to obtain its insertion position, (x_n, y_n) (see algorithm 5.2).

Algorithm 5.2 Determine insertion location

Input: (i, j), a, b, c
Output: (x_n, y_n)
BEGIN
- Compute d, such that $ad - bc = 1$
- Form matrix

$$A = \begin{bmatrix} a & b \\ c & d \end{bmatrix}$$

- Iterate (i, j) for n times using equation 5.5 with the coefficient matrix, A.
- Return (x_n, y_n)
END

Once the insertion location has been found, the second 1-D logistic map (equation 5.4) is used to modify pixel at the coordinate of (x_n, y_n) — selecting the bit of the binary representation of this pixel value to be replaced by the corresponding bit

Table 5.2: SSIM index at different bits location

Location	Mandril	Lena	House	Chilli	Boat
LSB	0.9977	0.9947	0.9940	0.9947	0.9963
2 Middle bits	0.9864	0.9679	0.9668	0.9692	0.9785
4 Middle bits	0.9698	0.9387	0.9275	0.9375	0.9530
MSB	0.7949	0.6839	0.6415	0.6739	0.7423

of W_{enc}. A simulation involving five host images is undertaken to obtain SSIM index for each group of pixel's bit replacement. Table 5.2 shows that inserting watermark signal in four least significant bits (LSB) exhibits the best performance of the watermarked image. However, inserting watermark bits in this area is weak against attacks as watermark can be easily detected. In contrast, inserting watermark bits in four most significant bits (MSB) yields a noticeable degradation in the host image. Wu's algorithm selected four middle pixels' bits to insert watermark bits [199, 200], but we use only two middle bits for a better watermarked image performance.

Suppose b_n is the binary representation of I_{x_n,y_n}, the host image's pixel value of coordinate (x_n, y_n). To determine one of the two middle bits of b_n that must be replaced by $W_{enc_{i,j}}$, an initial value z_{0_2} is iterated n times using the second logistic map. The obtained value z_n is then used to determine the kth bit of b_n that has to be replaced by $W_{enc_{i,j}}$. If $z_n \geq 0.5$ then $k = 4$; and if $z_n < 0.5$ then $k = 5$. The detail of the insertion process of a single encrypted watermark bit is presented by Algorithm 5.3.

The summary of the watermark insertion process is described as follows. Suppose I and W are the host image and binary watermark, respectively. The secret keys to insert W into I include six components — the initial values of the first and second logistic map, (z_{0_1} and z_{0_2}, respectively), three independent parameters of the 2-D chaotic map (a, b and c), and the number of iteration (n). The insertion stages are:

1. Run Algorithm 5.1 to encrypt W.

2. For each encrypted watermark's bit $W_{enc_{i,j}}$:

Algorithm 5.3 Pixel modification

Input: $W_{enc_{i,j}}$, b_n, z_{0_2} and n
Output: b'_n, the new binary representation of b_n
BEGIN
- Run equation 5.4 iterate z_{0_2} for n times
- Return z_n
if $z_n \geq 0.5$ **then**
    Set k to 4
else
    Set k to 5
end if
- Replace the kth bit of b_n by $W_{enc_{i,j}}$ to obtain b'_n
- Set b'_n as the new binary representation of I_{x_n,y_n}
- Set z_n as the initial value of the next iteration round.
END

(a) Run Algorithm 5.2 to determine insertion location for $W_{enc_{i,j}}$.

(b) Run Algorithm 5.3 to modify the pixel of I at the insertion location by $W_{enc_{i,j}}$.

3. After all bits of W_{enc} have been inserted, the watermarked image, I_{wat}, is obtained.

Watermark Extraction

The process to extract watermark is just the inverse of the insertion process. The secret keys (z_{0_1}, z_{0_2}, a, b, c, n) and the watermark length are needed in this process. Error in providing any of these components will result in failure in watermark extraction. Let I_{test} and $W'_{enc_{i,j}}$ denote the test image and the encrypted watermark pixel of coordinate (i, j) to be extracted, respectively. The watermark extraction is undertaken in the following stages.

1. For each coordinate (i, j) of W'_{enc}, run Algorithm 5.2 to determine from which I_{test}'s pixel the encrypted watermark's bit $W'_{enc_{i,j}}$ has to be extracted.

2. Let (x_n, y_n) is the coordinate obtained in stage 1. Suppose the binary representation of pixel value of I_{test} at coordinate (x_n, y_n) is b'_n. Run Algorithm 5.4

to get $W'_{enc_{i,j}}$ from b'_n.

3. Repeat stages 1 and 2 until all encrypted watermark pixels are extracted.

4. Once W'_{enc} has been completely reconstructed, run Algorithm 5.1 to decrypt W'_{enc} and obtain an extracted watermark W'_{ext}.

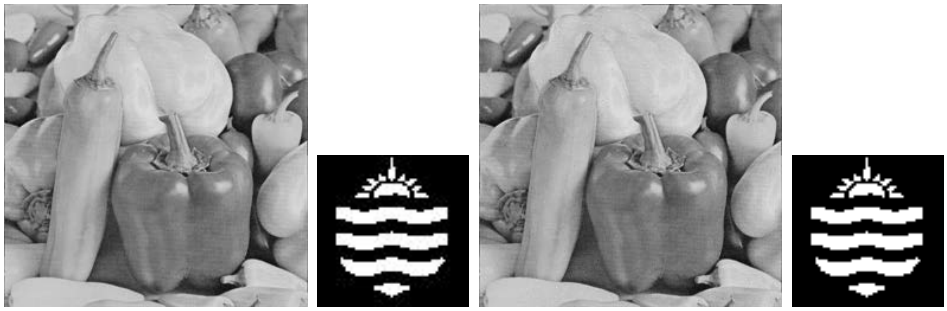
Algorithm 5.4 Extract a watermark bit

Input: b'_n, z_{0_2} and n
Output: $W'_{enc_{i,j}}$
BEGIN
- Iterate z_{0_2} using equation 5.4 for n times to obtain z_n
if $z_n \geq 0.5$ **then**
  Set k to 4
else
  Set k to 5
end if
- Take the kth bit of b'_n as $W'_{enc_{i,j}}$
- Set z_n as the initial value of the next iteration round.
END

Experimental Result

To implement the proposed algorithm, MATLAB simulations are performed by using 256×256 gray scale images ("boat", "chilli", "house", "lena" and "mandril") as the host images and a 64×64 binary "JCU Logo" as the watermark. The secret keys consist of $z_{0_1} = 0.642$, $z_{0_2} = 0.537$, $a = 1$, $b = 2$, $c = 3$ and $n = 20$. Figure 5.4 shows an example of the simulation result. This scheme exhibits a high imperceptibility value of watermark. With $SSIM = 0.9692$, the watermarked and the original image give no noticeable difference. The scheme is perfectly secure. By providing the correct value of all secret key components (z_{0_1}, z_{0_2}, a, b, c, n) plus the size of watermark, the extraction process successfully recovers the watermark. Otherwise, it will be failed.

To test the robustness of the watermarking scheme, some image processes are applied to the watermarked image including image enhancement (contrast and bright-



(a) (b) (c) (d)

Figure 5.4: Demonstration of imperceptibility of the first scheme: 5.4a: Original "chilli" image; 5.4b: Watermark: 64×64 binary JCU logo; 5.4c: Watermarked image; 5.4d: Extracted watermark

Table 5.3: SSIM of extracted watermark in the 1st scheme

Image Processing	Boat	Chilli	House	Lena	Mandril
Contrast	0.0612	0.0406	0.0256	0.0295	0.0338
Exposure	0.1509	0.1164	0.1514	0.0364	0.1395
JPEG Q. 20 %	0.0214	0.0289	0.0072	0.0162	0.0107
JPEG Q. 80 %	0.0975	0.1063	0.0750	0.0791	0.0898
Add Noise %	0.7826	0.7548	0.7772	0.7572	0.7396
Cropping 25 %	0.4105	0.4105	0.4105	0.4105	0.4105
Rotate 2^o	0.0174	0.0077	0.0073	0.0204	0.0003
Rotate 10^o	0.0079	0.0001	0.0255	0.0235	0.0177
Rotate 90^o	0.0111	0.0143	0.0167	0.0064	0.00005
Horizontal Flip	0.00002	0.0113	0.0005	0.0115	0.0074
Vertical Flip	0.0028	0.0108	0.0012	0.0031	0.014

ness), JPEG compression (with quality 20 % and 80 %), "salt & pepper" noise addition, cropping, rotation (degree 2, 10 and 90), horizontal and vertical flipping. After each processing, a watermark signal is extracted from the modified watermarked image. The extracted signal is then compared to the original watermark to measure their similarity index (SSIM).

Table 5.3 shows the SSIM index of the extracted signals. The recovered signal will provide a noticeable watermark if SSIM > 0.2. According to the table, however, watermark can only be recovered successfully after noise addition and cropping operations. Thus, in spite of a prefect watermark security, the first watermarking scheme is vulnerable against image enhancement (contrast and brightness), image compression and geometry operations. See Appendix D.1 for the visual presentation

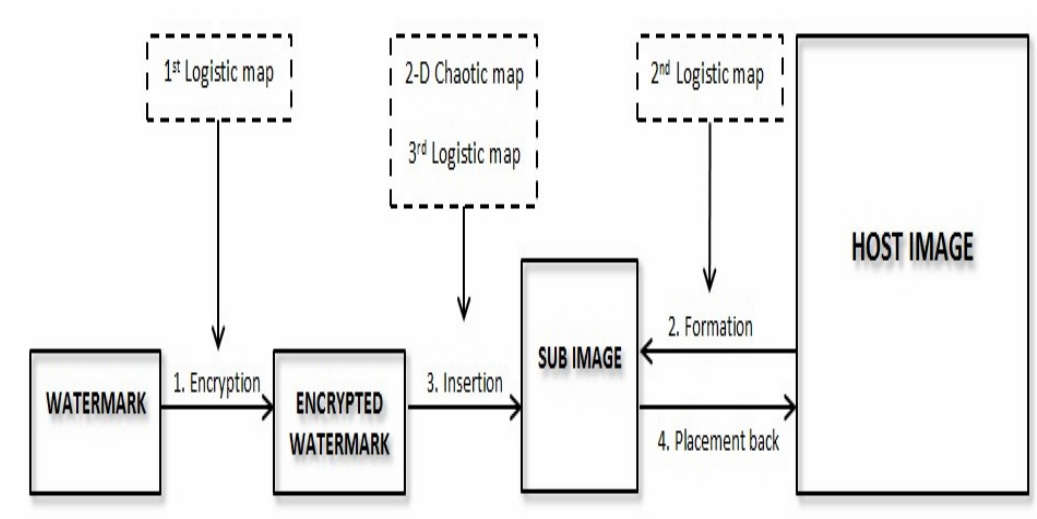


Figure 5.5: *Watermark insertion mechanism of the 2nd scheme*

of this result.

5.4.2 The Second Scheme

Like the first one, the second watermarking scheme is also undertaken in spatial domain. However, instead of using whole host image, this scheme uses a sub image extracted from the host image as the watermark insertion area. Besides using two 1-D logistic maps for watermark insertion process as in the first watermarking scheme, this scheme requires one more 1-D logistic map for constructing a sub image. Therefore, the secret keys for inserting watermark consist of seven components — the initial value of the first, second and third logistic maps (z_{0_1}, z_{0_2} and z_{0_3}, respectively), three independent parameters of the 2-D chaotic map (a, b and c), and the number of iteration (n).

Watermark Insertion

The insertion mechanism uses three 1-D logistic maps and one 2-D chaotic map (see Figure 5.5). The first logistic map is used to encrypt watermark. This encryption process is undertaken by Algorithm 5.1. The second logistic map is employed to construct a sub image I_{sub} from the host image I. Suppose I is the host image of size

$N \times N$ and m is a divisor of N, the construction I_{sub} is performed by Algorithm 5.5. The encrypted watermark is then inserted into I_{sub} according to the 2-D chaotic map (for insertion location determination) and the third logistic map (for pixel modification). Like the first scheme, for each bit of encrypted watermark W_{enc}, the insertion location determination and the pixel modification are performed using Algorithms 5.2 and 5.3, respectively. Recall equation 5.5, the computation is in modulo N. In the second scheme, N is the number of rows or columns of the constructed sub image. All blocks of the watermarked I_{sub} are then placed back into their original position in I according to the same sequence generated by the second logistic map.

Algorithm 5.5 Construct a sub image

Input: image I which is an $N \times N$ gray scale image, and initial value of the logistic map z_{0_2}
Output: sub image I_{sub}
BEGIN
- Split I into some disjoint blocks of $m \times m$ pixels each, where m is a divisor of N
- Label all blocks in a scan-line order by positive integers from 1 to $(N/m)^2$
- Generate a logistic sequence L of length $(N/m)^2$ using equation 5.4 with the initial value z_{0_2}
- Sort L in ascendance
- Construct a sequence S containing the original indexes of the elements of the sorted L. Note that the elements of S must be the first $(N/m)^2$ positive integers
- Select blocks from split I according to the first $1/4(N/m)^2$ elements of S
- Construct I_{sub} by arranging the selected blocks in a scan-line order. Note that I_{sub} is an $\frac{N}{2m} \times \frac{N}{2m}$ blocks or $\frac{N}{2} \times \frac{N}{2}$ pixels image.

END

Watermark Extraction

The secret keys to undertake a successful watermark extraction consist of eight components — seven components as those in the insertion process and the watermark length. The watermark extraction is just the inverse of the insertion process (see Figure 5.6).

Suppose I_{test} is the test image. A sub image is constructed from I_{test} according

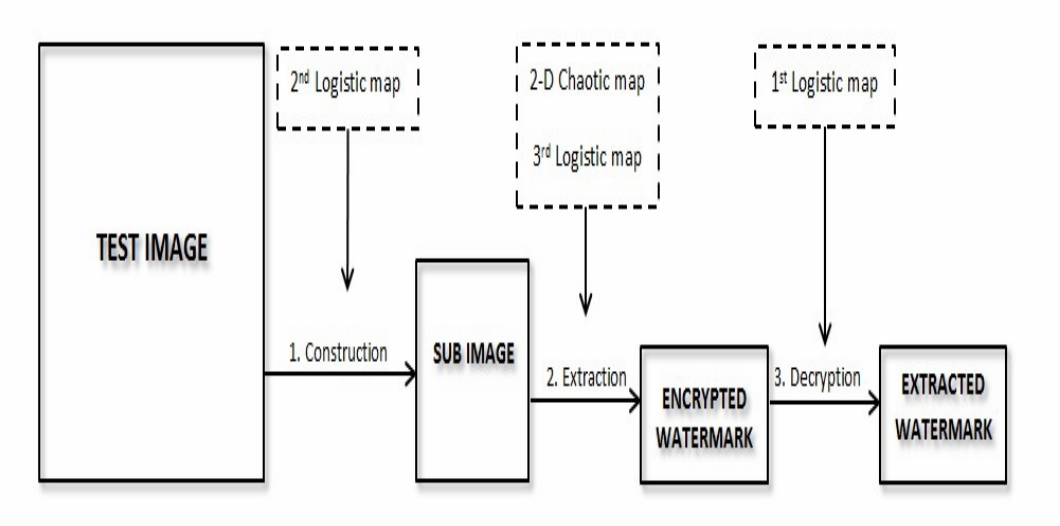


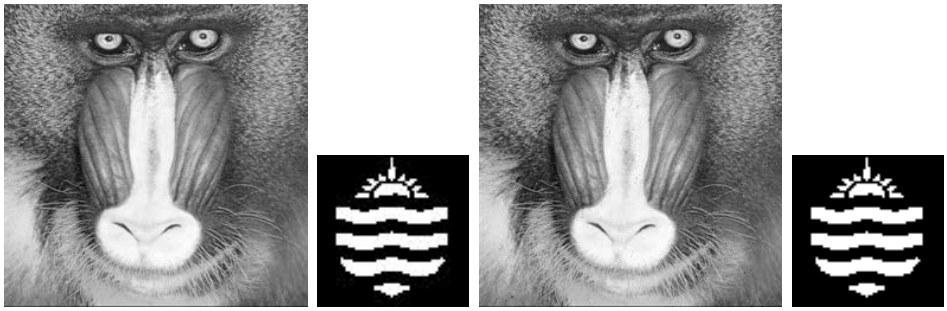
Figure 5.6: *Watermark Extraction of the 2nd Scheme*

to the second logistic map through the similar mechanism to Algorithm 5.5. With aid of the 2-D chaotic map and the third logistic map, an encrypted watermark is extracted from the sub image in the same way as those in Algorithm 5.2 and 5.4, respectively. Finally, the first logistic map is used to decrypt the encrypted watermark to obtain an extracted watermark. The decryption process is similar to the encryption mechanism by Algorithm 5.1.

Experimental Result

The simulation of this watermarking scheme uses the same five 256×256 gray scale images and a 64×64 binary watermark as those in the first scheme. The secret keys of the watermark embedding process consist of seven components — $z_{0_1} = 0.642$, $z_{0_2} = 0.537$, $z_{0_3} = 0.689$, $a = 1$, $b = 2$, $c = 3$ and $n = 20$.

First of all, the binary watermark is encrypted according to the first logistic map. Later, the encrypted watermark is inserted into a sub image constructed from the host image. To generate a sub image, the 256×256 host image is split into some 8×8 pixels blocks. There are 1024 such blocks and each block is labeled by positive integers from 1 to 1024 in a scan-line order. To select blocks, a logistic sequence of length 1024 is generated using the second logistic map and then sorted



(a) (b) (c) (d)

Figure 5.7: Demonstration of imperceptibility of the second scheme: 5.7a: Original "mandril" image; 5.7b: Watermark: 64×64 binary JCU logo; 5.7c: Watermarked image; 5.7d: Extracted watermark

in ascending order. 256 blocks are selected according to the original indexes of the first 256 sorted sequence's elements. The 256 selected blocks are used to construct a 16×16 blocks (i.e. a 128×128 pixels) image. This is the sub image that is to be used as the insertion area of the encrypted watermark. The embedding process is undertaken using the 2-D chaotic map and the third logistic map. After insertion, the watermarked sub image is placed back to the host image to obtain a watermarked image.

Figure 5.7 shows that there is no noticeable difference between the original and watermarked image. This means that the second scheme also achieves a good watermark imperceptibility. In the example shown in this figure, $SSIM$ of watermarked "mandril" is 0.9717. This scheme also achieves a perfect security. Unless provided with the correct value of all seven secret keys components (z_{0_1}, z_{0_2}, z_{0_3}, a, b, c,n) plus the size of the watermark, one cannot recover the watermark.

To test the robustness of this scheme, the same image processing as in the first scheme is applied to the watermarked image. A signal extracted from the modified watermarked image is compared to the original watermark to measure their similarity index (SSIM). Table 5.4 shows that the second watermarking scheme is robust against noise addition and cropping operations, but is weak to image enhancement, compression and geometry operations. See Appendix D.2 for the visual presentation of this result.

Table 5.4: SSIM of extracted watermark in the 2nd scheme

Image Processing	Boat	Chilli	House	Lena	Mandril
Contrast	0.0574	0.0492	0.1766	0.0564	0.104
Exposure	0.0486	0.0616	0.1921	0.0402	0.0351
JPEG Q. 20 %	0.0078	0.0171	0.0076	0.0102	0.0123
JPEG Q. 80 %	0.1250	0.1149	0.1252	0.1169	0.1075
Add Noise %	0.7934	0.7789	0.7428	0.7804	0.7756
Cropping 25 %	0.4190	0.4190	0.4190	0.4190	0.4190
Rotate 2^o	0.0161	0.0078	0.0176	0.0034	0.0052
Rotate 10^o	0.0027	0.0018	0.00006	0.0096	0.0024
Rotate 90^o	0.0118	0.0058	0.0085	0.0056	0.035
Horizontal Flip	0.0152	0.0157	0.0084	0.0014	0.0056
Vertical Flip	0.0109	0.0037	0.0228	0.0036	0.0056

5.4.3 The Third Scheme

The third scheme is proposed to overcome the drawback of previous schemes. This scheme uses a sub image as the watermark insertion area like the second scheme, but the insertion process is undertaken in the frequency domain [156]. That is, once the sub image has been generated, it is transformed into its spectrum. The information of the sub image spectrum and the encrypted watermark are used to construct a spread-spectrum watermark. The spread-spectrum watermark is then inserted into sub image spectrum.

In practice, we use Discrete Cosine Transform (DCT) to compute the sub image's spectrum, The choice of DCT among a number of transform domains available is based on the following consideration.

- DCT separates the image into spectral sub-bands of differing importance with respect to the image's visual quality.

- DCT is similar to the discrete Fourier transform (DFT) — transforming image from the spatial to the frequency domain —, but DCT has a simpler computation.

- It is computationally easier to implement and more efficient to regard the DCT

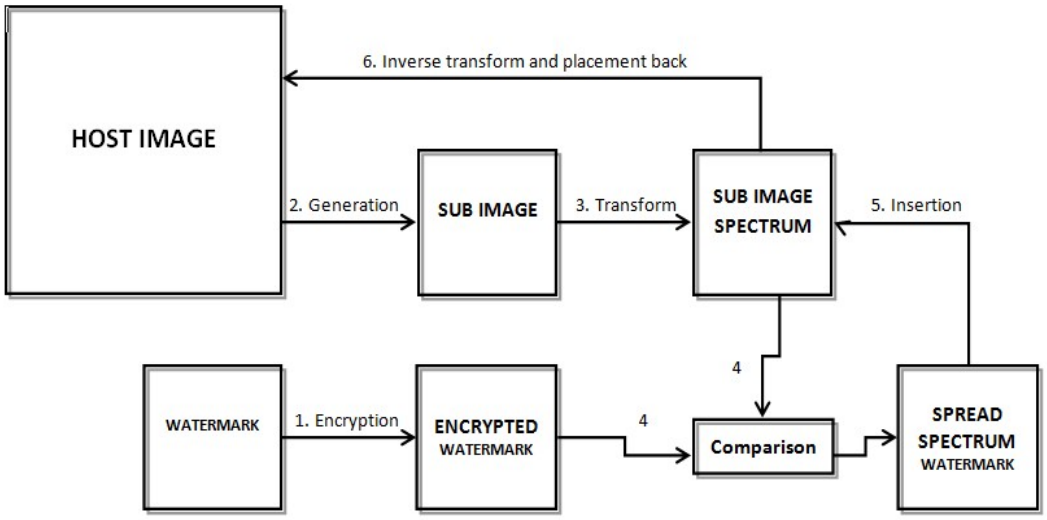


Figure 5.8: *Watermark Insertion of the 3rd Scheme*

as a set of basis functions which given a known input array size (8×8) can be computed and stored. This involves simply computing values for a convolution mask (8×8 window) that get applied. The values are simply calculated from the DCT formula.

- DCT is similar to the Fast Fourier Transform (FFT), but can approximate lines well with fewer coefficients.

Watermark Insertion

The watermark insertion process (see Figure 5.8) employs two 1-D logistic maps (equation 5.4) and one 2-D chaotic map (equation 5.5). The first and second 1-D logistic maps are used to encrypt the watermark W and construct a sub image from the host image, respectively, while the 2-D chaotic map is utilized to select the insertion locations. The process begins with Algorithm 5.1 to obtain an encrypted watermark W_{enc}. The next stage is performing Algorithm 5.5 to generate a sub image I_{sub} from the host image I.

The information of the sub image I_{sub} and the encrypted watermark W_{enc} are then used to generate a spread-spectrum watermark W_{spec}. First of all, each element of I_{spec} is compared with its neighbors. An element may have three, five or eight

neighbors depending on its position. Suppose $I_{spec_{i,j}}$ is the element of I_{spec} at the coordinate (i, j) and $t_{i,j}$ is the number of $I_{spec_{i,j}}$'s neighbors having less values than $I_{spec_{i,j}}$. The construction of W_{spec} is undertaken according to equation 5.6.

$$W_{spec_{i,j}} = \begin{cases} 1, & if \quad (t_{i,j} \geq 2 \bigwedge W_{enc_{i,j}} = 1) \bigvee (t_{i,j} < 2 \bigwedge W_{enc_{i,j}} = 0) \\ -1, & otherwise \end{cases} \tag{5.6}$$

The spread-spectrum watermark W_{spec} is then inserted into sub image spectrum I_{spec}. The insertion location of each element of W_{spec} is computed using Algorithm 5.2. All computations are taken in modulo N, where N is the number of rows or columns of the constructed sub image I_{sub}. For each element of W_{spec}, its coordinate (i, j) is served as the initial value and iterated n times using equation 5.5. The iteration result, (x_n, y_n), will be served as the insertion position of $W_{spec_{i,j}}$.

Once the insertion position has been selected, $W_{spec_{i,j}}$ is then combined with the element of I_{spec} of the coordinate (x_n, y_n) using equation 5.7.

$$I'_{spec_{x,y}} = I_{spec_{x,y}} + \beta W_{spec_{i,j}} |I_{spec_{x,y}}| \tag{5.7}$$

where β is an intensity parameter of the watermark. After all elements of W_{spec} have been inserted, the watermarked sub image spectrum (I'_{spec}) is finally transformed back into its spatial domain and then placed back into the host image to obtain a watermarked image.

A summary of the watermark insertion process is as follows. The insertion process requires six components of the secret keys — initial values of the first and second logistic map (z_{0_1} and z_{0_2}, respectively), three independent parameters (a, b and c) for the 2-D chaotic map, and the number of iterations (n). Suppose I and W are the host image and the watermark, respectively, the insertion process is undertaken through the following stages (refer to Figure 5.8).

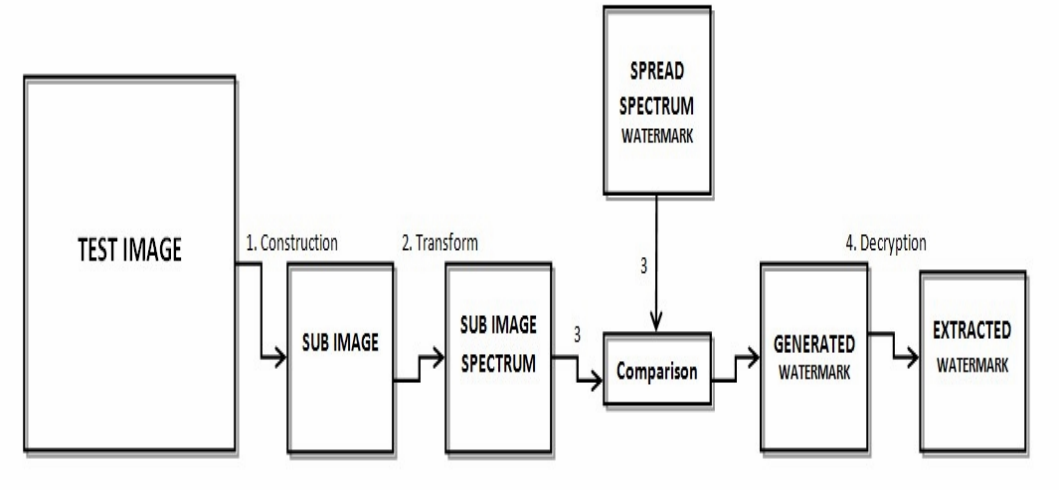


Figure 5.9: *Watermark Extraction of the 3rd Scheme*

1. W is encrypted according to the first logistic map by Algorithm 5.1 to obtain W_{enc}.

2. A sub image I_{sub} is constructed from I according to the second logistic map by Algorithm 5.5.

3. I_{sub} is transformed into its spectrum I_{spec}

4. The information of I_{spec} and W_{enc} are then used to construct a spread-spectrum watermark W_{spec}.

5. W_{spec} is embedded into I_{spec} according to the equation 5.5 (for the insertion location) and equation 5.7 to obtain a watermarked sub image spectrum I'_{spec}

6. Finally, I'_{spec} is inverse-transformed to its spatial domain and is then placed back into the host image I to obtain a watermarked image I_{wat}.

The spread-spectrum watermark W_{spec} generated in this insertion process has to be saved for the later watermark extraction.

Watermark Extraction

The watermark extraction process (see Figure 5.9) requires six components of the secret keys that were used in the insertion process, the size of watermark and the

spread-spectrum watermark W_{spec}. The extraction process begins with the construction of sub image I'_{sub} from a test image according to the second logistic map by Algorithm 5.5. The test image could be the watermarked image or its modifications. I'_{sub} is then transformed into its spectrum I'_{spec}. The information of I'_{spec} and W_{spec} are used to generate a binary image W'_{enc}. First of all, each element of I'_{spec} is compared with its neighbors. Suppose $t_{i,j}$ is the number of $I'_{spec_{i,j}}$'s neighbors having less values than $I'_{spec_{i,j}}$. W'_{enc} is generated according to equation 5.8.

$$W'_{enc_{i,j}} = \begin{cases} 1, & if \quad (t_{i,j} \geq 2 \bigwedge W_{spec_{i,j}} = 1) \bigvee (t_{i,j} < 2 \bigwedge W_{spec_{i,j}} = -1) \\ 0, & otherwise \end{cases} \tag{5.8}$$

Once W'_{enc} has been generated, it is then decrypted using the first logistic map through the same way as Algorithm 5.1 to obtain an extracted watermark.

Experimental Result

Simulations of watermarking algorithms use the same host images as in previous experiments and a 128×128 binary watermark "JCU Logo". The secret keys consist of $z_{0_1} = 0.642$, $z_{0_2} = 0.537$, $a = 1$, $b = 2$, $c = 3$ and $n = 20$. Figure 5.10 shows a perfectly imperceptibility watermark achieved by this watermarking scheme. There is no significant difference between the original and watermarked image. Moreover, SSIM of all images exhibit that this scheme achieves the best watermarked image performance compared to previous schemes (see Table 5.5). This scheme also achieves perfect security. The watermark can only be recovered using the correct values of all secret key components (z_{0_1}, z_{0_2}, a, b, c,n), the size of the watermark and the spread-spectrum watermarking matrix.

The same image processing used to test the robustness of previous schemes, is also used to test this scheme. The processes are applied to the watermarked images.

(a) (b) (c) (d)

Figure 5.10: Demonstration of imperceptibility of the third scheme: 5.10a: Original "boat" image; 5.10b: Watermark: 128×128 binary JCU logo; 5.10c: Watermarked image; 5.10d: Extracted watermark

Table 5.5: SSIM of watermarked image in three schemes

Scheme	Boat	Chilli	House	Lena	Mandril
1	0.9785	0.9692	0.9668	0.9679	0.9864
2	0.9583	0.9449	0.9417	0.9434	0.9717
3	1	1	1	1	1

Table 5.6: SSIM of extracted watermark in the 3rd scheme

Image Processing	Boat	Chilli	House	Lena	Mandril
Contrast	0.5176	0.5174	0.5097	0.5971	0.5214
Exposure	0.6645	0.6490	0.6318	0.6303	0.6437
JPEG Q. 20 %	0.4775	0.558	0.4605	0.5329	0.4468
JPEG Q. 80 %	0.6134	0.6167	0.5613	0.6182	0.5918
Add Noise %	0.3729	0.3680	0.3481	0.3679	0.4041
Cropping 25 %	0.4508	0.3774	0.3434	0.3863	0.3986
Rotate 2^o	0.2658	0.2734	0.2583	0.2820	0.2835
Rotate 10^o	0.2379	0.2403	0.2395	0.2401	0.2607
Rotate 90^o	0.2219	0.2293	0.2249	0.2350	0.2269
Horizontal Flip	0.2412	0.2519	0.2324	0.2378	0.2657
Vertical Flip	0.2372	0.2198	0.2364	0.2375	0.2369

After each process, a signal is extracted from the modified watermarked image and then compared to the original watermark to measure their structural similarity index (SSIM).

Table 5.6 shows that all SSIM indexes of watermark signals extracted from modified images are more than 0.2. The visualization of these results shows that the watermark can be recovered after the application of all processing. Though the ex-

tracted signals are noisy, the watermark pattern can be visibly distinguished from the noise. Nevertheless, the qualities of the extracted signals vary. These results show that the watermarking scheme is robust enough to withstand image enhancement, compression, noise addition and cropping, but is vulnerable to some geometric operations, such as rotating and flipping. The watermark signals extracted after these geometric operations suffer significant degradation. This condition could be a challenge for the watermark in preserving protection of the image.

To improve the capability of the watermark as a protection preserver, especially after geometric operations, the extraction mechanism can be undertaken in another way. Assume that a geometric operation that was used to modify an image can be identified, the inverse of the operation can be applied to revert the modified image back to its original. Therefore, instead of directly extracting from the modified image, the watermark could be revealed from the recovered one. This mechanism relies on the reverse function that is used to make the recovered image as close as possible to the original. The closer the recovered image is to the original one, the higher the similarity of the extracted signal to the original watermark.

Apparently, the mechanism to recover an image depends on the operation that was used to modify the image. For example, a cropped image can be recovered by first identifying the cropping coordinates and then replacing the missing parts with the corresponding pixels of the watermarked image. A rotated image can be recovered by applying the same degree of rotation with the reverse direction. A flipped image can be reverted back to its origin by implementing the same kind of flipping. In addition to previous recovery mechanisms, an image that was modified by adding noise, can be recovered by applying some filtering.

The recovery mechanisms significantly improve the quality of the extracted watermark signals. Table 5.7 shows that the signals extracted from each recovered image have a higher SSIM index. Images recovered from cropping, flipping and 90 degree rotation reveal perfect watermark signals (see Figure D.13f, D.15 and D.14i).

Table 5.7: SSIM of extracted watermarks from recovered images

Image Processing	Boat	Chilli	House	Lena	Mandril
Adding Noise	0.4799	0.5788	0.4808	0.5741	0.4347
Cropping 25 %	0.9355	0.9355	0.9355	0.9355	0.9355
Rotate 2^o	0.4739	0.4401	0.3911	0.4383	0.4754
Rotate 10^o	0.4133	0.3729	0.3479	0.3632	0.4161
Rotate 90^o	0.9355	0.9355	0.9355	0.9355	0.9355
Horizontal Flip	0.9355	0.9355	0.9355	0.9355	0.9355
Vertical Flip	0.9355	0.9355	0.9355	0.9355	0.9355

However, the signal that is extracted from the image recovered from 2 degree rotation, still has noise in it (Figure D.14c). This is because the recovered image is not as perfect as one that is recovered from the 90 degree rotation. This problem could be overcome by modifying the computation. The recovery process is not undertaken just by applying the same degree of rotation in the reverse direction, but also by finding the matching features between the rotated and original watermarked images. A different story comes from the image recovered from adding noise. Though the recovery mechanism succeeds in removing noise, the recovered image is a bit blurry because of the filtering process. Thus, it cannot reveal a perfect watermark signal (Figure D.13c). Nevertheless, the recovery mechanisms can be used to improve the capability of the watermark at preserving image protection. See Appendix D.3 for more visual presentations of the simulation result of this scheme.

5.5 Comparative Evaluation

This section provides a comparison between our proposed watermarking schemes and existing chaotic map based watermarking literatures. The literatures emerged in the early 2000s. In 2004 Dawei *et al.* utilised two logistic maps to construct a frequency domain based watermarking scheme. The first map is used to construct a sub image and the second to determine the watermark insertion locations inside the sub image's spectrum. Similar approach was also proposed by a number of researchers [65, 111, 122]. Another proposed scheme is applied in the spatial domain.

Wu *et al.* used a 2-D invertible chaotic map to determine watermark insertion locations [200, 199]. Utilising the 2-D map was intended to improve the success of watermark extraction. However, we investigate that Wu's technique is not always success to extract watermark because of placement conflicts. In our first proposed scheme, we overcome this problem. Instead of using a chain of dependant iteration rounds, we compute the watermark insertion location independently. Our technique keeps the natural property of the 2-D chaotic map, which is a one to one map, and thus, avoids the placement conflicts.

A number of watermarking techniques utilised chaotic maps to encrypt watermark before insert it into a host image. For instance, a watermarking scheme proposed by Munir *et al.* [123] employed two logistic maps to encrypt a binary watermark and then spread it into the host image spectrum using Cox's technique [45]. Munir's scheme is a non-blind watermarking and susceptible to cropping, resizing and adding noise. Another watermark modifications include mutation [12], permutation [192] and mixture [77]. All these modifications were intended to improve the security of watermark by obfuscating it.

Our watermarking schemes make use and combine the advantages of existing watermarking literatures. All the schemes utilise two or three logistic maps and one 2-D chaotic map — for encrypting and selecting the insertion locations of watermarks. The first two schemes are undertaken in the spatial domain and vulnerable to some geometric operations. The third scheme is done in the frequency domain to overcome the drawback of the previous schemes. This scheme employs two logistic map — one for encrypting watermark and the other for constructing a sub image — and one 2-D chaotic map for selecting watermark insertion locations.

The watermark insertion algorithm of our watermarking schemes require at least six parameters of the secret keys. The parameters include initial values of two or three logistic maps, three independent parameters (a, b and c) for the 2-D chaotic map, and the number of iterations (n). All of these parameters must be correctly

provided for the watermark extraction algorithm. If one fails to input any parameter, the watermark would be wrongly extracted. By this mechanism, our watermarking schemes achieve perfect security.

5.6 Chapter Remarks

Chaotic maps can be utilized to achieve perfect watermarking security because of their natural sensitivity to initial values. However, this characteristic is also a challenge for chaotic based watermarking schemes to achieve robustness.

With the aid of chaotic maps, three blind watermarking schemes are constructed. The experimental results show that the watermarking undertaken in the frequency domain is much more robust than the spatial domain schemes. The experimental results show that the frequency domain watermarking is robust against image enhancement, compression, adding noise and cropping. However, as with spatial domain schemes, rotating and flipping are still challenges for the frequency domain scheme. The signals extracted from the watermarked image after rotating or flipping suffer a significant degradation. In this case, a watermark can be extracted from the recovered image to improve the capability of the watermark to preserve image protection. In any future work, an improvement is needed to make this scheme more robust to withstand such geometric operations.

Robustness and security have to be balanced in a watermarking scheme. These properties are the success keys to optimize watermarking functionality. With nature of watermark which remains associated with the host content in its subsequent flow, digital watermarking is a potential tool to preserve protection of digital content. Digital watermarking may accomplish copyright protection in two approaches: preventive and curative, that is, before and after content decryption, respectively.

Watermarking alone, however, may be less useful. This technology needs to be integrated in a protection system. In term of preventive protection, for example,

watermark can be used to support content scrambling system (CSS) to make content duplication more difficult to do, or the functionality of an illegal copy reduced. In the context of curative approached protection, the capability of watermark can be improved by integrating it with traitor tracing schemes that will be discussed in the next chapter. In these schemes, watermark is used to build a content version. If pirate content is discovered, the version of such content is identified using watermark extracted from the pirate content.

Chapter 6

Deterring Traitors

6.1 Introduction

The aim of a protection mechanism in a content distribution system is to prevent unauthorised access to the distributed content. In some particular cases, however, legitimate users may aid a pirate. Such users are called traitors. They are likely to deny copyright and retransmit content or any compromised keys to unauthorised users. Tracing traitors can be the first aid to stop piracy in the system. Once a traitor has been discovered, content provider can take legal measures, such as, disconnecting him from further content distribution.

Keeping track of how content is being used and who is using it is an effective way to trace violations back to the misbehaving users. Taking these actions, however, causes DRM to deal with users' privacy concerns [124]. Tracing everything that users do with the content is obviously unacceptable from the users' point of view. Though content providers demand security, users need to have their privacy protected. Security and privacy in the traitor tracing context can be achieved when the tracing mechanism is capable of identifying the actual guilty users accurately without harming innocent users.

Traitor tracing scheme is a copyright infringement detection system which works

by tracing the source of leaked information. Tracing traitors may be operated statically or dynamically. In a static scheme, keys or marks are allocated only once and remain unchanged during the lifetime of the content. This model is appropriate when whole content is distributed in one package, such as a DVD movie. The tracing and incrimination algorithms are performed only when a black-market copy is discovered. A dynamic scheme changes keys or marks allocation at particular intervals of the content lifetime to adapt the real-time action of a pirate. This model is suitable when content is distributed online, e.g. in a pay TV broadcast. The pirate may rebroadcast the content, e.g. on the Internet. In this case, the tracing scheme relies on online feedback from the pirate subscribers to the content provider. To be effective, a traitor tracing scheme should be designed according to the piracy strategy it aims to counter.

6.2 Possible Ways for Piracy

Naturally, there are two strategies that are likely to be used by traitors to facilitate illegal access to a protected content — constructing pirate decoder and redistributing pirate copy of content [56].

6.2.1 Constructing Pirate Decoder

A group of users may subscribe to a content distribution system with a malicious objective. They aim to extract the decryption keys from their decoders and combine the keys to construct a pirate decoder that is then sold to unauthorised users [21, 36, 97, 103]. If content provider assigns the same key to all legitimate users, then such a malicious scenario would be completely risk-free for the traitors as their identities are not related to the decryption key within the illegal decoder. Even though the decryption keys are fingerprinted and bound to users' identities, the traitors may still attempt to obtain an untraceable working decryption key by combining their own

keys. In this case, with only an initial effort of the traitors, the content distributed by the system will be available in the black-market with no additional expense. This piracy strategy is more likely to occur in a broadcast encryption system, such as Pay-TV. This piracy strategy is less expensive, and thus, more likely to be chosen by the pirates to enable illegal mass-access to copyrighted content.

6.2.2 Redistributing Pirate Content

A traitor may first subscribe to a content distribution system. As a legitimate user, the traitor can decrypt the protected content and then redistribute the pirate copy of content to his own set of customers [62, 92, 167]. In a broadcast encryption system, this piracy scenario is less practical and costly because rebroadcasting content requires the maintenance of an independent broadcast infrastructure. Additionally, redistributing broadcast content has a higher perceptibility, and consequently, higher risk of being discovered. Redistributing content, however, is more likely to occur in a system that distributes content in physical media, such as DVD or blue ray disc format. An attacker may redistribute illegal copy of content, or in a worst case, he may redigitise the analogue output from a compliant device and redistribute the content in an unprotected form. In this case, the unprotected copy of content is the only forensic evidence for any copyright claim.

6.3 Tracing Pirate Decoder Constructors

Tracing traitors who construct pirate decoders is studied in the context of broadcast encryption [61]. In this context, the information is distributed securely to a dynamically changing privileged subset of users over an insecure network. A basic barrier is that the decryption key is initially split into several shares. Each authorised user has a share as their personal key. This mechanism aims at keeping the decryption key secret, while enabling users to decrypt the content using their traceable personal

key. If a pirate decoder is discovered, its constructors can be traced by identifying the keys used to construct the decoder. Thus, the security of the tracing scheme relies on how securely the decryption key is split.

6.3.1 User-Based Key Splitting

User-based key splitting was introduced in the first formal traitor tracing scheme [35, 36]. In this scheme, a message (m) is encrypted using a session key, s. To keep s secret, it is split into several shares $s_1, s_2, ..., s_{\lceil log_2 N \rceil}$, such that $s = s_1 \oplus s_2 \oplus ... \oplus s_{\lceil log_2 N \rceil}$, where N is number of users. Each share is then encrypted separately. The encryption of message and key's shares are done block by block; each encrypted block consists of an *enabling block* (EB) and a *cipher block* (CB) containing encrypted session key's shares and encrypted message, respectively.

In a typical construction, each share s_j is encrypted using two keys $k_{j,0}$ and $k_{j,1}$ so that the enabling block consists of $2\lceil log_2 N \rceil$ sub blocks [188]. Suppose there are N users,

$$U = \{u_0, u_1, u_2, ..., u_{N-1}\}$$

and $2\lceil \log_2 N \rceil$ keys,

$$\alpha = \{k_{1,0}, k_{1,1}, k_{2,0}, k_{2,1}, ..., k_{\lceil \log_2 N \rceil, 0}, k_{\lceil \log_2 N \rceil, 1}\}.$$

The personal key of user u_i is the set of $\lceil \log_2 N \rceil$ keys:

$$P_\alpha(u_i) = \{\ k_{j,b_{i,j}} \quad | \quad 1 \leq j \leq \lceil \log_2 N \rceil\ \}$$

where $b_{i,j}$ is the jth bit in the binary representation of i. The transmitted message is in the form of pairs of enabling block and cipher block,

$$(EB, CB),$$

where $CB = E_s(m)$ and

$$EB = \ ||_{j=1}^{\lceil \log_2 N \rceil} \ (\ E_{k_{j,0}}(s_j) \ || \ E_{k_{j,1}}(s_j) \ ),$$

where $||$ denotes concatenation.

In the decryption process, a user decrypts $\lceil log_2 N \rceil$ sub-blocks to obtain the shares. Combining the shares enables user to reconstruct the session key and use it to decrypt the cipher block. Each user can decrypt all s_j by deciphering a certain sub block of EB and hence obtain s. Since each user has a unique personal key, a user can be identified from the disclosed key.

A collusion of dishonourable users may leak information of their keys to an adversary so that he can construct a pirate decoder. Upon confiscation of the decoder, the traitor tracing algorithm is utilised to determine the identity of a traitor. To measure the resiliency of a system to collusion of traitors, Chor *et al.* [35] introduced the notion of *k-resiliency*. A scheme is *k-resilient* if given access to a pirate decoder built by combining at most k decryption keys, it is possible to trace, in polynomial time, at least one of the traitors that contributed to the decoder construction. In their scheme, there is a trade-off between the size of the enabling block and the number of keys held by each decoder. If the number of users is large, the schemes become impractical as the size of the enabling block grows proportionally with the number of users.

The public-key approaches [21, 103] attempt to have the size of the enabling block independent of the number of users with each decoder holding only one key. The collusion of k or fewer traitors who construct a single-key pirate decoder can be traced efficiently. These schemes, however, require higher computation time. Another public-key traitor tracing scheme, proposed by Tzeng *et al.* [191], has revocation capability. The enabling block of their scheme is independent of the number of users, but is dependent on the collusion and revocation thresholds, which are

k and z, respectively. That is, their scheme can find all traitors if the number of them is k or less (fully k-resilient), and revoke up to z traitors' private keys without updating any private key of the remaining users.

6.3.2 ID-Based Key Splitting

ID-based key splitting is another approach to share the secret decryption key by involving the users' ID, proposed by Joye and Lepoint [97]. The decryption key, d, is split into two components — σ_{ID} and d_{ID}. Personal string σ_{ID} is derived from a unique identifier ID, while the secret value d_{ID} is defined in such a way that $d = \mathcal{R}(\sigma_{\mathrm{ID}}, d_{\mathrm{ID}})$, where $\mathcal{R}$ denotes the combining function.

When a user wants to subscribe to the system, he has to provide a unique identifier ID. The system then computes $\sigma_{\mathrm{ID}} = f(\mathrm{ID})$. The function f can be an identity map, a cryptographic hash function, or a symmetric encryption function. The user then receives his personal string σ_{ID} together with a protected decoder containing the matching secret value d_{ID}. To decrypt the ciphertext c, the user inputs σ_{ID} into the decoder which then decrypts c in two steps:

- evaluate $d = \mathcal{R}(\sigma_{\mathrm{ID}}, d_{\mathrm{ID}})$;
- decrypt c using the key d.

The traitor tracing mechanism is simple. If an authorised user is allegedly suspected to have made illegal copies of his decoder, the pirate decoder can be examined to see if it relates to the user's ID.

6.4 Tracing Pirate Content Redistributors

Tracing traitors who illegally redistribute content is studied in the context of data fingerprinting [22]. In this context, traitors may combine their copies to construct

pirate content and distribute it. A trivial solution to trace the traitor who redistributes content is to assign a different watermark for each user. However, this solution requires high bandwidth for transmitting different copies of content to different users. To reduce bandwidth requirement, a watermark can be assigned to a set of users, rather than to an individual user.

6.4.1 Dynamic Tracing Scheme

The dynamic tracing scheme enables content provider to trace all traitors with much lower bandwidth [62]. In this scheme, content is divided into consecutive segments and each segment has q variations. In each round of broadcasting, the set of users are split into q disjoint subsets and each subset receives one variation of a segment. The subsets are changed in each interval using rebroadcasted content. If a rebroadcasted segment is found, its variation could be the evidence that the corresponding subset contains a traitor. The scheme then changes the allocations of variations to the users and starts a new round. It is assumed that there is an efficient group key management scheme that enables the broadcaster to efficiently regroup the users and securely deliver the allocated variation. Eventually, the collected information enables the scheme to identify and disconnect all traitors. The scheme implementation needs two broadcasting components — individual keys transmission before each segment and multiple variations broadcasting of every segment. The latter is a high overhead component as it multiplies the total bandwidth by the number of different variations.

The algorithm of the dynamic scheme was improved using an undirected graph representation [16]. In each round, the algorithm partitions the set of users into disjoint subsets, and gives all users in the same subset a common variation of the current segment from a set of variations C. If a subset is given variation $c \in C$, the the subset is said to be *coloured* by c. If a pirate broadcasts colour c, then the algorithm is said to receive as an *answer* the colour c. This answer indicates that

one of the users coloured by c is a traitor. A traitor is *identified* if only a single user is coloured by c, and c is given as an answer.

The dynamic tracing scheme has two drawbacks. Firstly, the scheme is susceptible to a *delayed rebroadcast attack* as regrouping users and allocating marks to users in each round depends on the rebroadcast content (a.k.a *feedback* from the channel). If an attacker rebroadcasts a variation with some delay, there is no feedback from the channel, and thus, no regrouping will occur. As a result, the broadcaster keeps the mark allocation unchanged and the system ultimately fails to trace the traitor. Secondly, dynamic tracing schemes require high real-time computation for regrouping the users and allocating marks to subsets. As a consequence, the length of a segment cannot be short.

6.4.2 Sequential Tracing Scheme

The sequential tracing scheme overcomes the dynamic tracing scheme's drawback by using a different mark allocation technique [167]. In this scheme, the channel feedback is only used for tracing traitors and not for allocating marks to users. Marks allocation in each interval is done according to a predefined table, regardless of the channel feedback. Even though the rebroadcast is delayed until the whole content is transmitted, at least one traitor will be traced. Traitors are traced sequentially, i.e., when a traitor is detected, he is disconnected from the system and the tracing mechanism continues to search the remaining traitors. Moreover, all computations related to key management of the group are also performed as pre-computation.

In this scheme, as in dynamic tracing, content is divided into segments. A q-ary watermarking system, $\mathcal{W} = \{1, 2, ..., q\}$, is utilised to create q variations of each segment. These variations are delivered to the set of users $U = \{u_1, u_2, ..., u_N\}$ according to a *mark allocation table* M. The mark allocation table $M = (m_{ij})$ is an $N \times L$ array over $\mathcal{W}$, where m_{ij} is the mark allocated to the user u_i in segment j and L is the *convergence length* of the tracing algorithm, that is, the number of

steps needed by the algorithm to find all traitors.

Sequential tracing scheme [167] runs a hybrid scenario. This scheme integrates previous models, i.e., allocates the keys or marks statically and traces traitors dynamically. This scheme overcomes the delayed rebroadcast attack and high real-time computations issues in the dynamic setting. However, the main tool of this scheme (the mark allocation table) is constructed based on the predetermined number of traitors t. If the traitors is at most t, the scheme successfully catches all traitors; otherwise, the tracing scheme fails.

6.4.3 Sequence Keys Scheme

The sequence keys scheme is a method to assign variants used in the Advanced Access Content System (AACS). The AACS is a content protection technology for the next generation of high-definition DVD optical discs including both the HD-DVD and Blue Ray disc formats [1]. Basically, the AACS' traitor tracing scheme has the same notion as those in the dynamic and sequential schemes, that is, it assigns a different version of content to a different user. A content version is produced from a sequence of segments' variants. In the AACS, however, content is not partitioned into segments, instead a segment is chosen from a particular point in the content, such as two seconds of movie. Therefore, all versions of a movie may differ at the selected segments and be identic at elsewhere.

The sequence keys scheme introduces *inner code* and *outer code* as the assigning tools. The inner code assigns variations of each segment and, thus, creates multiple movie versions for a movie. The outer code assigns different movie version to each user's device for a sequence of movies. Figures 6.1 and 6.2 illustrate inner code and outer code, respectively. Both the inner and outer codes are generated using *Reed-Solomon* (RS) codes [160]. For a q-ary RS code, the encoder takes k source symbols and appends parity symbols to make a code word of length $m = q-1$. Such an RS code is represented by a triple (m, q, k) and its minimum Hamming distance

Movie version	segment #0	segment #1	segment #2	segment #3	...	segment #13	segment #14
Ver 0	v 0	v 3	v 13	v 2	...	v 2	v 3
Ver 1	v 1	v 15	v 7	v 4	...	v 4	v 15
Ver 2	v 2	v 5	v 8	v 2	...	v 6	v 1
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
Ver 254	v 14	v 6	v 9	v 1	...	v 13	v 10
Ver 255	v 4	v 12	v 10	v 0	...	v 8	v 4

Inner code words

Figure 6.1: *Example of (15,16,2) inner code*

Device #	Movie #1	Movie #2	Movie #3	Movie #4	...	Movie #254	Movie #255
1	ver 0	ver 142	ver 213	ver 113	...	ver 153	ver 225
2	ver 1	ver 34	ver 57	ver 155	...	ver 122	ver 19
3	ver 2	ver 87	ver 18	ver 193	...	ver 125	ver 243
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
4294999999	ver 254	ver 26	v 9	ver 209	...	ver 81	ver 4
4295000000	ver 255	ver 94	ver 110	ver 195	...	ver 247	ver 131

Outer code words

Figure 6.2: *Example of (255,256,4) outer code*

between two code words is $d = m - k + 1$.

The inner code used in the AACS' sequence key scheme is $(15, 16, 2)$ [93]. It means that a movie has 15 segments and each segment has 16 variants. Thus, a movie has $16^2 = 256$ different versions and for any two different versions will differ at least in 14 segments. Because a movie has 256 versions, then 256 symbols can be used to construct the outer code. A symbol in an outer code word associates with an inner code word. The outer code used in AACS scheme is intended to assign a sequence of movies to at least 1 billion devices. For this need, the encoder takes 4 source symbols to make code words of length 255. Therefore, the outer code is $(255, 256, 4)$. As a result, the total number of outer code words is 256^4, meaning that the system can accommodate more than 4 billion devices. There will be 255

movies assigned to each device and any two different devices will have at most 3 movies of the same version (or the same inner code).

6.5 Double Encryption Scheme to Deter Piracy

In this section, a double encryption scheme is proposed to deter pirate decoder and pirate content. The double encryption scheme presented in this section is intended to improve the scheme that was previously proposed (see [153]), especially to make the scheme feasible for large-scale commercial use. This is because the protection mechanism in this scheme is applied to a single content item, instead of to a set of content items. The scheme prevents users from constructing pirate decoder and from constructing pirate content.

To deter pirate decoder, content is encrypted and the system gives each user a unique traceable personal key to decrypt content. If a pirate decoder is discovered, the key used by the decoder can be identified. To sign user's personal key, the ID-based key splitting [97], presented in subsection 6.3.2, is adopted with a modification. To avoid impersonation or re-obfuscation attacks, this scheme does not allow user to hold his personal key (string) σ_{ID}. Instead, σ_{ID} is stored inside an inaccessible area of the user's device and its associate secret value d_{ID} is stored in a smart card.

To deter pirate content, the system allocates a content version for each user. To construct content versions, m main segments are selected inside content. Each main segment has q variations. Each variation of each main segment is independently watermarked and encrypted. A sequence of the main segments' variations construct a version of content. For simplicity, this sequence is called the main code. If a pirate copy of content is found, its version can be identified from the watermarks extracted from its decrypted segments. In this work, *variation* refers to variant of segments, while *version* refers to variant of content. To sign variation of each main segment, the inner code of the sequence key scheme [91], as presented in subsection 6.4.3, is

adopted.

To make the scheme feasible for large-scale commercial use, the users are grouped. Any two users in a group receive different content versions, but two users from different groups may have the same content version. Suppose g is the length of the binary representation of the number of groups. To allocate a group to each user, g entry segments are chosen inside content. Each entry segment has two variations (0 and 1). Like the main segments, each variation of each entry segment is independently watermarked and encrypted. A sequence of the entry segments' variations identifies the group of the user. For shortness this sequence is called the entry code.

A segment is not an element of the content's partition, instead it is chosen from a particular point in the content. This segment determination aims to make the scheme more practical. For traitor tracing purposes, it is required that the number of variations for each segment cannot be big, but can address a large number of users; and the scheme should take no more than 10% of the space to store the variations. For example, assuming 10 segments per content, where each segment has 20 variations, the amount of data increases by roughly 5% [91].

The proposed scheme consists of two basic steps:

1. *Assignment* — content provider allocates a version of content and assigns a unique personal key to each user's device;

2. *Decryption* — users/decoders unlock the content and play it.

6.5.1 Assignment Step

At the assignment step, content is protected using double encryption (see Figure 6.3). First of all, each variation of each segment is separately encrypted (*inner encryption*). After all variations of all segments are encrypted, the whole content is then encrypted using *outer encryption*.

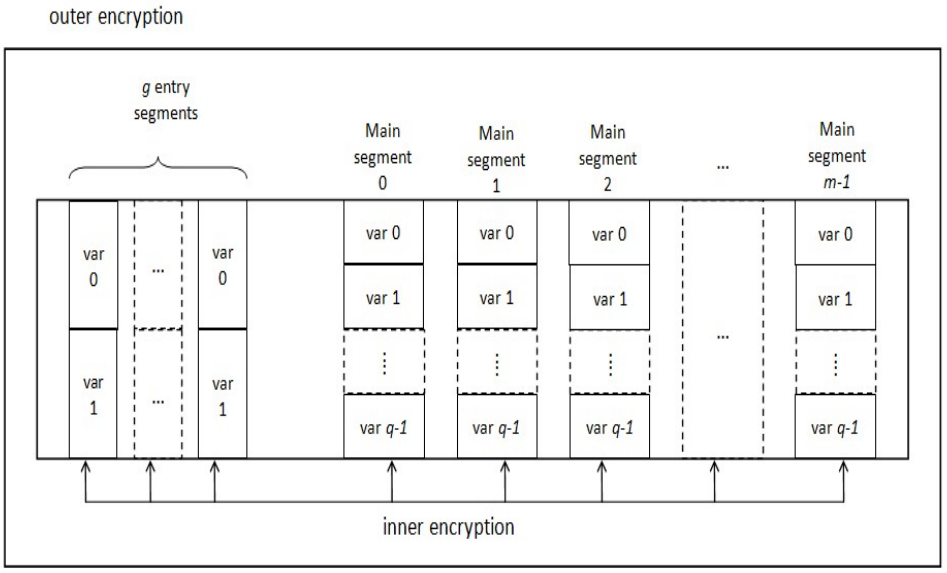


Figure 6.3: *Inner and outer encryption at the assignment step*

Inner Encryption

Inner encryption is separately applied to all variations of entry and main segments. This encryption is undertaken using symmetric cryptography and in conjunction with digital watermarking. First of all, after g entry segments are selected, a watermarking algorithm is used to insert 2 different marks into each entry segment, hence producing 2 variations of an entry segment. The watermarking system is assumed to be robust and secure, so that the inserted marks cannot be changed or removed. A chaotic-based watermarking, such as a scheme presented in subsection 5.4.3, could be used to achieve robustness and security that are required. After being watermarked, each variation of an entry segment is then separately encrypted.

A sequence of the entry segments' variations, called the entry code, determines a user's group. This code refers to the binary representation of the group. Figure 6.4 shows a simulation of the group allocation. In this simulation, the users are grouped into 65536 groups and therefore, 16 entry segments are needed to represent each group. For example, users of the group 2 and 65534 are assigned content with the entry code '0000000000000010' and '1111111111111110', respectively.

A similar mechanism is applied to the main segments. Once m main segments

User's group	entry segment #0	entry segment #1	entry segment #2	entry segment #3	...	entry segment #13	entry segment #14	entry segment #15
Group 0	v 0	v 0	v 0	v 0	...	v 0	v 0	v 0
Group 1	v 0	v 0	v 0	v 0	...	v 0	v 0	v 1
Group 2	v 0	v 0	v 0	v 0	...	v 0	v 1	v 0
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
Group 65534	v 1	v 1	v 1	v 1	...	v 1	v 1	v 0
Group 65535	v 1	v 1	v 1	v 1	...	v 1	v 1	v 1

Figure 6.4: *Example of variations allocation for each entry segment. A sequence of entry segments' variations (called the entry code) identifies the user's group*

are selected, a watermarking algorithm is employed to produce q variations of each segment. Each variation is then separately encrypted. A sequence of the main segments' variations, called the main code, constructs a content version. With m main segments and q variations per segment, one can construct q^m versions of content. However, there will be pairs of versions that differ only at one segment. This condition provides poor resistance against colluding attacks. By replacing only one segment, traitors can redistribute a version of the content such that an innocent third party appears to be the culprit. Indeed, this is an ideal situation from the attackers' perspective.

To avoid the incrimination of innocent users, the allocation of variations is done systematically based on a Reed-Solomon code (*RS-code*) [160]. With the alphabet size q, which is a power of a prime number, block length $m = q - 1$ and k source symbols, the (m, q, k)-code contains q^k code words with minimal distance $m - k + 1$. For example, suppose there are 15 main segments with 16 variations per segment and three data symbols. A $(15, 16, 3)$-*RS-code* will produce $16^3 = 4096$ versions of the content, and every two different versions will differ in at least 13 segments. This example is illustrated in Figure 6.5.

Content version	main segment #0	main segment #1	main segment #2	main segment #3	...	main segment #13	main segment #14
Ver 0	v 1	v 4	v 3	v 10	...	v 15	v 1
Ver 1	v 2	v 0	v 7	v 5	...	v 6	v 3
Ver 2	v 5	v 11	v 13	v 3	...	v 10	v 2
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
Ver 4094	v 6	v 15	v 8	v 9	...	v 14	v 8
Ver 4095	v 9	v 12	v 14	v 0	...	v 11	v 7

Figure 6.5: *Example of variations allocation for each main segment. A sequence of main segments' variations (called the main code) constructs a version of the content*

This assignment mechanism makes the scheme efficient. Content provider only needs to release the same copy of content master to all users, and each user's device will decrypt the master according to content version and group assigned to it. For example (see Figure 6.5), a user who is in group 65534 and assigned version 4094 will decrypt the entry segments through the sequence '111111111111110' and the main segments through the variants 6, 15, 8, 9, 6, 0, 1, 1, 14, 7, 15, 7, 0, 14 and 8. Note that all content versions may differ at the selected segments and be identical elsewhere. Furthermore, this assignment mechanism also makes the scheme feasible for large-scale commercial use. With 4096 content versions and 65536 groups of users, the scheme can cover 268,435,456 users. The data symbols used to generate *RS-code* could be increased to raise the content version. For example, a $(15, 16, 4)$-*RS-code* constructs 65536 version. With the same number of groups of users, this construction can cover more than 4 billion users.

Outer Encryption

Outer encryption is applied to the whole part of content after inner encryption has been completed. Suppose d is the decryption key. To keep d secret, it is split into

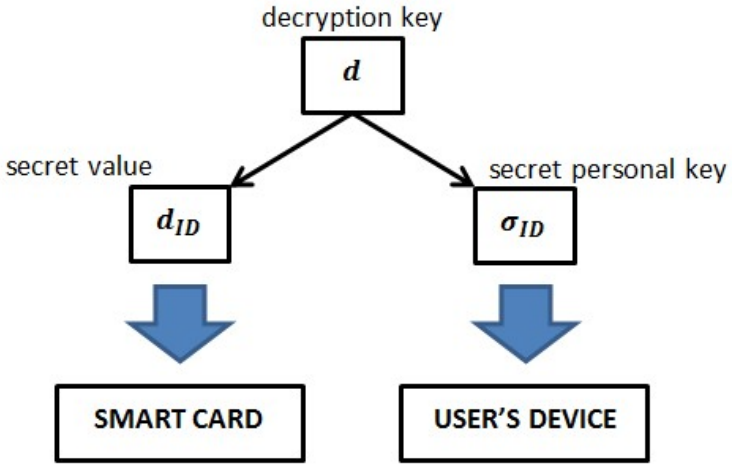


Figure 6.6: *Splitting the decryption key to keep it secret*

two shares (see Figure 6.6). One of the shares is derived from user's ID to guarantee that each user has a unique personal key. First of all, the system computes personal string $\sigma_{\mathrm{ID}} = f(\mathrm{ID})$. The function f can be an identity map, a cryptographic hash function, or a symmetric encryption function. For security, we suggest that f must be private (e.g. an encryption of AES).

The personal string is then used to compute secret value d_{ID}, such that

$$d = \mathcal{R}(\sigma_{\mathrm{ID}}, d_{\mathrm{ID}}).$$

The function $\mathcal{R}$ could be XOR operation or one of splitting mechanisms described in the Joye-Lepoint's scheme [97] — additive, multiplicative or Euclidean splitting. Consider an RSA cryptosystem [161] of modulus $n = pq$ where p and q are two large primes. A public encryption key e is coprime to $\varphi(n)$ and corresponds to a private decryption key d, such that $ed \equiv 1 \bmod (\varphi(n))$. $\varphi(n)$ denotes the Euler function and equals to $(p-1)(q-1)$.

1. In additive splitting, $d = \sigma_{\mathrm{ID}} + d_{\mathrm{ID}} \bmod \varphi(n)$.

2. In the multiplicative method, $d = \sigma_{\mathrm{ID}} d_{\mathrm{ID}} \bmod \varphi(n)$.

3. In Euclidean splitting, $d = \sigma_{\mathrm{ID}}.d_{\mathrm{ID}}^{(1)} + d_{\mathrm{ID}}^{(2)}$, where $d_{\mathrm{ID}}^{(1)} = \lfloor \frac{d}{\sigma_{\mathrm{ID}}} \rfloor$ and $d_{\mathrm{ID}}^{(2)} =$

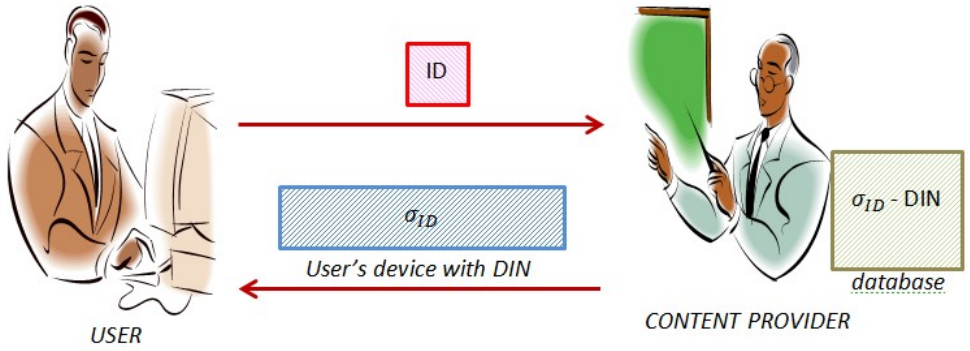


Figure 6.7: *Subscription process*

$d \bmod \sigma_{\mathrm{ID}}$.

Personal Key and Content Version Allocations

Allocating a unique personal key to a user is as follows (see Figure 6.7). When a user subscribes to the system, he has to provide a unique identifier ID. The system then computes σ_{ID} based on the user's ID. In return, the user receives a protected decoder containing his personal string σ_{ID}. This personal string is stored in an inaccessible area inside the decoder memory so that even the user will not know his σ_{ID}. This storing mechanism is intended to avoid impersonation attacks. The user knows only the decoder identification number (DIN). The DIN and its associated σ_{ID} are recorded in the content provider database for content purchasing purposes.

To allocate content version, a smart card is used to store a sequence of m variations' keys used to decrypt content's segments. Recall the example in Figure 6.5: Let $s_{i,j}$ denote the key to decrypt variation j of segment i. To allocate content version 4094, for instance, the smart card bears the sequence of variants' keys $s_{0,6}, s_{1,15}, s_{2,8}, s_{3,9}, ..., s_{13,14}$, and $s_{14,8}$.

A decoder with σ_{ID} stored in it can be used to decrypt multiple purchased contents. When a user purchases content, he has to provide his DIN. The system matches the DIN with the associated personal string σ_{ID} stored in the data base and then computes a secret value d_{ID} based on σ_{ID} and the content key, d. The

system also allocates the user a group, U_G, that is represented by the entry code. The secret value d_{ID} and the entry code U_G are stored in a smart card together with the sequence of the variants' keys, $s_{0,j_0}, s_{1,j_1}, s_{2,j_2}, ..., s_{m-1,j_{m-1}}$, which indicates the content version. With this mechanism, a content version will be uniquely assigned to a user.

6.5.2 Decryption Protocol

Decryption protocol is undertaken in the user's device (see Figure 6.8). In this protocol, user's device has two tasks — decrypt the encrypted content and play it. To decrypt an encrypted content, its associated smart card has to be connected to the user's device. The device then proceeds through the following steps:

1. reads d_{ID} inside the smart card;
2. computes $d = \mathcal{R}(\sigma_{\mathrm{ID}}, d_{\mathrm{ID}})$;
3. decrypts content using the key d;
4. plays the content by decrypting the entry and main segments according to the entry code U_G and the sequence of the main segments' variations (the main code), respectively.

6.5.3 Security Analysis and Traitor Detection

Withstanding Pirate Decoder

The scheme can perfectly prevent users from constructing pirate decoder. Assuming that σ_{ID} is stored in an inaccessible area of a tamper-resistant device, the scheme is secure against key extraction attacks, because even the user could not know his σ_{ID}. The associated secret value d_{ID} and the entry code U_G are also stored inside a smart card together with the content version keys at the time of purchasing

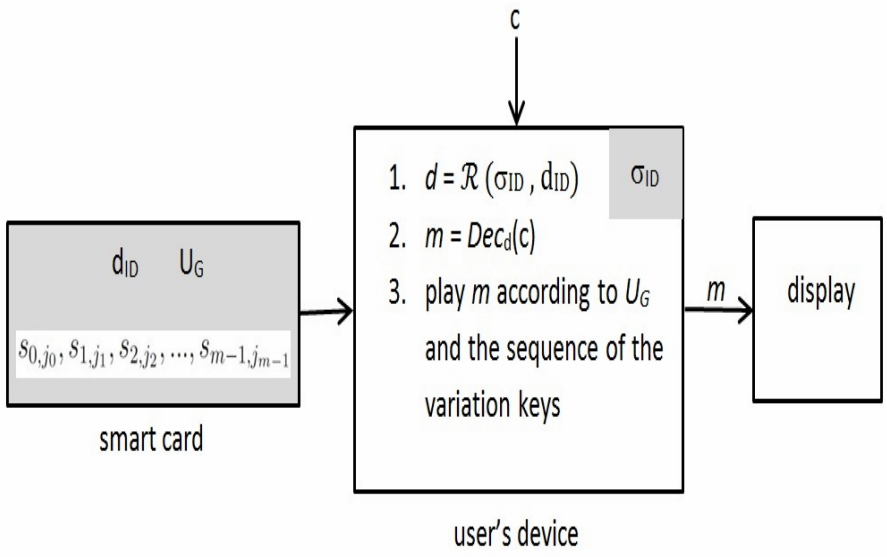


Figure 6.8: *Decryption protocol in the double encryption scheme*

content. With this mechanism, a particular version is only playable by the device with corresponding σ_{ID}. As both σ_{ID} and d_{ID} are inaccessible, the decryption key d is perfectly kept unrevealed. Additionally, in the decryption process, there is no opportunity for user to input anything, so that impersonation attacks can be avoided.

Even though a user may be able to obtain his σ_{ID} from his device with much effort, it will not be useful. This personal key is not a sensitive value and does not reveal any sensitive information about the secret value d_{ID}. A group of users may successfully extract their σ_{ID} and combine them. This attack is also useless. The knowledge of some σ_{ID} does not supply useful information to the coalition of users for learning the secret decryption key d_{ID} and/or d. The combination of some σ_{ID} may be similar to a registered σ_{ID}, but without knowing the corresponding DIN, the constructed decoder cannot be used to decrypt content. Even when the combination yields an unregistered σ_{ID}, the constructed decoder cannot be used even for purchasing legal content as it has no registered DIN. As a result, collusion attack does not apply in this scenario.

The only successful attack is when a user copies the memory of his decoder to

construct a pirate decoder. However, he has to use the same smart card for his pirate decoder to decrypt content. Moreover, content provider can compute d_{ID} from the pirate decoder as the provider has the secret d and, thus, reveals the σ_{ID}. Therefore, once a pirate decoder is discovered, the tracing algorithm will immediately point it out to the ID of the guilty user without harming any innocent user. Briefly, there is no way for any traitor to construct a pirate decoder.

Withstanding Pirate Content

Once the protected content is decrypted, the user's device will play it through the segments' variations allocated to the user. The sequences of the decrypted entry and main segments' variations will identify the user's group and the content version, respectively. The assignment mechanism (see subsection 6.5.1) guarantees that each content version is assigned to only one user in a group. Thus, overall, different users will play the content through different combinations of entry and main segments' variations.

Assuming that the watermarking system used to construct segments' variations is robust and secure, the inserted marks cannot be changed or removed. Additionally, using an error-correcting code to generate the main segments' variations makes every two versions of content differ at most segments. In this situation it will be hard for a coalition of users to combine their segments' variations to create an untraceable pirate copy or to counterfeit another users' content. Even though with much effort a coalition of users can construct pirate content, its version will be close to their versions. The revealed sequence of segments' variations from the pirate content will directly point out the real traitor without harming innocent users.

With regard to pirate content, the threat model considered in this scheme is as follows. Given two variations v_1 and v_2 of a segment, a pirate copy can be either v_1 or v_2 and no other variations can be generated [92, 94, 93]. This means that if some colluders can mix their segments and construct a content version, each segment's

variant of this version has to belong to at least one colluder. If pirate content is found, the simplest way to the traitor is the *scoring* method. That is, the sequence of the segments' variations of the pirate content is matched to the main code allocated to every user's device. Each user's device is scored based on how much its main code matches with the segments' variations of the pirate content. The device with the highest score, meaning its main code is the closest one to the pirate content version, is identified as a traitor. However, the traceability of the scoring scheme is limited because the false positive (the probability of identifying innocent users as traitors) of the tracing is high. A research result showed that using *mix-and-match* attack, a group of at least 20 colluding devices can successfully frame an innocent device [208].

Another traitor tracing mechanism that can be used to reduce the false positive is the *set-cover* scheme, which is claimed to be able to identify multiple traitors simultaneously and deduce the coalition size in the tracing procedure [93]. This scheme is looking for the smallest subset of users that matches with the pirate contents' version. Suppose the sequence of segments identified from the pirate content is $v_{0,j_0}, v_{1,j_1}, v_{2,j_2}, ..., v_{m,j_m}$. First of all, the set-cover scheme searches to find whether any single device has main code that matches with this sequence. If yes, that device's keys have been compromised with high probability. Otherwise, the scheme would search for coalitions of two devices that produced this content version. If the scheme does not find any coalition of size two, the coalition must be of size three, and so on.

The entry code might be easier to attack, because each entry segment has only two variations. If a user can replace even only one entry segment in his content, another user from the other group would be counterfeited. The security of the entry code does rely on the security and robustness of the watermarking scheme. Utilising chaotic based watermarking, such as presented in subsection 5.4.3, makes the insertion locations of the watermarks undetectable by unauthorised persons. As

a result the selected segments are also undetected and thus, it is hard for the user to change even a single segment.

6.5.4 Users' Privacy Analysis

Recall the mechanism for allocating the personal key presented in subsection 6.5.1. A user needs to provide his ID only once, that is, when he subscribes to the system. The system then computes a personal string σ_{ID} based on the ID. The user receives a protected decoder containing σ_{ID}. This personal string is stored in an inaccessible area so that even the user will not know his σ_{ID}, but only the decoder identification number (DIN). The DIN and its associated σ_{ID} are recorded in the content provider database for the purpose of content purchasing. Every time the user purchases an item of content, he only needs to provide the DIN. With this mechanism, the user's identity will not be connected to the purchased item, as long as there is no violation in using the item. However, if an allegedly pirate content or decoder is found, the system will run the traitor detection mechanism (described in subsection 6.5.3) and can reveal a user's identity behind this piracy.

Indeed, for the sake of traitor tracing, the system needs to collect the users' information. However, to protect users' privacy, the information is collected once and safely kept in the content provider's data base. For subsequent transactions, the system only needs to deal with the decoder identification number (DIN) and the personal string σ_{ID}. A user's information will be recalled from the database only when the user is alleged to do a violation.

6.6 Comparative Evaluation

Generally, in existing literatures, a traitor tracing scheme is designed to trace pirate decoder only or pirate content only. Our scheme is intended to deter both pirate decoder construction and pirate content redistribution. To deter pirate decoder,

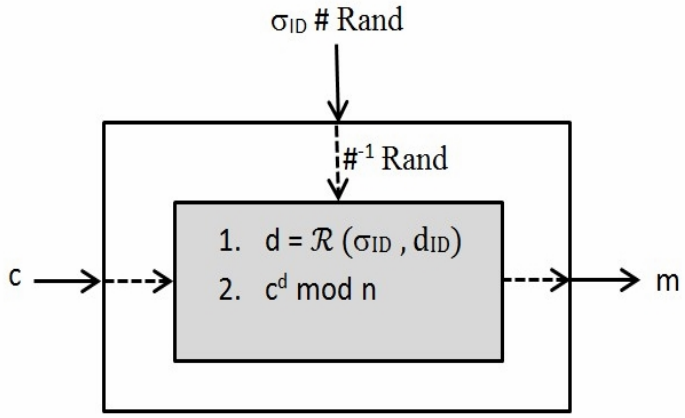


Figure 6.9: *1-traitor threat model*

the system gives each user a unique traceable personal key. To sign user's personal key, we adopt the ID-based key splitting scheme proposed by Joye and Lepoint [97], as presented in subsection 6.3.2, with a modification. To deter pirate content, the system gives each user a content version. Content versions are constructed by firstly selecting m main segments inside content. Each main segment has q variations. To sign variation of each main segment, we adopt the inner code of the sequence key scheme [91], as presented in subsection 6.4.3.

The ID-based key splitting scheme was patented in 2013 [98]. However, the scheme can be broken even by a single traitor. A malicious user may use his original decoder to construct a tamper-resistant pirate decoder (Figure 6.9). He puts the original decoder inside the pirate one and creates a new personal string, $\sigma^*_{\mathrm{ID}} = \sigma_{\mathrm{ID}} \# Rand$, that is, a combination of the original personal string and a random one. Symbol # represents a binary operation, such as XOR, addition or multiplication. The input of the pirate decoder includes the encrypted content c and σ^*_{ID}. The decoder then computes σ^*_{ID} $\#^{-1}$ $Rand$. Symbol $\#^{-1}$ represents the inverse of the binary operation used to create σ^*_{ID}. If # is XOR, addition or multiplication, then $\#^{-1}$ will be XOR, substraction or division, respectively. These operations exactly reconstruct σ_{ID}. Both c and σ_{ID} are then passed to the original decoder for the normal process. The main strategy of this threat model is to obfuscate the original personal string σ_{ID} outside the pirate decoder and decrypt the content using an

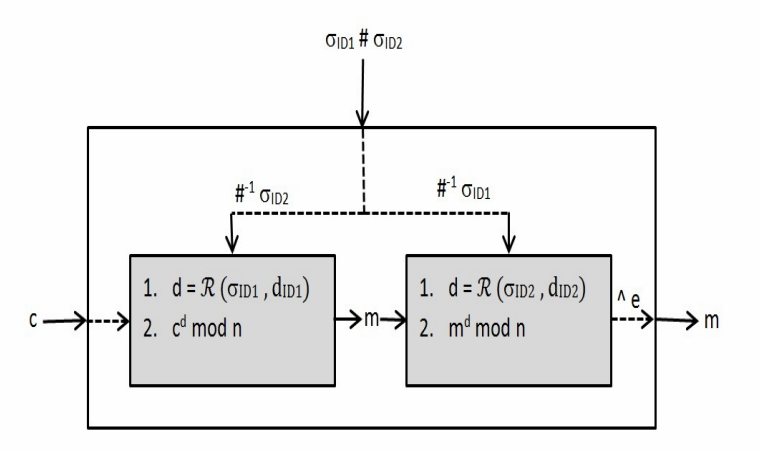


Figure 6.10: *2-traitor threat model*

untraceable personal string.

The same threat strategy can also be used by a coalition of two traitors. Two users, u_1 and u_2 may combine their decoders, Decoder_1 and Decoder_2, respectively, to construct a pirate decoder (Figure 6.10). They also combine their personal string, σ_{ID_1} and σ_{ID_2}, to construct an obfuscated string

$$\sigma^*_{\text{ID}} = \sigma_{ID_1} \ \# \ \sigma_{\text{ID}_2}$$

as an input for the pirate decoder.

Actually, Joye and Lepoint have considered these threat models. They called these attacks impersonation or re-obfuscation attacks. They argued that such attacks are readily ruled out in semi-open environments, that is, executing only signed code. Moreover, the attacks will not work if there is a global integrity check of the original decoder, that is, the software decoder detects that it is part of another program and takes appropriate actions. However, most protection systems, such as Digital Rights Management, are often running in an open environment, where users have full control over the system. A factor in Joye-Lepoint's scheme which opens the possibility of infringement is that it allows the user to enter a personal key when decrypting content. This mechanism is vulnerable to impersonation attack. Therefore, storing the personal decryption key in a tamper-resistance device,

disabling user from manually entering the key in the decryption process, will reduce the opportunity of such an attack.

To avoid impersonation or re-obfuscation attacks, our proposed scheme does not allow user to hold his personal key (string) σ_{ID}. Instead, σ_{ID} is stored inside an inaccessible area of the user's device and its associate secret value d_{ID} is stored in a smart card.

The sequence key scheme [91] relies on providing a number of items (e.g. 255 movies) to cover a huge amount of users. This is because a single movie has only 256 versions and, for the sake of traitor tracing, it can only be assigned to at most 256 different users. Furthermore, if a violation occurs, this scheme relies on a number of allegedly pirate items. That is, if a set of pirate movies is found, the versions of these movies are matched to the outer code of every user's device. Thus, finding only a single pirate movie might not give a clue about the real traitor. However, in real applications a user may not subscribe to such a big number of movies.

Our proposed double encryption scheme would be more flexible because it provides protection for a single movie. An allegedly pirate movie can release a clue about the traitor. Of course, with a few main segments, the scheme can only cover a few number of users. To make the scheme feasible for large-scale commercial use, the users are grouped. Any two users in a group receive different content versions, but two users from different groups may have the same content version. For this reason, we also select g entry segments inside content where each entry segment has two variations (0 and 1).

6.7 Chapter Remarks

The ideas of traitor tracing schemes are combined to construct a traitor deterring mechanism. This mechanism is perfectly able to prevent coalition attacks from constructing an untraceable pirate decoder or counterfeiting another user. With a

secure and robust watermarking, the mechanism also discourages user from making an illegal copy of content. If a user does so, the identified pirate content version will immediately point to him as the real traitor.

In the traitor tracing perspective, the content provider's security and the users' privacy can be equally achieved when the tracing mechanism is capable of identifying the actual guilty users accurately without harming innocent users. For this tracing purpose, indeed, the system needs to collect the users' personal information. To protect users' privacy, the users' information is collected once and safely stored in the database. The user's information is not connected to the purchased content and remains unrevealed unless the user is alleged to commit a violation.

Based on the capability of the proposed scheme, it can be concluded that the scheme preserves security for the content provider and privacy for the users. The scheme is able to prevent both pirate decoder and pirate content. However, if a violation occurs, the real source of it will accurately be identified without framing innocent users. This mechanism has a potential to improve DRM for the content distribution system that is required to provide balanced protection for content provider and users.

Chapter 7

Conclusion

7.1 Synopsis

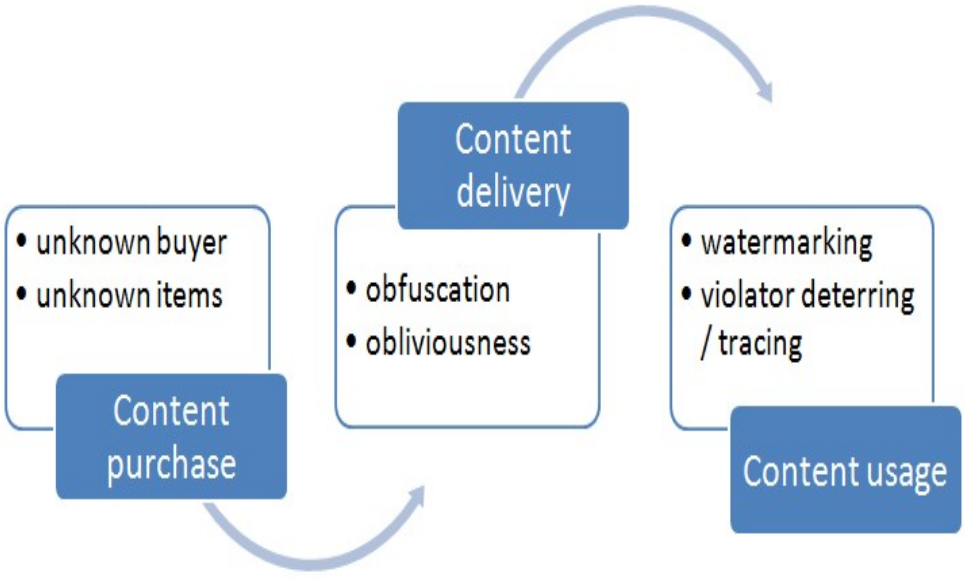


Figure 7.1: *Privacy-aware protection along the content life time*

The major problems for the DRM system are related to the need for balanced protection of both the content provider's security and the users' privacy. These security and privacy aspects need to be considered equally in each protection scenario throughout the content's life time (see Figure 7.1). First of all, the content provider's security and users' privacy can be achieved equally at the time of con-

tent purchase. At this stage, users' privacy protection is approached by minimising personal data acquisition. However, while the system allows users to maintain their privacy, it must prevent any malicious action that disadvantages the content providers' security. Before content reaches end users, the protection scenario is focused on securing content delivery while still preserving users' privacy. When content reaches the end users, preserving content protection is essential. The integration of digital watermarking and traitor tracing scheme is viewed as a potential tool to preserve protection.

The protection scenarios at the time of content purchase and content delivery have the same characteristics. In these scenarios, a user can purchase content anonymously or blindly, meaning the content provider will not know who is purchasing or which content is being purchased, respectively. Briefly, the system does not connect the user's identity to the purchased content. Using this mechanism, while the system protects the users' privacy, it still maintains the content provider's security.

The protection scenario when content has reached the end user, however, seems to have contrary characteristics. In this scenario, the system needs to collect information about users that might ultimately be connected to the content they access for the sake of traitor tracing. Though the users' personal data acquisition is urgently needed in this protection scenario, the system has to use the data appropriately. The user's data must not be connected to the accessed content and remains unrevealed unless the user is alleged to have violated copyright.

7.2 Applications

This section provides some applications of the protection schemes developed in this research.

7.2.1 Online Transaction

The improved anonymous cash and blind decryption schemes are intended to protect users' privacy when they purchase digital content online. The content includes image, movie, music, electronic book and application software. The blind decryption scheme might be preferable as it is more efficient than the anonymous cash scheme. Sections 3.4 and 3.5 provide some scenarios on how to implement the blind decryption scheme when items' prices vary and buyer authorisation is applied in the transaction stage, respectively.

7.2.2 Off-line Content Distribution

The advantages of smart card helps the white-box implementation and the oblivious content distribution (OCD) schemes to equally achieve security for the content provider and privacy for the users in an off-line content distribution system. The distribution of digital images, movies, music, e-book and application software could be covered by such schemes.

The white-box model is similar to the DRM threat model, in the assumption that the data, programs and devices that are used to access content are under control of the adversary. A persistent protection can be achieved by encrypting content and deploying tamper resistant devices. Subsection 4.2.5(1) describes the white-box implementation in off-line content distribution. In this case, the users' privacy protection relies on the anonymity of purchasing the smart card.

The users' privacy protection can be improved when the content provider provides a package of content items and the user can select some of the items. Though purchasing the smart card might be done non-anonymously, the content provider will not know which items are selected by the user. This scenario is accomplished by the OCD scheme. Subsection 4.3.7(1) describes the application of this scheme in an off-line content distribution. In this application, the content provider's security can be preserved, since the user cannot select more items than he purchased.

7.2.3 Online Content Distribution

Though, with the aid of smart card, the white-box implementation and the OCD schemes are primarily applied in an off-line distribution, the scenarios of these schemes can be adopted by an online content distribution. In the context of an online content distribution scenario, the role of smart cards could be filled by a secure distributor server. Subsections 4.2.5(2) and 4.3.7(2) describe on how to apply the white-box implementation and the OCD scheme, respectively, in online distribution.

7.2.4 Controlling Access to Movies

The assignment step and the decryption protocol of the double encryption scheme (mentioned in section 6.5) are designed to enforce conditional access and use of content consisting of segments, such as movies. A version of the movie can only be decrypted and played by a single device. The device containing the user's personal key (string) σ_{ID} can decrypt the movie with the aid of a smart card containing the associated secret value d_{ID}, such that the combination of σ_{ID} and d_{ID} yields the correct decryption key. Therefore, the smart card cannot be used to decrypt the movie in other devices. As a result, each user has to purchase an appropriate smart card.

7.2.5 Deterring Violations

Besides enforcing conditional access, the scenario presented in subsection 7.2.4 is also discourages users from transferring their content to unauthorized users. Additionally, in this scenario, besides d_{ID}, the smart card also contains the entry code and the main code that represent the user's group and the content version assigned to the user. After decrypting content, the device will play it by decrypting a sequence of the variations of the entry and main segments according to the entry and main codes, respectively. If a user illegally duplicates the clear content, then

the user's group and the content version can be identified from the sequence of the decrypted segments' variations revealed from the pirate content. This mechanism will directly point out the real traitor.

Unlike the content transaction and distribution, in which the protection scheme does not connect the user's identity to the accessed content, the applications for deterring or tracing traitors need to record the identity of users and content they access. However, to protect the users' privacy, the users' identity should be collected once and not be used for the subsequent transactions. This scenario is applied by the double encryption scheme (section 6.5), in which a user needs to provide his identity only once, that is, when he is subscribing to the system. In the subsequent transactions, the user needs only to provide his decoder identification number (DIN). The user identity will not be released unless he is alleged to commit a violation.

7.2.6 Preserving e-Books Users' Privacy

The double encryption scheme can be adopted to preserve users' privacy in an e-book distribution system. Refer back to the Amazon Kindle store (see subsection 2.3.3). To be able to purchase an e-book from this store, a user is required to have an account in the Amazon web site, install the Amazon Kindle e-book reader and register it under his account. The users must provide their accounts every time they purchase e-book.

The users' privacy preservation could be improved when they do not need to provide their accounts at every purchasing time. Instead, they may only need to provide their Kindle reader's serial number, just like the content transaction scenario presented in the double encryption scheme (section 6.5).

7.3 Limitations and Open Problems

7.3.1 Limitations

1. To achieve the goals, some proposed schemes, such as white-box implementation based content distribution scheme, oblivious content distribution scheme and double encryption scheme to deter piracy, relies on the aid of smart cards. The smart cards are used to store the essential parts of the content key and perform the main operation to reconstruct the key. This means that the smart card helps to protect content provider's security. On the other hand, users can anonymously purchase a smart card corresponding to their chosen content so that their identity will not be connected to the content they access, and thus, their privacy is protected. The construction of the proposed schemes was motivated by the theoretical smart card's advantages. However, a cheap smart card technology may not be available immediately.

2. Chaotic based watermarking provides a perfect watermark security. This is because of the chaos' extreme sensitivity to initial conditions and the outspreading of its orbits over the entire space. These features, however, also challenge the robustness of the watermark. A frequency domain based watermarking exhibits a more robust watermark, but still has to struggle against geometric operations.

7.3.2 Open Problems

1. How to implement white-box based content distribution scheme, oblivious content distribution scheme and double encryption scheme without depending on the physical security (i.e. the use of smart cards).

2. The chaotic map based blind frequency domain watermarking needs to be improved to make the watermark stronger to withstand geometric operations.

Appendices

Appendix A

UCON Model for DRM

Current usage control perspective is a broader view of the traditional access control viewpoint. The traditional access control was intended to allow content providers to selectively determine who can access which content and what access is exactly provided. Regardless the user is currently accessing the content or not, the right exists and enables multiple access until it is explicitly canceled. In this case, the control relies only on the authorization aspect which evaluates access requests based on user attributes, content attributes and requested rights. In addition to authorization, the modern information systems require more aspects including obligation and condition [140]. Obligation is the requirement that has to be fulfilled by users for usage allowance. For example, to access digital resources, someone may have to fill out a certain form or accept the licence agreement. Condition includes the environmental and system requirements that have to be satisfied for accessing content. For example, some digital resources may be playable only on a certain device or location in a certain period of time.

Usage Control (UCON) does not come to replace the traditional access control, but rather encompasses it and goes beyond its capability. A basic usage control framework, UCON_{ABC}, was introduced by Park and Sandhu [139], integrates three aspects: Authorization (A), oBligation (B) and Condition (C). Authorizations are functional predicates that have to be evaluate for usage decision and return whether the subject is permitted to perform the requested rights on the object. Obligations are functional predicates that verify mandatory requirements a subject has to perform before or during a usage exercise. Conditions are environmental or system-oriented decision factors. These predicates evaluate current environmental or system status to check whether relevant requirements are satisfied and return either *true* or *false*.

A view of the UCON_{ABC} model (see Fig. A.1) focuses on the relations among the components. Usage decision is shown as a relationship among subjects, objects and rights that requires authorizations, obligations and conditions. Another innovation in this model is that subject and object attributes can be mutable. Mutable attributes are changed as a consequence of access, while immutable attributes can be changed only by administrative action.

Usage decisions in the DRM solutions usually make use of user-defined, application-level, payment-based security policies and do not include traditional access control

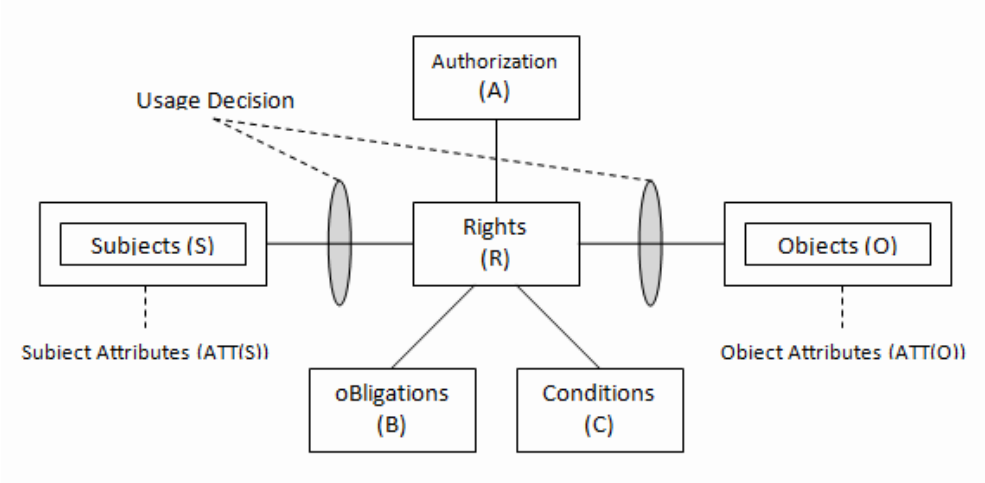


Figure A.1: *A view of* $UCON_{ABC}$ *[139]*

policies. To clarify the implementation of the $UCON_{ABC}$ model in DRM, Park *et al* [139] provided some examples. For all following examples, S,O and R stand for Subject, Object and Rights, respectively.

Example A.1. DRM pay-per-use*:*
M is a set of money amount
$credit : S \rightarrow M$
$value : O \times R \rightarrow M$
$ATT(S) : \{credit\}$
$ATT(O, R) : \{value\}$
$allowed(S, O, R) \Rightarrow credit(S) \geq value(O, R)$
$preUpdate(credit(S)) : credit(S) = credit(S) - value(O, R)$

In Example A.1, a requested usage is permitted if the subject has adequate prepaid credits to use certain rights on specific objects. The credit is considered as a subject attribute and the value of the requested usage as an attribute of the object and right. An update procedure is then required in a payment-based authorization to determine usage cost. In this case, the subject's credit is reduced by the value of usages once the request is approved.

Example A.2. DRM pay-per-use, one credit, multiple values*:*
M is an ordered set of money amount
$credit : S \rightarrow M$
$value : O \times R \rightarrow 2^M$
$ATT(S) : \{credit\}$
$ATT(O, R) : \{value\}$
$\overline{M} = \{m | m \in value(O, R), m \leq credit(S)\}$ *is a set of available values for selection*
$allowed(S, O, R) \Rightarrow \exists m \in value(O, R), m \leq credit(S)$
$preUpdate(credit(S)) : credit(S) = credit(S) - \lambda(\overline{M})$*, where* λ *is a selection function to select a value for update.*

Example A.2 illustrates the case when a usage of an object has more than one value. In this case, the system has to select a value for update. The selection may

be based on subject's membership ranks, sale period or multiple purchases. The selection policies can vary. Utilizing a nondeterministic function in this example is to simplify the selection functionality.

Example A.3. DRM pay-per-use, multiple credit, one values*:*
M *is an ordered set of money amount*
$credit : S \rightarrow 2^M$
$value : O \times R \rightarrow M$
$ATT(S) : \{credit\}$
$ATT(O, R) : \{value\}$
$\overline{M} = \{\overline{m} | \overline{m} \in credit(S)\}$
$\widehat{M} = \{\widehat{m} | \sum \widehat{m} = value(O)\}$
$\lambda : \overline{M} \rightarrow \widehat{M}, \overline{m} \geq \widehat{m}$
$allowed(S, O, R) \Rightarrow \sum credit(S) \geq value(O, R)$
$preUpdate(credit(S)) : \forall \overline{m}, \overline{m} = \overline{m} - \lambda(\widehat{m})$.

Example A.3 shows a case when a user has more than one credit account. The request will be permitted if the sum of credits is more than a value. In the updating process, some or all of the credit accounts have to be reduced in total by the usage value.

Appendix B

Abstracts of Published Contributions

B.1 DRM's Rights Protection Capability: A Review

Digital Right Management (DRM) is a digital content providers' weapon to protect their intellectual property rights. The use of DRM has been rising in recent years. However, though digital technologies have changed enormously in recent decades, many existing DRM technologies only provide partial solutions to an immense problem and thus are still not effective at combating piracy. Additionally, they fail to maintain fair use and protect users' privacy. This paper provides a literature review on the current state of DRM research and a major problem for the DRM system.

Proceedings of the First International Conference on Computational Science and Information Management (ICOCSIM). Volume 1 (2012), pp. 12-17. Medan, Indonesia (December 2012). ISBN 978-967-0120-60-7.

B.2 Obfuscation and WBC: Endeavour for Securing Encryption in the DRM Context

Under Digital Rights Management (DRM) protection, digital content is usually delivered in an encrypted form so that only authorized users can access it. Most DRM applications use secure symmetric algorithms, such as DES and AES, to protect content. However, executing these algorithms in an insecure environment may allow adversaries to compromise the system and obtain the key. To withstand such attack, algorithm implementation is modified in such a way to make the implementation unintelligible, namely obfuscation approach. The two most considered obfuscation techniques are code obfuscation and white-box cryptography. In this paper we study the notions of code obfuscation and white-box implementation with regard to the DRM context. We then propose an implementation of white-box cryptography to advantage DRM which is required to provide a balanced protection for content provider and users.

Proceedings of the International Conference on Computer Science and Information Technology. CSIT 2013, pp. 150-155. Yogyakarta, Indonesia (June 2013). ISBN 978-979-3812-20-5 2.

B.3 Secure and Private Content Distribution in the DRM Environment

Digital Rights Management (DRM) is required to provide balanced protection for both the content provider and the users in a content distribution system. The content provider demands secure content delivery so that only authorized users are able to access the content and use it properly. On the other hand, users require that their privacy be protected. However, most DRM systems tend to put greater emphasis on content providers security and neglect users privacy. This study aims to improve DRM by constructing a content distribution protocol that preserves the security of content provider and the privacy of users. To achieve this goal, we utilize the oblivious transfer (OT) concept. This concept allows a sender to securely send a set of information to a receiver in such a way that, at the end of the protocol, the receiver cannot learn more than he was supposed to learn, while the sender cannot determine what the receiver has learned. Assuming that tamper-proof device exists, the constructed protocol achieves perfect security for the content provider and privacy for the users. This oblivious content distribution ultimately enables DRM to be a privacy-aware protection system. The system does not merely focus on content providers rights, but also seriously considers users privacy protection.

Proceedings of the Information System International Conference. ISICO 2013, p. 659-664. Bali, Indonesia (December 2013). ISBN 978-979-18985-7-7.
Available online at Open Access Journal of Information Systems (http://is.its.ac.id/pubs/oajis/index.php/home/detail/1156/ Secure-and-Private-Content-Distribution-in-the-DRM-Environment)

B.4 Deterring Traitor Using Double Encryption Scheme

Traitor is a legitimate user in a content distribution system who facilitates illegal content access to unauthorized party. Naturally, there are two strategies that are likely used by the traitors to do violations. Firstly, the traitors may attempt to extract the decryption keys to construct a pirate decoder. Secondly, they may legally decrypt the content and then illegally redistribute the decrypted content for their own benefit. We propose a content protection scheme to deter both piracy strategies. However, if a pirate decoder or content is discovered, the scheme is still able to identify the real traitors without incriminating innocent users. Thus, our scheme preserves security for the content provider and privacy for the users in the traitor tracing context.

Proceedings of the IEEE International Conference on Communication, Network and Satellite. COMNETSAT 2013, p. 100-104. Yogyakarta, Indonesia (December 2013). ISBN 978-1-4673-6054-8.
Available online at IEEE Xplore Digital Library. INSPEC Accession Number: 14515736; DOI: 10.1109/COMNETSAT.2013.6870869.
(http://http://ieeexplore.ieee.org/xpl/articleDetails.jsp?arnumber=6870869)

B.5 Blind Image Watermarking Based on Chaotic Maps

Security of a watermark refers to its resistance to unauthorized detecting and decoding, while watermark robustness refers to the watermark's resistance against common processing. Many watermarking schemes emphasize robustness more than security. However, a robust watermark is not enough to accomplish protection because the range of hostile attacks is not limited to common processing and distortions. In this paper, we give consideration to watermark security. To achieve this, we employ chaotic maps due to their extreme sensitivity to the initial values. If one fails to provide these values, the watermark will be wrongly extracted. While the chaotic maps provide perfect watermarking security, the proposed scheme is also intended to achieve robustness.

Presented at *the IEEE International Conference on Information Technology and Applications*. ICITA 2014. Sydney, Australia (July 2014). ISBN 978-0-9803267-6-5. To be published at *IT in Industry Journal*, vol. 2, no. 2, 2014, pp. 44-50. ISSN 2203-1731

Appendix C

MATLAB scripts of the Watermarking Schemes

C.1 Watermark Insertion of the First Scheme

```
% winsert1.m
% inserting binary watermark based on chaotic map and Arnold map
% ------------------------------------------------

clear
clc

% select host image
[fname1,pthname1] = uigetfile('*.tiff;*.png;*.jpg','Select the host
image');
% select watermark image
[fname2,pthname2] = uigetfile('*.tiff;*.png;*.jpg','Select the
watermark image');

% read host image
I_ori = imread([pthname1,fname1]);
% gray scale host image
I_gray =  I_ori(:,:,1);
[irow,icol]=size(I_gray);

% read watermark image
W = imread([pthname2,fname2]);
% convert watermark image into binary image
W_bw = im2bw(W);

[wrow,wcol]=size(W_bw);

% INPUT
x10 = input('initial value of Log Map 1 = ');
```

```
x20 = input('initial value of Log Map 2 = ');

a = input('the first independent parameter = ');

b = input('the second independent parameter = ');

c = input('the third independent parameter = ');

n = input('the iteration number of the Arnold map = ');

% STEP 1 ENCRYPT BINARY WATERMARK

% generate chaotic map

L1=logmapf(x10,4,wrow*wcol); m = mean(L1);

% convert to binary
for i=1:wrow*wcol
    if L1(i) >= m
        L1_bin(i) = 1;
    else
        L1_bin(i) = 0;
    end
end

%reshape L1_bin into matrix 64 x 64
L1_binmat = reshape(L1_bin,wrow,wcol);

% encrypt W_bw by xoring it with L1_binmat
W_enc = bitxor(W_bw,L1_binmat);

% STEP 2 DETERMINE INSERTION PLACE

I_marked = I_gray;

A=[a b;c (1+b*c)/a];

for i=1:wrow
    for j=1:wcol
        p0 = [i;j];
        it = 1;
        while it <= n
        q = mod(A*p0,irow);
        for t=1:2
            if q(t)==0
```

```
                q(t)=irow;
            end
        end
        p0=q;
        it=it+1;
        end
        pv = I_marked(q(1),q(2));
        binpv = de2bi(pv,8);
        L2 = logmapf(x20,4,n);
        if L2(n) > 0.75
            k=4;
        else if L2(n) > 0.5
                k=4;
        else if L2(n) >0.25
                k=5;
            else
                k=5;
            end
            end
        end
        newbinpv = binpv;
        newbinpv(k) = W_enc(i,j);
        newpv = bi2de(newbinpv);
        I_marked(q(1),q(2))= newpv;
        x20 = L2(n);
   end
end

I_mark = I_ori;
I_mark(:,:,1) = I_marked;

% save watermarked image into files
name = regexp(fname1,'\.','split');

wat_img_name = strcat(name(1),'marked.',name(2));
imwrite(I_mark,char(wat_img_name));

% display original image
subplot(1,2,1), imshow(I_ori), title('original host image');
% display watermarked image
subplot(1,2,2), imshow(I_mark), title('watermarked image');

ssim = ssim_index(I_gray,I_marked)
```

C.2 Watermark Extraction of the First Scheme

```
% wextract1.m
% extracting binary watermark based on chaotic map and Arnold map
% watermark must be square
% ------------------------------------------------

clear
clc
% select test image
[fname1,pthname1] = uigetfile('*.tiff;*.png;*.jpg','Select the test
image');

% read the test image
I = imread([pthname1,fname1]);
% extract the gray scale part of the test image
I_size = imresize(I,[256 256]);
I_gray = I_size(:,:,1);
[irow,icol]=size(I_gray);
imin = min(irow,icol);

% INPUT
x10 = input('initial value of Log Map 1 = ');

x20 = input('initial value of Log Map 2 = ');

a = input('the first independent parameter = ');

b = input('the second independent parameter = ');

c = input('the third independent parameter = ');

n = input('the iteration number of the Arnold map = ');

wrow = input('the number of rows of watermark = ');

% EXTRACT

% STEP 1. DETERMINE EXTRACTION LOCATION

I_test = I_gray;

A=[a b;c (1+b*c)/a];

for i=1:wrow
    for j=1:wrow
```

```
        p0 = [i;j];
        it = 1;
        while it <= n
        q = mod(A*p0,imin);
        for t=1:2
            if q(t)==0
                q(t)=imin;
            end
        end
        p0=q;
        it=it+1;
        end
        pv = I_test(q(1),q(2));
        binpv = de2bi(pv,8);
        L2 = logmapf(x20,4,n);
        if L2(n) > 0.75
            k=4;
        else if L2(n) > 0.5
                k=4;
        else if L2(n) >0.25
                k=5;
            else
                k=5;
            end
            end
        end
        bin_ext(i,j)=binpv(k);
        x20=L2(n);
   end
end

% generate chaotic map
L1=logmapf(x10,4,wrow^2); m=mean(L1);

% convert to binary
for i=1:wrow^2
    if L1(i) >= m
        L1_bin(i) = 1;
    else
        L1_bin(i) = 0;
    end
end
%reshape d into matrix 64 x 64
L1_binmat = reshape(L1_bin,wrow,wrow);

W_ext = bitxor(double(bin_ext),double(L1_binmat));
```

```
% display test image
subplot(1,2,1), imshow(I), title('test image');
% display extracted watermark
subplot(1,2,2), imshow(W_ext), title('extracted watermark');

% save extracted watermark into files
name = regexp(fname1,'\.','split'); wat_ext_name =
strcat(name(1),'ex.',name(2)); imwrite(W_ext,char(wat_ext_name));
```

C.3 Watermark Insertion of the Second Scheme

```
% winsert2.m
% based on chaotic map, Arnold map and sub image
% original image size = 256x256
% watermark binary image size = 64 x 64
% ------------------------------------------------

clear
clc

% select host image
[fname1,pthname1] = uigetfile('*.tiff;*.png;*.jpg','Select the host
image');
% select watermark image
[fname2,pthname2] = uigetfile('*.tiff;*.png;*.jpg','Select the
watermark image');

% read host image
I_ori = imread([pthname1,fname1]);
% gray scale host image
I_gray =  I_ori(:,:,1); [irow,icol]=size(I_gray);

% read watermark image
W = imread([pthname2,fname2]);
% convert watermark image into binary image
W_bw = im2bw(W);

[wrow,wcol]=size(W_bw);

% INPUT
x10 = input('initial value of Log Map 1 = ');

x20 = input('initial value of Log Map 2 = ');

x30 = input('initial value of Log Map 3 = ');
```

```
a = input('the first independent parameter = ');

b = input('the second independent parameter = ');

c = input('the third independent parameter = ');

n = input('the iteration number of the Arnold map = ');

% STEP 1 ENCRYPT BINARY WATERMARK

% generate chaotic map

L1=logmapf(x10,4,wrow*wcol); m = mean(L1);
% convert to binary
for i=1:wrow*wcol
    if L1(i) >= m
        L1_bin(i) = 1;
    else
        L1_bin(i) = 0;
    end
end
%reshape L1_bin into matrix 64 x 64
L1_binmat = reshape(L1_bin,wrow,wcol);

% encrypt W_bw by xoring it with L1_binmat
W_enc = bitxor(W_bw,L1_binmat);

% STEP 2 CONSTRUCT SUB IMAGE FROM THE HOST IMAGE

% split I_gray into 8x8 disjoint blocks
I_block = block(I_gray,8,8);

% FORM SUB IMAGE

% constructing logistic map
L2 = logmapf(x20,4,1024);
[sorted index]=sort(L2,'ascend');

% reshape I_block into vector
I_vec = reshape(I_block,1,1024);

% create a sub image with 1/4 blocks less than I_block
for i = 1:256
    I_sub_vec(i) = I_vec(index(i));
end
```

```
% reshape I_sub_vec into matrix 16 x 16
I_sub_cell = reshape(I_sub_vec,16,16);
% convert into SUB IMAGE matrix
I_sub = cell2mat(I_sub_cell); % I_sub is an 128 x 128 matrix

% STEP 2 DETERMINE INSERTION PLACE

I_submarked = I_sub;

A=[a b; c (1+b*c)/a];

for i=1:wrow
    for j=1:wcol
        p0=[i;j];
        it = 1;
        while it <= n
        q = mod(A*p0,128);
        for t=1:2
            if q(t)==0
                q(t)=128;
            end
        end
        p0=q;
        it=it+1;
        end
        pv = I_submarked(q(1),q(2));
        binpv = de2bi(pv,8);
        L3 = logmapf(x30,4,n);
        if L3(n) > 0.75
            k=3;
        else if L3(n) > 0.5
                k=4;
        else if L3(n) >0.25
                k=5;
            else
                k=6;
            end
            end
        end
        newbinpv = binpv;
        newbinpv(k) = W_enc(i,j);
        newpv = bi2de(newbinpv);
        I_submarked(q(1),q(2))= newpv;
        x30=L3(n);
   end
end
```

```
% PLACE BACK the watermarked sub image into original image

% divide watermarked sub image into block
I_submarked_block = block(I_submarked,8,8);

% reshape into vector
I_submarked_block_vec = reshape(I_submarked_block,1,256);

% place the sub image 256 blocks into host image 1024 blocks
I_markedvec = I_vec;

for i=1:256
      I_markedvec(index(i)) = I_submarked_block_vec(i);
end I_marked_block = reshape(I_markedvec,32,32);

I_marked_gray = cell2mat(I_marked_block);

I_marked = I_ori;
I_marked(:,:,1) = I_marked_gray;

% save watermarked image into files
name = regexp(fname1,'\.','split'); wat_img_name =
strcat(name(1),'marked.',name(2));
imwrite(I_marked,char(wat_img_name));

% display original image
subplot(1,2,1), imshow(I_ori), title('original host image');
% display watermarked image
subplot(1,2,2), imshow(I_marked), title('watermarked image');

ssim = ssim_index(I_gray,I_marked_gray)
```

C.4 Watermark Extraction of the Second Scheme

```
% wextract2.m
% based on chaotic map, Arnold map and sub image
% original image size = 256x256
% watermark binary image size = 64 x 64
% ------------------------------------------------

clear
clc
% select test image
[fname1,pthname1] = uigetfile('*.tiff;*.png;*.jpg','Select the test
image');
```

```
% read the test image
I = imread([pthname1,fname1]);
% extract the gray scale part of the test image
I_size = imresize(I,[256 256]);
I_gray = I_size(:,:,1);
[irow,icol]=size(I_gray);
imin = min(irow,icol);

% INPUT
x10 = input('initial value of Log Map 1 = ');

x20 = input('initial value of Log Map 2 = ');

x30 = input('initial value of Log Map 3 = ');

a = input('the first independent parameter = ');

b = input('the second independent parameter = ');

c = input('the third independent parameter = ');

n = input('the iteration number of the Arnold map = ');

wrow = input('the number of rows of watermark = ');

% STEP 1 CONSTRUCT SUB IMAGE FROM THE TEST IMAGE

% split I_gray into 8x8 disjoint blocks
I_block = block(I_gray,8,8);

% FORM SUB IMAGE

% constructing logistic map
L2 = logmapf(x20,4,1024);
[sorted index]=sort(L2,'ascend');

% reshape I_block into vector
I_vec = reshape(I_block,1,1024);

% create a sub image with 1/4 blocks less than I_block
for i = 1:256
    I_sub_vec(i) = I_vec(index(i));
end
% reshape I_sub_vec into matrix 16 x 16
I_sub_cell = reshape(I_sub_vec,16,16);
```

```
% convert into SUB IMAGE matrix
I_sub = cell2mat(I_sub_cell); % I_sub is an 128 x 128 matrix

% STEP 2 DETERMINE Extraction PLACE

I_submarked = I_sub;

A=[a b;c (1+b*c)/a];

for i=1:wrow
    for j=1:wrow
        p0=[i;j];
        it = 1;
        while it <= n
        q = mod(A*p0,128);
        for t=1:2
            if q(t)==0
                q(t)=128;
            end
        end
        p0=q;
        it=it+1;
        end
        pv = I_submarked(q(1),q(2));
        binpv = de2bi(pv,8);
        L3 = logmapf(x30,4,n);
        if L3(n) > 0.75
            k=3;
        else if L3(n) > 0.5
                k=4;
        else if L3(n) >0.25
                k=5;
            else
                k=6;
            end
            end
        end
        binpv_ext(i,j) = binpv(k);
        x30=L3(n);
   end
end

% DECRYPT THE EXTRACTED MARK
% generate chaotic map
L1=logmapf(x10,4,wrow^2); m=mean(L1);
% convert to binary
```

```
for i=1:wrow^2
    if L1(i) >= m
        L1_bin(i) = 1;
    else
        L1_bin(i) = 0;
    end
end
%reshape d into matrix 64 x 64
L1_binmat = reshape(L1_bin,wrow,wrow);

W_ext = bitxor(double(binpv_ext),double(L1_binmat));

% display test image
subplot(1,2,1), imshow(I), title('test image');
% display extracted watermark
subplot(1,2,2), imshow(W_ext), title('extracted watermark');

% save extracted watermark into files
name = regexp(fname1,'\.','split');

wat_ext_name = strcat(name(1),'ex.',name(2));
imwrite(W_ext,char(wat_ext_name));
```

C.5 Watermark Insertion of the Third Scheme

```
% winsert3.m
% based on chaotic map, Arnold map and sub image
% in frequency domain
% original image size = 256x256
% watermark binary image size = 128 x 128
% ------------------------------------------------

clear
clc

% select host image
[fname1,pthname1] = uigetfile('*.tiff;*.png;*.jpg;*.jpeg','Select
the host image');
% select watermark image
[fname2,pthname2] = uigetfile('*.tiff;*.png;*.jpg;*.jpeg','Select
the watermark image');

% read host image
I_gray = imread([pthname1,fname1]);
[irow,icol]=size(I_gray);
```

```
% read watermark image
W = imread([pthname2,fname2]);
% convert watermark image into binary image
W_bw = im2bw(W);

[wrow,wcol]=size(W_bw);

% INPUT
x10 = input('initial value of Log Map 1 = ');

x20 = input('initial value of Log Map 2 = ');

a = input('the first independent parameter = ');

b = input('the second independent parameter = ');

c = input('the third independent parameter = ');

n = input('the iteration number of the Arnold map = ');

% STEP 1 ENCRYPT BINARY WATERMARK

% generate chaotic map
L1=logmapf(x10,4,wrow*wcol); m=mean(L1);
% convert to binary
for i=1:wrow*wcol
    if L1(i) >= m
        L1_bin(i) = 1;
    else
        L1_bin(i) = 0;
    end
end
%reshape L1_bin into matrix 128 x 128
L1_binmat = reshape(L1_bin,wrow,wcol);

% encrypt W_bw by xoring it with L1_binmat
W_enc = bitxor(W_bw,L1_binmat);

% STEP 2 CONSTRUCT SUB IMAGE FROM THE HOST IMAGE

% split I_gray into 8x8 disjoint blocks
I_block = block(I_gray,8,8);

% FORM SUB IMAGE

% constructing logistic map
```

```
L2 = logmapf(x20,4,1024);
% sort L2
[sorted index]=sort(L2,'ascend');

% reshape I_block into vector
I_vec = reshape(I_block,1,1024);

% create a sub image with 1/4 blocks less than I_block
for i = 1:256
    I_sub_vec(i) = I_vec(index(i));
end
% reshape I_sub_vec into matrix 16 x 16
I_sub_cell = reshape(I_sub_vec,16,16);
% convert into SUB IMAGE matrix
I_sub = cell2mat(I_sub_cell); % I_sub is an 128 x 128 matrix

% FORM SPREAD SPECTRUM WATERMARK

% convert encrypted watermark into bipolar
for i=1:wrow
    for j=1:wcol
        if W_enc(i,j)== 1
            W_enc_bip(i,j)=1;
        else
            W_enc_bip(i,j)=-1;
        end
    end
end

% transform I_sub into its spectrum
I_sub_dct = dct2(I_sub);

% construct spectrum watermark
for i=1:wrow
    for j=1:wcol
        t(i,j)=neighborless(I_sub_dct,i,j);
        if or(and(t(i,j)>=2,W_enc_bip(i,j)==1),
        and(t(i,j)<2,W_enc_bip(i,j)==-1))
            W_spec(i,j) = 1;
        else
            W_spec(i,j) = -1;
        end
    end
end

% STEP 2 DETERMINE INSERTION PLACE
```

```
A=[a b;c (1+b*c)/a];

for i=1:wrow
   for j=1:wcol
        p0=[i;j];
        it = 1;
        while it <= n
        q = mod(A*p0,128);
        %q = mod(A*p0,64);
        for t=1:2
            if q(t)==0
                q(t)=128;
                %q(t)=64;
            end
        end
        p0=q;
        it=it+1;
        end
        % insertion spectrum watermark into sub image spectrum
        I_sub_dct_wat = I_sub_dct;
        I_sub_dct_wat(q(1),q(2))=I_sub_dct(q(1),q(2))
        +0.01*W_spec(i,j)*abs(I_sub_dct(q(1),q(2)));
       end
 end

% transform back DCT coeficient into watermarked sub image
I_sub_wat = uint8(idct2(I_sub_dct_wat));

% PLACE BACK the watermarked sub image into original image

% divide watermarked sub image into block
I_sub_wat_block = block(I_sub_wat,8,8);

% reshape into vector
I_sub_wat_block_vec = reshape(I_sub_wat_block,1,256);

% place the sub image 64 blocks into host image 1024 blocks
I_wat_vec = I_vec;

for i=1:256
      I_wat_vec(index(i)) = I_sub_wat_block_vec(i);
end

I_wat_block = reshape(I_wat_vec,32,32);
```

```
I_wat_gray = cell2mat(I_wat_block);

% save watermarked image into files
name = regexp(fname1,'\.','split');

wat_img_name = strcat(name(1),'marked.',name(2));

wat_spec_name = strcat(name(1),'spec.','mtr');
imwrite(I_wat_gray,char(wat_img_name));
dlmwrite(char(wat_spec_name),W_spec);

% display original image
figure, subplot(1,2,1), imshow(I_gray), title('original host
image');
% display watermarked image
subplot(1,2,2), imshow(I_wat_gray), title('watermarked image');

mssim = ssim_index(I_gray,I_wat_gray)
```

C.6 Watermark Extraction of the Third Scheme

```
% wextract3.m
% based on chaotic map, Arnold map and sub image
% in the frequency domain
% original image size = 256x256
% watermark binary image size = 128 x 128
% ------------------------------------------------

clear
clc
% select test image
[fname1,pthname1] = uigetfile('*.tiff;*.png;*.jpg;*.jpeg','Select
the test image');
% select spectrum watermark
[fname2,pthname2] = uigetfile('*.mtr','Select the spectrum
watermark');

% read the test image
I_test = imread([pthname1,fname1]);

I_size = imresize(I_test,[256 256]);
I_gray = I_size;
[irow,icol]=size(I_gray);
imin = min(irow,icol);

% INPUT
```

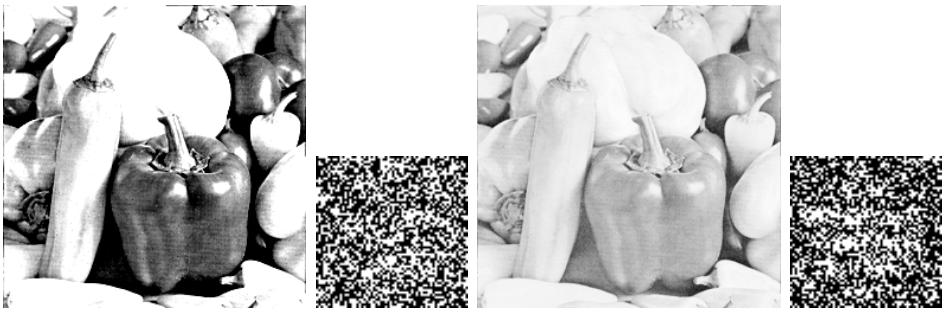
```
x10 = input('initial value of Log Map 1 = ');

x20 = input('initial value of Log Map 2 = ');

a = input('the first independent parameter = ');

b = input('the second independent parameter = ');

c = input('the third independent parameter = ');

n = input('the iteration number of the Arnold map = ');

wrow = input('the number of rows of watermark = ');

W_spec = dlmread([pthname2,fname2]);

% STEP 1 CONSTRUCT SUB IMAGE FROM THE TEST IMAGE

% split I_test into 8x8 disjoint blocks
I_block = block(I_gray,8,8);

% FORM SUB IMAGE

% constructing logistic map
L2 = logmapf(x20,4,1024);
[sorted index]=sort(L2,'ascend');

% reshape I_block into vector
I_vec = reshape(I_block,1,1024);

% create a sub image with 1/4 blocks less than I_block
for i = 1:256
    I_sub_vec(i) = I_vec(index(i));
end
% reshape I_sub_vec into matrix 16 x 16
I_sub_cell = reshape(I_sub_vec,16,16);
% convert into SUB IMAGE matrix
I_sub = cell2mat(I_sub_cell); % I_sub is an 128 x 128 matrix

% transform I_sub into its spectrum
I_sub_dct = dct2(I_sub);

% construct spectrum watermark
for i=1:wrow
    for j=1:wrow
        t(i,j)=neighborless(I_sub_dct,i,j);
```

```
        if or(and(t(i,j)>=2,W_spec(i,j)==1),
        and(t(i,j)<2,W_spec(i,j)==-1))
            W_enc_bip(i,j) = 1;
        else
            W_enc_bip(i,j) = -1;
        end
    end
end

for i=1:wrow
    for j=1:wrow
        if W_enc_bip(i,j) == -1
            W_enc_bin(i,j) = 0;
        else
            W_enc_bin(i,j) = 1;
        end
    end
end

% generate chaotic map
L1=logmapf(x10,4,wrow^2); m=mean(L1);
% convert to binary
for i=1:wrow^2
    if L1(i) >= m
        L1_bin(i) = 1;
    else
        L1_bin(i) = 0;
    end
end
%reshape d into matrix 64 x 64
L1_binmat = reshape(L1_bin,wrow,wrow);

W_ext = bitxor(double(W_enc_bin),double(L1_binmat));

% display test image
subplot(1,2,1), imshow(I_test), title('test image');
% display extracted watermark
subplot(1,2,2), imshow(W_ext), title('extracted watermark');

% save extracted watermark into files
name = regexp(fname1,'\.','split'); wat_ext_name =
strcat(name(1),'ex.',name(2)); imwrite(W_ext,char(wat_ext_name));
```

Appendix D

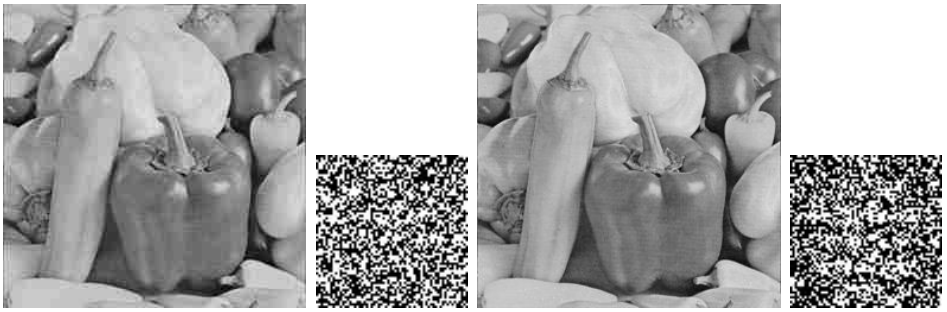
Examples of the Watermarking Scheme's Visual Results

D.1 Examples of the First Scheme's Visual Results



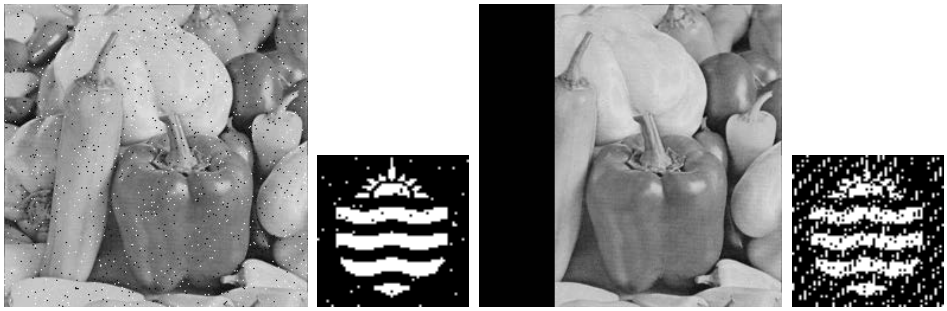
(a) (b) (c) (d)

Figure D.1: The 1st scheme results: D.1a: contrasted image; D.1c: brightened image; D.1b & D.1d: the corresponding extracted watermarks



(a) (b) (c) (d)

Figure D.2: The 1st scheme results: D.2a: compressed image with Quality 20 %; D.2c: compressed with Quality 80 %; D.2b & D.2d: the corresponding extracted watermarks

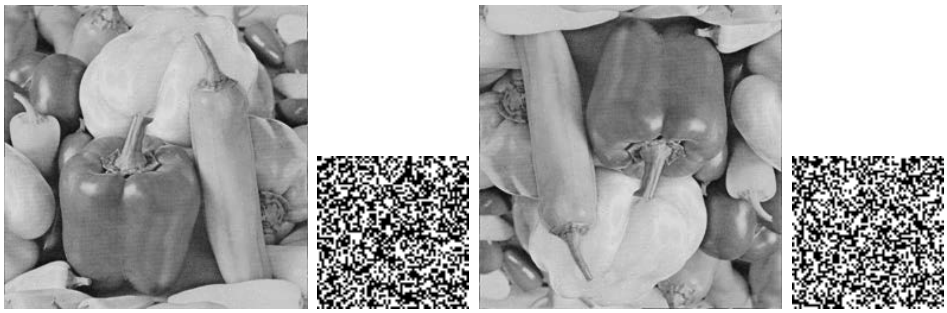


(a) (b) (c) (d)

Figure D.3: The 1st scheme results: D.3a: noisy image; D.3c: 25 % cropped image; D.3b & D.3d: the corresponding extracted watermarks

(a) (b) (c) (d)

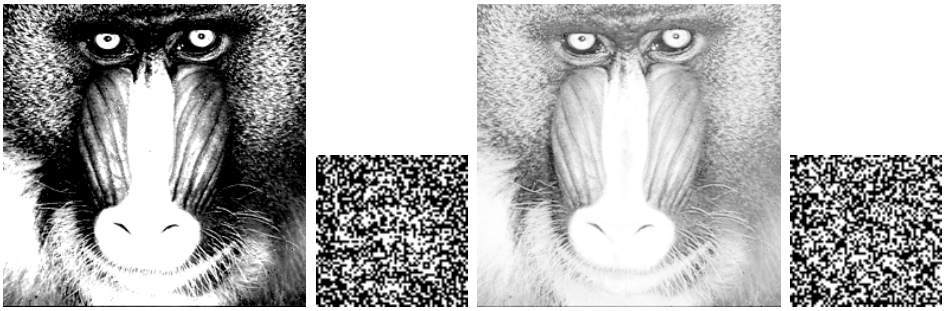
Figure D.4: The 1st scheme results: D.4a: 2 degree rotated image; D.4c: 10 degree rotated image; D.4b & D.4d: the corresponding extracted watermarks



(a) (b) (c) (d)

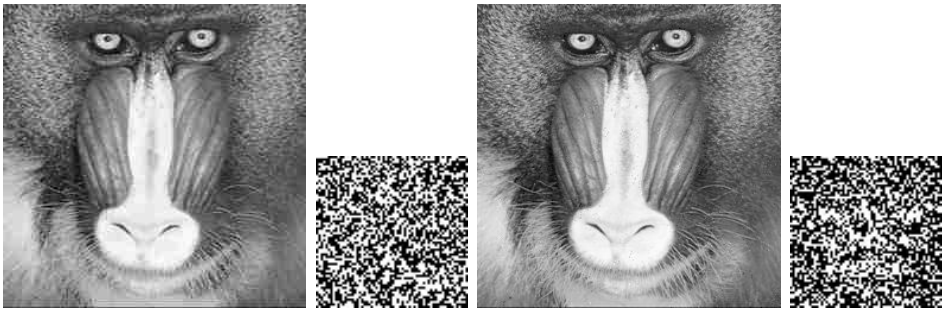
Figure D.5: The 1st scheme results: D.5a: horizontally flipped image; D.5c: vertically flipped image; D.5b & D.5d: the corresponding extracted watermarks

D.2 Examples of the Second Scheme's Visual Results



(a) (b) (c) (d)

Figure D.6: The 2nd scheme results: D.6a: contrasted image; D.6c: brightened image; D.6b & D.6d: the corresponding extracted watermarks



(a) (b) (c) (d)

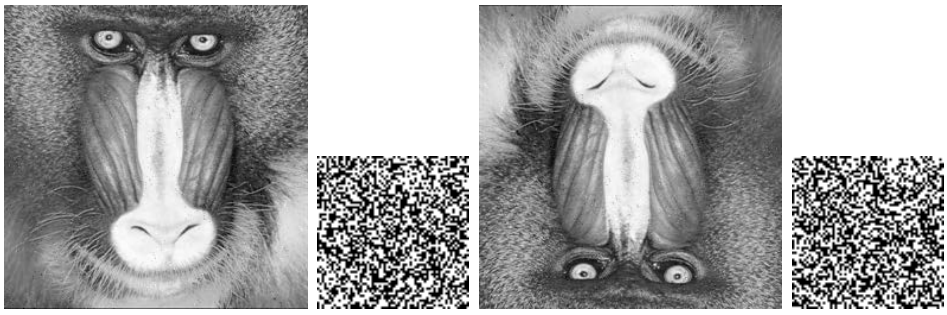
Figure D.7: The 2nd scheme results: D.7a: compressed image with Quality 20 %; D.7c: compressed with Quality 80 %; D.7b & D.7d: the corresponding extracted watermarks

(a) (b) (c) (d)

Figure D.8: The 2nd scheme results: D.8a: noisy image; D.8c: 25 % cropped image; D.8b & D.8d: the corresponding extracted watermarks

(a) (b) (c) (d)

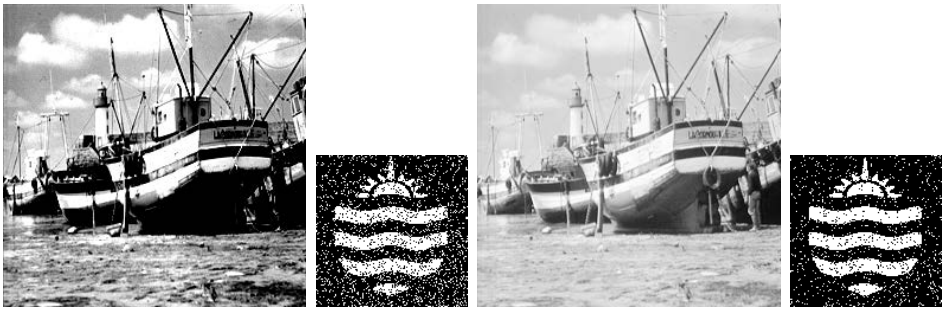
Figure D.9: The 2nd scheme results: D.9a: 2 degree rotated image; D.9c: 10 degree rotated image; D.9b & D.9d: the corresponding extracted watermarks



(a) (b) (c) (d)

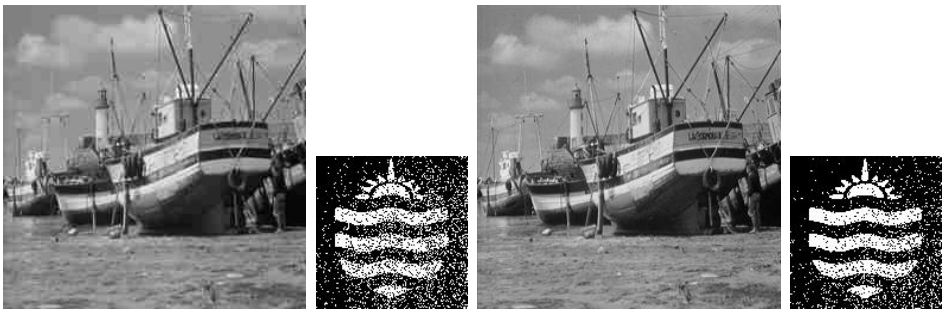
Figure D.10: The 2nd scheme results: D.10a: horizontally flipped image; D.10c: vertically flipped image; D.10b & D.10d: the corresponding extracted watermarks

D.3 Examples of the Third Scheme's Visual Results



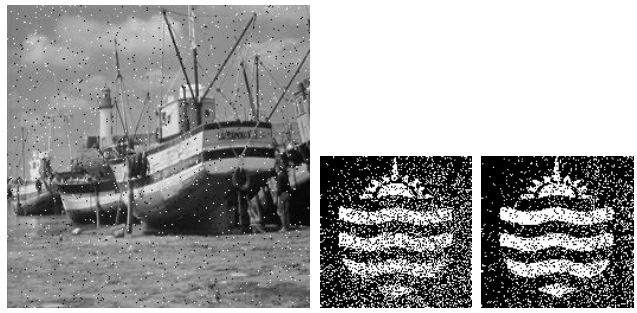
(a) (b) (c) (d)

Figure D.11: The 3rd scheme results: D.11a: contrasted image, D.11c: brightened image; D.11b & D.11d: the corresponding extracted watermarks

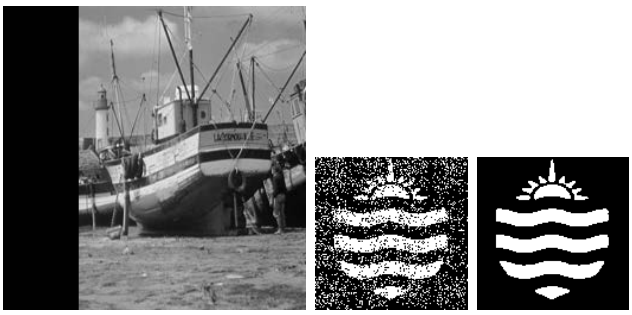


(a) (b) (c) (d)

Figure D.12: The 3rd scheme results: D.12a: compressed image with Quality 20 %; D.12c: compressed with Quality 80 %; D.12b & D.12d: the corresponding extracted watermarks

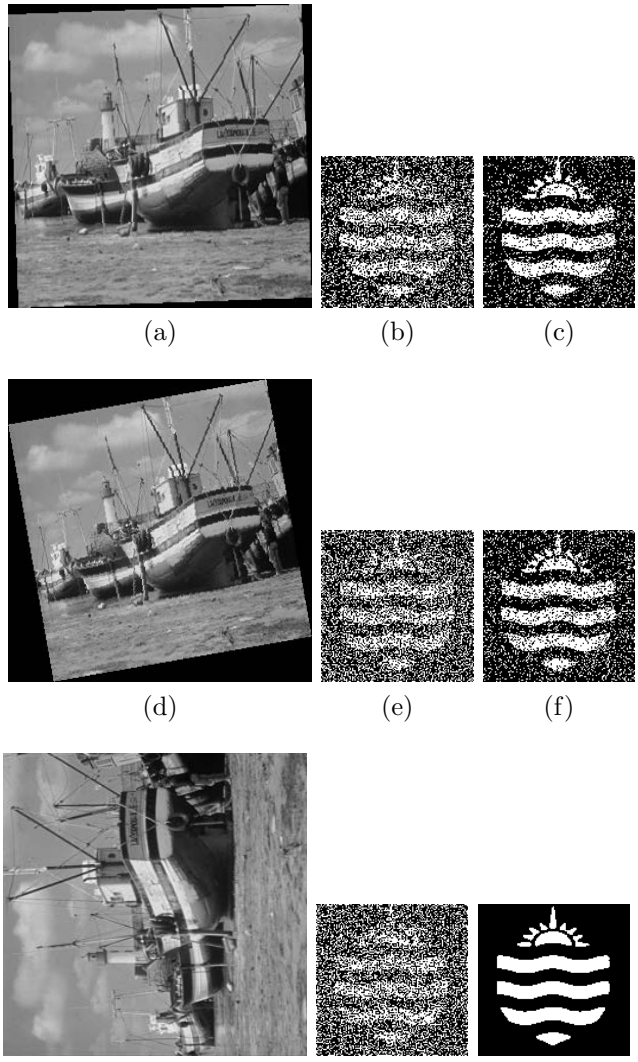


(a) (b) (c)



(d) (e) (f)

Figure D.13: The 3rd scheme simulation results: D.13a: noisy image; D.13d: 25 % cropped image; D.13b & D.13e: the corresponding direct extracted watermarks; D.13c: & D.13f: watermarks extracted from the corresponding recovered images

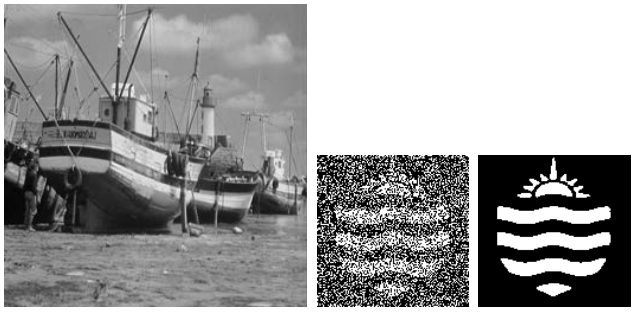


(a) (b) (c)

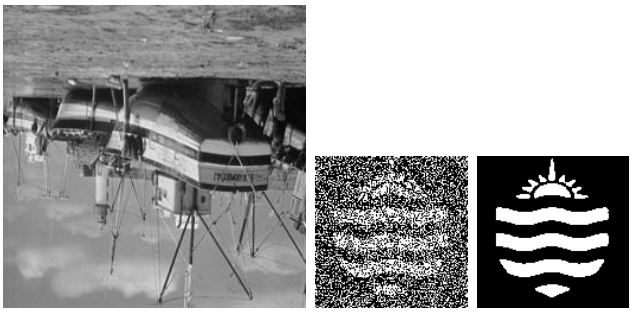
(d) (e) (f)

(g) (h) (i)

Figure D.14: The simulation results: D.14a: 2 degree rotated image; D.14d 10 degree rotated image; D.14b; D.14g: 90 degree rotated image; D.14b, D.14e & D.14h: the corresponding direct extracted watermarks; D.14c, D.14f & D.14i: watermarks extracted from the corresponding recovered images



(a) (b) (c)



(d) (e) (f)

Figure D.15: The simulation results: D.15a: horizontally flipped image; D.15d: vertically flipped image; D.15b & D.15e: the corresponding direct extracted watermarks; D.15c & D.15f: watermarks extracted from the corresponding recovered image

Bibliography

[1] AACS, "Advanced access content system," 2012. [Online]. Available: http://www.aacsla.com/home 18, 143

[2] Adobe, "Adobe digital editions," 2009. [Online]. Available: http://www.adobe.com/products/digitaleditions/ 20

[3] ADRI, "Statement on the application of digital rights management technology to public records," Council of Australasian Archives and Records Authorities, Tech. Rep. 1.0, August 2008. [Online]. Available: http://www.adri.gov.au/ 3

[4] I. Ahn and I. Shin, "On the optimal level of protection in DRM," *Information Economics and Policy*, vol. 22, pp. 341–353, 2010. 39

[5] M. Al-Fayoumi and S. Aboud, "Blind decryption and privacy protection," *American Journal of Applied Science*, vol. 2, no. 4, pp. 873–876, 2005. 52

[6] B. S. Alliance, "Fifth annual BSA and IDC global software: Piracy study," Business Software Alliance, Tech. Rep., 2007. [Online]. Available: http://global.bsa.org/idcglobalstudy2007//studies/2007_global_piracy_study.pdf xxvii, 2

[7] ——, "Shadow market: 2011 BSA global software piracy study," Business Software Alliance, Tech. Rep., May 2012. [Online]. Available: http://portal.bsa.org/globalpiracy2011 xxvii, 2

[8] Amazon, "Amazon.com:kindle ebooks." [Online]. Available: www.amazon.com 21

[9] C. Antons, *Copyright Law, Digital Content and the Internet in the Asia-Pacific*. Sydney University Press, March 2008, ch. Copyright Law Reform and the Information Society in Indonesia, pp. 235–256. 3

[10] Apple, "iTunes home page," 2012. [Online]. Available: http://www.apple.com/itunes/ 19

[11] A. Arnab and A. Hutchison, "Digital rights management - an overview of current challenges and solutions," in *Infosec South Africa 2004 Conference*, 2004. 40

[12] J. Ayubi, S. Mohanna, F. Mohanna, and M. Rezaei, "A chaos based blind digital image watermarking in the wavelet transform domain," *IJCSI - International Journal of Computer Science Issues*, vol. 8, no. 4/2, pp. 192–199, July 2011. 110, 131

[13] S. E. Baba, L. Z. Krikor, T. Arif, and Z. Shaaban, "Watermarking of digital images in freqeuncy domain," *International Journal of Automation and Computing*, vol. 7, no. 1, pp. 17–22, February 2010. 103

[14] B. Barak, O. Goldreich, R. Impagliazzo, S. Rudich, A. Sahai, S. Vadhan, and K. Yang, "On the (im)possibility of obfuscating program," in *Advance in Cryptology - CRYPTO 2001: 21st Annual International Cryptology Conference*, J. Kilian, Ed., vol. LNCS 2139. Berlin: Springer, August 2001, pp. 1–18. 78, 79

[15] E. Becker, W. Buhse, D. Gunnewig, and N. Rump, "DRM as an interlocking challenge for different scientific disciplines: Introduction," in *Digital Rights Management: Technological, Economic, Legal and Political Aspects*, E. Becker, W. Buhse, D. Gunnewig, and N. Rump, Eds., vol. LNCS 2770. Berlin Heidelberg: Springer-Verlag, 2003, pp. 1–2. 12

[16] O. Berkman, M. Parnas, and J. Sgall, "Efficient dynamic traitor tracing," in *11th Annual ACM-SIAM Symposium on Discrete Algorithms*, vol. SODA 2000, 2000, pp. 586–595. 141

[17] O. Billet and H. Gilbert, "A traceable block cipher," in *Advances in Cryptology - ASIACRYPT 2003*, C.-S. Laih, Ed., vol. LNCS 2894. Springer, 2003, pp. 331–346. 81

[18] O. Billet, H. Gilbert, and C. Wch-Chatbi, "Cryptanalysis of a white box AES implementation," in *SAC 2004*, H. Handschuh and A. Hasan, Eds., vol. LNCS 3357, 2005, pp. 227–240. 81

[19] J. A. Bloom, I. J. Cox, T. Kalker, J.-P. M. Linnartz, M. L. Miller, and C. B. S. Traw, "Copy protection for DVD video," *Proceedings of the IEEE*, vol. 87, no. 7, pp. 1267–1276, July 1999. 16

[20] C. Blundo, P. D'Arco, A. D. Santis, and D. R. Stinson, "On unconditionally secure distributed oblivious transfer," *Journal of Cryptology*, vol. 20, no. 3, pp. 323–373, 2007. 98

[21] D. Boneh and M. Franklin, "An efficient public key traitor tracing scheme," in *CRYPTO'99*, vol. LNCS 773. Springer-Verlag, 1999, pp. 338–353. 136, 139

[22] D. Boneh and J. Shaw, "Collusion-secure fingerprinting for digital data," *IEEE Transactions on Information Theory*, vol. 44, no. 5, pp. 1897–1905, September 1998. 140

[23] B. E. Boyden, "Is DRM working? how could we tell?" in *Proceeding of the 11th Annial ACM Workshop on Digital Rights Management*. ACM Digital Library, 2011, pp. 1–2. 35

[24] J. T. Brassil, S. Low, and N. F. Maxemchuk, "Copyright protection for the electronic distribution of text document," *Proceedings of the IEEE*, vol. 87, no. 7, pp. 1181–1196, July 1999. 101

[25] J. Bringer, H. Chabanne, and E. Dottax, "Perturbing and protecting a traceable block cipher," in *Proceedings of the 10th Communications and Multimedia Security (CMS) 2006*, vol. LNCS 4237. Springer-Verlag, 2006, pp. 109–119. 81

[26] B. Brown, "Sony BMG rootkit scandal: 5 years later," November 2010. [Online]. Available: http://www.networkworld.com/news/2010/110110-sonybmg-rootkit-fsecure-drm.html 20

[27] I. Brown, "Implementing the EU copyright directive," The Foundation for Information Policy Research (FIPR), Tech. Rep., 2001. [Online]. Available: http://www.fipr.org/copyright/guide/eucd-guide.pdf 3

[28] M. Campidoglio, F. Frattolillo, and F. Landolfi, "The copyright protection problem: Challenges and suggestions," in *Fourth International Conference on Internet and Web Applications and Services, 2009. ICIW'09*, Venice/Mestre, 2009, pp. 522–526. 35, 38, 39

[29] R. Canetti and R. R. Dakdouk, "Obfuscating point functions with multibit output," in *EUROCRYPT 2008*, N. Smart, Ed., vol. LNCS 4965. Springer, 2008, pp. 489–508. 79

[30] D. Castro, "Targeting websites dedicated to stealing American intellectual property," February 2011. [Online]. Available: www.itif.org/files/2011-coica-testimony.pdf 36

[31] D. Chaum, "Blind signatures for untreacable payments," in *Advances in Cryptology - CRYPTO'82*, Plenum, NY, 1982, pp. 199–203. 45, 47

[32] D. Chaum, A. Fiat, and M. Naor, "Untraceable electronic cash," in *Advance in Cryptology - CRYPTO'88*, S. Goldwasser, Ed., vol. LNCS 403. Berlin Heidelberg: Springer-Verlag, 1988, pp. 319–327. 48, 49, 50, 77

[33] Z. Chen, *Java Card Technology for Smart Cards: Architecture and Programmer's Guide*. USA: Addison-Wesley, 2000. 73

[34] Y. Cheng, Q. Liu, X. Zhu, C. Zhao, and S. Li, "Research on digital content protection technology for video and audio based on ffmpeg," *International Journal of Advancements in Computing Technology*, vol. 3, no. 8, pp. 9–17, September 2011. 37

[35] B. Chor, A. Fiat, and M. Naor, "Tracing traitors," in *Proceedings Advance in Cryptology - CRYPTO'94*, vol. LNCS 839 (1994). Springer-Verlag, 1994, pp. 257–270. 138, 139

[36] B. Chor, A. Fiat, M. Naor, and B. Pinkas, "Tracing traitors," *IEEE Transaction on Information Theory*, vol. 46, no. 3, pp. 893–910, May 2000. 136, 138

[37] S. Chow, P. Eisen, H. Johnson, and P. C. van Oorschot, "A white-box DES implementation for DRM applications," in *DRM 2002*, J. Feigenbaum, Ed., vol. LNCS 2696. Berlin Heidelberg: Springer-Verlag, 2003, pp. 1–15. 80

[38] ——, "White-box cryptography and an AES implementation," in *SAC 2002*, K. Nyberg and H. Heys, Eds., vol. LNCS 2595. Berlin Heidelberg: Springer-Verlag, 2003, pp. 250–270. xxvii, 80, 81

[39] C.-K. Chu and W.-G. Tzeng, "Efficient k-out-of-n oblivious transfer schemes with adaptive and non-adaptive queries," in *PKC 2005*, S. Vaudenay, Ed., vol. LNCS 3386, 2005, pp. 172–183. 83

[40] J. E. Cohen, "Drm and privacy," *Berkeley Technology Law Journal*, vol. 18, pp. 575–617, 2003. 31

[41] M. D. Community, "Marlin architecture overview," 2007. [Online]. Available: http://www.marlin-community.com/files/Marlin10531.pdf xxi, 27, 28

[42] C. Conrado, M. Petkovic, and W. Jonker, "Privacy-preserving digital rights management," in *Secure Data Management*, ser. LNCS, vol. 3178. Springer, 2004, pp. 83–99. 33, 63

[43] C. L. Corniaux and H. Ghodosi, "An information-theoretically secure threshold distributed oblivious transfer protocol," in *Information Security and Cryptology - ICISC 2012*, vol. LNCS 7839. Springer Berlin Heidelberg, 2013, pp. 184–201. 98

[44] ——, "A verifiable 1-out-of-n distributed oblivious transfer protocol," Cryptology ePrint Archive, Report 2013/063, 2013, http://eprint.iacr.org/. 98

[45] I. J. Cox, J. Kilian, T. Leighton, and T. Shamoon, "Secure spread spectrum watermarking," *IEEE Trans. on Image Processing*, vol. 6, no. 12, pp. 1673–1687, 1997. 107, 110, 131

[46] I. J. Cox and J.-P. M. Linnartz, "Some general methods for tampering with watermarks," *IEEE Journal on Selected Areas in Communications*, vol. 16, no. 4, pp. 587–593, May 1998. 103

[47] I. J. Cox, M. L. Miller, J. A. Bloom, J. Fridrich, and T. Kalker, *Digital Watermarking and Steganography*, 2nd ed., E. A. Fox, Ed. Burlington, USA: Morgan Kaufmann Publisher, 2008. 102

[48] Z. Dawei, C. Guanrong, and L. Wenbo, "A chaos-based robust wavelet-domain watermarking algorithm," *Chaos, Solitons and Fractals*, vol. 22, pp. 47–54, 2004. 109, 111

[49] E. Diehl, "A four-layer model for security of digital rights management," in *Proceedings of 18th ACM Workshop on Digital Rights Management*, Alexandria, Virginia, USA, 2008. xxi, 14, 15

[50] W. Diffie and M. E. Hellman, "New direction in cryptography," *IEEE Transactions on Information Theory*, vol. IT-22, no. 6, pp. 644–654, November 1976. 53

[51] M. Dorairangaswamy, "A novel invisible and blind watermarking scheme for copyright protection of digital images," *IJCSNS International Journal of Computer Science and Network Security*, vol. 9, no. 4, pp. 71–78, April 2009. 37

[52] J. Du, C.-S. Woo, and B. Pham, "Recovery of watermark using differential affine motion estimation," in *Proceedings Australian Information Security Workshop*, Newcastle, Australia, 2005. 104

[53] D. Eran, "How fairplay works: Apple's itunes DRM dilemma," 2007. [Online]. Available: http://www.roughlydrafted.com/RD/RDM.Tech.Q1.07/2A351C60-A4E5-4764-A083-FF8610E66A46.html 21

[54] S. Even, O. Goldriech, and A. Lempel, "A randomized protocol for signing contracts," *Communications of the ACM*, vol. 28, no. 6, pp. 637–647, June 1985. 83

[55] EZDRM, "Windows media DRM (WMDRM)," 2000. [Online]. Available: http://www.ezdrm.com/html/Microsoft/Windows_Media/WMDRM/about_WMDRM.asp 18

[56] N. Fazio, "On cryptographic techniques for digital rights management," Ph.D. dissertation, Departement of Computer Science, New York University, 2006. 136

[57] J. Feigenbaum, M. J. Freedman, T. Sander, and A. Shostack, "Privacy engineering for digital rights management systems," in *Security and Privacy in Digital Rights Management*, ser. Lecture Notes in Computer Science, vol. 2320. Springer Berlin Heidelberg, 2002, pp. 76–105. 31

[58] M. Feng and B. Zhu, "A DRM system protecting consumer privacy," in *Consumer Communications and Networking Conference*, vol. 5th IEEE CCNC 2008, IEEE. IEEE, 2008, pp. 1075–1079. 40

[59] G. Fernando, T. Jacobs, and V. Swaminathan, "Project DReaM: An architectural overview," Sun Microsystems, White Paper, 2005. [Online]. Available: http://www.docstoc.com/docs/425459/ xxi, 25, 26

[60] M. Fetscherin and M. Schmid, "Comparing the usage of digital rights management systems in the music, film and print industry," in *Proceedings of the 5th International Conference on Electronic Commerce*, vol. ICEC'03, 2003, pp. 316–325. 36

[61] A. Fiat and M. Naor, "Broadcast encryption," in *Proceeding of Crypto 93*, vol. LNCS 773. Springer-Verlag, 1993, pp. 480–491. 137

[62] A. Fiat and T. Tassa, "Dynamic traitor tracing," *Cryptology*, vol. 2001, no. 14, pp. 211–223, 2001. 137, 141

[63] P. Foris and D. Levicky, "Implementations of HVS models in digital image watermarking," *Radioengineering*, vol. 16, no. 1, pp. 45–50, April 2007. 105

[64] F. Gao, *Copyright Law, Digital Content and the Internet in the Asia-Pacific*. Sydney University Press, March 2008, ch. A Legal Framework for the Development of the Content Industry in the People's Republic of China, pp. 29–48. 3

[65] M. Ghebleh, A. Kanso, and H. S. Own, "A blind chaos-based watermarking technique," *Security and Communication Networks*, vol. 2013, 2013. 111, 130

[66] H. Ghodosi, "A general model for oblivious transfer," in *Proceedings of the Sixth International Workshop for Applied PKC*, vol. IWAP2007, Perth, Australia, December 2007, pp. 79–87. 83

[67] ——, "Analysis of an unconditionally secure distributed oblivious transfer," *Journal of Cryptology*, vol. 2013, no. 26, pp. 75–79, 2013. 98

[68] H. Ghodosi and I. Lee, "Unconditional security and privacy preserving oblivious transfer," in *Proceedings of the 7th International Conference on Computing and Convergence Technology*, vol. ICCCT2012, Seoul, Korea, December 2012, pp. 1042–1047. 83

[69] G. Goble, "DRM from 1998 to present: A brief history of copy protection," 2011. [Online]. Available: http://www.maximumpc.com 39

[70] O. Goldreich and R. Ostrovsky, "Software protection and simulation on oblivious RAMs," *Journal of the Association of Computing Machinery*, vol. 43, no. 3, pp. 431–473, 1996. 78

[71] S. Goldwasser, Y. T. Kalai, and G. N. Rothblum, "One-time programs," in *CRYPTO 2008*, D. Wagner, Ed., vol. LNCS 5157. International Association for Cryptologic Research 2008, 2008, pp. 39–56. 85, 86

[72] S. Goldwasser and G. N. Rothblum, "On best-possible obfuscation," in *TCC 2007*, S. P. Vadhan, Ed., vol. LNCS 4392. Amsterdam, The Netherland: Springer, 2007, pp. 194–213. 79

[73] R. Gooch, "Requirements for DRM systems," in *Digital Rights Management: Technological, Economic, Legal and Political Aspects*, E. Becker, W. Buhse, D. Gunnewig, and N. Rump, Eds., vol. LNCS 2770. Berlin Heidelberg: Springer-Verlag, 2003, pp. 16–25. 12

[74] K. Goyal, "Calibre: e-book management." [Online]. Available: calibre-ebook.com 21

[75] M. Green, "Secure blind decryption," in *Public Key Cryptography - PKC 2011*, D. Catalano, Ed., vol. LNCS 6571. Springer Berlin Heidelberg, 2011, pp. 265–282. 52

[76] P. Guarda and N. Zannone, "Towards the development of privacy-aware systems," *Information and Software Technology*, vol. 51, pp. 337–350, 2009. 39

[77] C. Guyeux and J. M. Bahi, "A new chaos-based watermarking algorithm," in *International Conference on Security and Cryptography*, vol. SECRYPT'10, 2010. 110, 131

[78] S. Haber, B. Horne, J. Pato, T. Sander, and R. E. Tarjan, "If piracy is the problem, is DRM the answer?" in *Digital Rights Management: Technological, Economic, Legal and Political Aspects*, E. Becker, W. Buhse, D. Gunnewig, and N. Rump, Eds., vol. LNCS 2770. Berlin Heidelberg: Springer-Verlag, 2003, pp. 224–233. 34

[79] S. Hada, "Zero-knowledge and code obfuscation," in *ASIACRYPT 2000*, T. Okamoto, Ed., vol. LNCS 1976. Berlin Heidelberg: Springer-Verlag, 2000, pp. 443–457. 78

[80] V. Halttunen, M. Makkonen, L. Frank, and P. Tyrvalnen, "Perspectives on digital content markets: A literature review of trends in technologies, business and consumer behaviour," *Communicating of IBIMA*, vol. 2010, pp. 1–14, 2010. 36

[81] J. He and H. Zhang, "Digital rights management model based on cryptography and digital watermarking," in *2008 International Conference on Computer Science and Software Engineering*, 2008, pp. 656–660. 34, 37

[82] N. Helberger, N. Dufft, S. van Gompel, K. Kerenyi, B. Krings, R. Lambers, C. Orwat, and U. Riehm, "Digital right management and consumer acceptability," Indicare, State-of-the-Art Report, 2004. 40

[83] D. Hofheinz, J. Malone-Lee, and M. Stam, "Obfuscation for cryptographic purposes," in *TCC 2007*, S. P. Vadhan, Ed., vol. LNCS 4392. Amsterdam, The Netherland: Springer, 2007, pp. 214–232. 79

[84] S. Hohenberger, G. N. Rothblum, A. Shelat, and V. Vaikuntanathan, "Securely obfuscating re-encryption," in *TCC 2007*, S. P. Vadhan, Ed., vol. LNCS 4392. Amsterdam, The Netherland: Springer, 2007, pp. 233–252. 80

[85] D. Huang and H. Yan, "Interword distance changes represented by sine wives for watermarking text images," *IEEE Trans. Circuits and Systems for Video Technology*, vol. 11, no. 12, pp. 1237–1245, December 2001. 101

[86] R. Iannella, "Digital rights management (DRM) architectures," *D-Lib Magazine*, vol. 7, no. 6, June 2001. xxi, 14

[87] IFPI, "IFPI digital music report 2011: Music at the touch of a buton," IFPI:representing the recording industry worldwide, Tech. Rep., 2011. [Online]. Available: www.ifpi.org/content/library/DMR2011.pdf 1, 2

[88] "Open digital rights language (ODRL)," IPR Systems Pty Ltd, August 2002. [Online]. Available: http://odrl.net/1.1/ODRL-11.pdf 24

[89] Z. Jalil, A. M. Mirza, and M. Sabir, "Content based zero-watermarking algorithm for authentication of text documents," *(IJCSIS) International Journal of Computer Science and Information Security*, vol. 7, no. 2, pp. 212–217, February 2010. 102

[90] C. Z. Jiang, *Copyright Law, Digital Content and the Internet in the Asia-Pacific*. Sydney University Press, March 2008, ch. The Judicial Protection of Copyright on the Internet in the People's Republic of China, pp. 15–25. 3

[91] H. Jin, J. Lotspiech, and S. Nusser, "Traitor tracing for prerecorded and recordable media," in *ACM DRM Workshop*, vol. DRM'04, 2004, pp. 83–90. 19, 145, 146, 157, 159

[92] H. Jin, J. Lotspiech, and N. Megiddo, "Efficient traitor tracing," IBM Research Division, IBM Research Report RJ10390 (A0610-018), 2006. 19, 137, 154

[93] ——, "Efficient coalition detection in traitor tracing," in *Proceeding of IFIP International Conference on Information Security*, 2008, pp. 365–379. 19, 144, 154, 155

[94] H. Jin, J. Lotspiech, M. Nelson, and N. Megiddo, "Adaptive traitor tracing for large anonymous attack," in *Proceedings of the 8th ACM Workshop on Digital Rights Management*, G. Heileman and M. Joye, Eds., vol. DRM'08. ACM, 2008, pp. 1–7. 19, 154

[95] S. Jobs, "Thoughts on music," February 2007. [Online]. Available: http://www.apple.com/kr/hotnews/thoughtsonmusic/ 19

[96] M. Joye, "On white-box cryptography," in *Security of Information and Networks*, A. Elci, S. Ors, and B. Prencel, Eds. Trafford Publishing, 2008, pp. 7–12. 71

[97] M. Joye and T. Lepoint, "Traitor tracing schemes for protected software implementations," in *Proceedings of the 11th Annual ACM Workshop on Digital Rights Management*, Y. Chen, S. Katzenbeisser, and A.-R. Sadeghi, Eds., vol. DRM'11. ACM, 2011, pp. 15–21. 136, 140, 145, 150, 157

[98] ——, "Traitor tracing for software-implemented decryption algorithms," Google Patent WO2013004691A1, January 2013. 157

[99] R. Kasprowski, "Perspective on DRM: Between digital rights management and digital restrictions management," *Bulletin of the American Society for Information Science*, vol. 36, no. 3, pp. 49–54, February 2010. 38

[100] R. Kirk, "2010 special 301 report," The Office of the United States Trade Representative, Tech. Rep., April 2010. 3

[101] T. Kohda and K. Aihira, "Chaos in discrete systems and diagnosis of experimental chaos," *IEICE Transactions*, vol. E 73, no. 6, pp. 772–783, 1990. 112

[102] A. Kumar, V. Tyagi, M. Dilshad, and K. Kumar, "A practical buyer-seller watermarking protocol based on discrete wavelet transform," *International Journal of Computer Applications*, vol. 21, no. 8, pp. 46–51, 2011. 37

[103] K. Kurosawa and Y. Desmedt, "Optimum traitor tracing and asymetric schemes," in *EUROCRYPT'98*, vol. LNCS 1403. Springer, 1998, pp. 145–157. 136, 139

[104] S. Kwok, C. Yang, and K. Tam, "Intellectual property protection for electronic commerce applications," *Journal of Electronic Commerce Research*, vol. 5, no. 1, pp. 1–13, 2004. 4

[105] J. Layton, "How digital rights management works," 2011. [Online]. Available: http://computer.howstuffworks.com/drm.htm 38, 39

[106] M. Lesk, M. R. Stytz, and R. L. Trope, "Digital rights management and individualized pricing," *IEEE Security and Privacy*, vol. 6, no. 3, pp. 76–79, 2008. 12

[107] Y. F. Lim, *Copyright Law, Digital Content and the Internet in the Asia-Pacific*. Sydney University Press, March 2008, ch. New Hope for Consumers of Digital Copyright Material in Hongkong, pp. 349–358. 3

[108] Q. Liu, R. Safavi-Naini, and N. P. Sheppard, "Digital rights management for content distribution," in *Proceedings of Australian Information Security Workshop on ACSW Frontiers '03*, vol. 21. Darlinghurst, Australia: Australian Computer Society, Inc., 2003. xxi, 13

[109] J. Lubowitz, "Bsa news release," The Software Alliance, Tech. Rep., 2014. [Online]. Available: http://globalstudy.bsa.org/2013/downloads/press/pr_global_en.pdf xxvii, 2

[110] B. Lynn, M. Prabhakaran, and A. Sahai, "Positive results and techniques for obfuscation," in *Advances in Cryptology - EUROCRYPT 2004*, C. Cachin and J. Camenisch, Eds., vol. LNCS 3027. Springer, 2004, pp. 20–39. 79

[111] S. Mabtoul, E. Ibn-Elhaf, and D. Aboutajdine, "A blind chaos-based complex wavelet-domain image watermarking technique," *IJCSNS - International Journal of Computer Science and Network Security*, vol. 6, no. 3, pp. 134–139, 2006. 111, 130

[112] M. Maes, T. Kalker, J.-P. M. Linnartz, J. Talstra, G. F. Depovere, and J. Haitsma, "Digital watermarking for DVD video copy protection," *IEEE Signal Processing Magazine*, pp. 47–57, September 2000. 16

[113] R. McMillan, "Settlement ends sony rootkit case," May 2006. [Online]. Available: http://www.pcworld.com/article/id.125838-page.1- 20

[114] W. Michiels and P. Gorissen, "Mechanism for software tamper resistance: An application of white-box cryptography," in *Proceedings of 7th ACM Workshop on Digital Rights Management*, M. Yung, A. Kiayias, and A. Reza-Sadeghi, Eds., vol. DRM'07. Alexandria, Virginia, USA: ACM Press, 2007, pp. 82–89. 81

[115] Microsoft, "Microsoft windows media DRM home page," 1999. [Online]. Available: http://www.microsoft.com/windows/windowsmedia/licensing/drmlicensing.aspx 18

[116] ——, "Protected media path," 2006. [Online]. Available: $http://msdn.microsoft.com/en-us/library/windows/desktop/aa376846%28v=vs.85%29.aspx$ 18

[117] C. Miller, "OpenIPMP seeks to break the DRM stalemante with release of open source, open-standard system." 23

[118] M. L. Miller, I. J. Cox, and J. A. Bloom, "Watermarking in the real world: An application to DVD," in *Proceedings of the 33rd Asilomar Conference on Signals, System and Computer*. IEEE, 1999, pp. 1496–1502. xxi, 17

[119] N. Morimoto, "Digital watermarking technology with practical applications," *Informing Science Special Issue on Multimedia Informing Technologies*, vol. 2, no. 4, pp. 107–111, 1999. 37

[120] Y. D. Mulder, B. Wyseur, and B. Preneel, "Cryptanalysis of a pertubated white-box AES implemenation," in *Progress in Cryptology - INDOCRYPT 2010*, G. Gong and K.C.Gupta, Eds., vol. LNCS 6498. Berlin Heidelberg: Springer-Verlag, 2010, pp. 292–310. 81

[121] D. K. Mulligan, J. Han, and A. J. Burstein, "How DRM-based content delivery systems disrupt expectations of personal use," in *DRM'03*. ACM, October 2003. 32

[122] R. Munir, B. Riyanto, S. Sutikno, and W. P. Agung, "Metode blind image watermarking berbasis chaos dalam ranah discrete cosine transform (DCT)," in *National Conference on Computer Science and Information Technology 7*, 2007. 111, 130

[123] ——, "Modifikasi spread spectrum watermarking dari Cox berbasiskan pada enkripsi chaotic," 2007. [Online]. Available: http://www.academia.edu/1357660 110, 131

[124] M. Naor, "The privacy of tracing traitors," in *Proceedings of the Tenth Annual ACM Workshop on Digital Rights Management*, E. Al-Shaer, H. Jin, and M. Joye, Eds., vol. DRM'10. ACM, 2011, pp. 1–2. 135

[125] M. Naor and B. Pinkas, "Oblivious transfer and polynomial evaluation," in *Proceedings of the Thirty-first Annual ACM Symposium on Theory of Computing*, ser. STOC '99. New York, NY, USA: ACM, 1999, pp. 245–254. [Online]. Available: http://doi.acm.org/10.1145/301250.301312 83

[126] ——, "Oblivious transfer with adaptive queries," in *CRYPTO'99*, M. Wiener, Ed., vol. LNCS 1666, 1999, pp. 573–590. 83

[127] ——, "Distributed oblivious transfer," in *ASIACRYPT 2000*, T. Okamoto, Ed., vol. LNCS 1976, 2000, pp. 205–219. 98

[128] ——, "Efficient oblivious transfer protocols," in *Proceedings of the Twelfth Annual ACM-SIAM Symposium on Discrete Algorithms*, vol. SODA'01, 2001, pp. 448–457. 99

[129] P. Newswire, "Leading open-source, open-standard DRM project OpenIPMP announces new platform version, boasting cross-platform oma, ismacryp, mpeg-2 and mpeg-4 support," 2006. [Online]. Available: http://enterprisearchitecture.ulitzer.com/node/247970 24

[130] H. Nyeem, W. Boles, and C. Boyd, "On the robustness and security of digital image watermarking," in *Proceedings of International Conference on Informatics and Vision (In Press)*. Dhaka, Bangladesh: IEEE, 2012. 107

[131] Objectlab, "OpenIPMP," 2003. [Online]. Available: www.objectlab.com 23

[132] T. S. of Queensland, "Digital rights management technologies and public records - a guideline for queensland public authorities," The State of Queensland, Tech. Rep. 1.0, February 2010. [Online]. Available: http://www.archives.qld.gov.au/ 34

[133] U. S. C. Office, "The digital millenium copyright act of 1998," December 1998. [Online]. Available: http://www.copyright.gov/legislation/dmca.pdf 3

[134] ——, "Fair use," November 2009. [Online]. Available: www.copyright.gov/fls/fl102.html 38

[135] ——, "Circular 92: Copyright law of the United States and related laws contained in title 17 of the United States code," December 2011. [Online]. Available: www.copyright.gov/title17/circ92.pdf 38

[136] OMA, "DRM architecture v 2.0," March 2004. [Online]. Available: www.openmobilealliance.org 22

[137] ——, "DRM architecture v 2.1," October 2008. [Online]. Available: www.openmobilealliance.org/ xxi, 22

[138] ——, "DRM rights expression language v 2.1," October 2008. [Online]. Available: www.openmobilealliance.org/ 22

[139] J. Park and R. Sandhu, "The UCON-ABC usage control model," *ACM Transactions on Information and System Security*, vol. 7, no. 1, pp. 128–174, February 2004. xxiii, 169, 170

[140] J. Park, X. Zhang, and R. Sandhu, "Atrribute mutability in usage control," in *18th Annual Conference on Data and Applications Security*, ser. International Federation for Information Processing, C. Farkas and P. Samarati, Eds., vol. 144. Springer US, 2004, pp. 15–29. 169

[141] V.-V. Patriciu, I. Bica, M. Togan, and S.-V. Ghita, "A generalized DRM architectural framework," *Advances in Electrical and Computer Engineering*, vol. 11, no. 1, pp. 43–48, 2011. [Online]. Available: www.aece.ro xxi, 11, 29, 30

[142] F. Perez-Gonzalez and J. R. Hernandez, "A tutorial on digital watermarking," in *Proceedings IEEE 33rd Annual 1999 International Carnahan Conference on Security Technology*, 1999, pp. 286–292. 106

[143] R. Perlman, "The ephemerizer: Makin data disappear," *Journal of Information System Security*, vol. 1, no. 1, 2005. 52

[144] R. Perlman, C. Kaufman, and R. Perlner, "Privacy-preserving DRM," in *Preceedings of the 9th Symposium on Identity and Trust on the Internet*, ser. IDTRUST'10, 2010, pp. 69–83. 33, 40, 47, 51, 57, 64

[145] F. A. Petitcolas, "Components of DRM systems: Digital watermarking," in *Digital Rights Management: Technological, Economic, Legal and Political Aspects*, E. Becker, W. Buhse, D. Gunnewig, and N. Rump, Eds., vol. LNCS 2770. Berlin Heidelberg: Springer-Verlag, 2003, pp. 81–92. 36, 107, 108

[146] R. Petrlic and C. Sorge, "Privacy-preserving drm for cloud computing," in *26th International Conference on Advanced Information Networking and Applications Workshops*, IEEE. IEEE Computer Society, 2012, pp. 1286–1291. 33, 64

[147] A. Piva, F. Bartolini, and M. Barni, "Managing copyright in open networks," *IEEE Internet Computing*, vol. 6, no. 3, pp. 18–26, 2002. 37

[148] B. Podder, "DRM system: A comparative study of protecting owner and consumer rights," *Bulletin of Applied Computing and Information Technology*, vol. 3, no. 2, pp. 1–7, 2005. 4, 35, 38

[149] C. I. Podilchuk and E. J. Delp, "Digital watermarking: Algorithms and applications," *IEEE Signal Processing Magazine*, vol. CERIAS Tech Report 2001-71, pp. 33–46, July 2001. 102

[150] M. Prasad and S. Koliwad, "A comprehensive survey of contemporary researches in watermarking for copyright protection of digital images," *International Journal of Computer Science and Network Security*, vol. 9, no. 4, pp. 91–107, 2009. 37

[151] A. Pretschner, M. Hilty, and D. Basin, "Distributed usage control," *Communication of the ACM*, vol. 49, no. 9, pp. 39–44, September 2006. 31

[152] A. Pretschner, M. Hilty, F. Schutz, C. Schaefer, and T. Walter, "Usage control enforcement, present and future," *IEEE Security and Privacy*, vol. 6, no. 4, pp. 44–53, 2008. 31

[153] A. C. Prihandoko, H. Ghodosi, and B. Litow, "Deterring traitor using double encryption scheme," in *Proceedings of the IEEE International Conference on Communication, Network and Satellite*, vol. IEEE COMNETSAT 2013. Yogyakarta, Indonesia: IEEE Xplore Digital Library, December 2013, pp. 100–104. [Online]. Available: http://ieeexplore.ieee.org/xpl/articleDetails.jsp?arnumber=6870869 145

[154] ——, "Obfuscation and WBC: Endeavour for securing encryption in the DRM context," in *Proceedings of the International Conference on Computer Science and Information Technology*, T. Herawan, N. Ahmad, and P. Vitasari, Eds., vol. CSIT 2013. Yogyakarta, Indonesia: Universitas Ahmad Dahlan Press, June 2013, pp. 150–155. 73

[155] ——, "Secure and private content distribution in the DRM environment," in *Proceedings of the Information System International Conference*, A. Herdiyanti, M. P. Syafitri, and R. V. Margareta, Eds., vol. ISICO 2013. Bali, Indonesia: Department of Information System, ITS, Surabaya, December 2013, pp. 659–664. [Online]. Available: http://is.its.ac.id/pubs/oajis/index.php/home/detail/1156/Secure-and-Private-Content-Distribution-in-the-DRM-Environment 86

[156] ——, "Blind image watermarking based on chaotic maps," *IT in Industry*, vol. 2, no. 2, pp. 44–50, September 2014. [Online]. Available: http://www.it-in-industry.org/index.php/itii/article/view/27 123

[157] A. C. Prihandoko, B. Litow, and H. Ghodosi, "DRM's rights protection capability: A review," in *Proceeding of The First International Conference on Computational Science and Information Management*, B. B. Nasution, M. Zarlis, J. M. Zain, R. W. Sembiring, and T. Herawan, Eds., vol. 1. Medan, Indonesia: Universiti Malaysia Pahang Press, December 2012, pp. 12–17. 34, 42

[158] M. O. Rabin, "How to exchange secrets with oblivious transfer," Aiken Computation Lab, Harvard University, Technical Report TR-81, 1981. 83

[159] T. Rayna and L. Striukova, "Privacy or piracy, why choose? two solutions to the issues of digital rights management and the protection of personal information," *International Journal Intellectual Property Management*, vol. 2, no. 3, pp. 240–252, 2008. 39

[160] I. Reed and G. Solomon, "Polynomial codes over certain finite fields," *Journal of the Society for Industrial and Applied Mathematics*, vol. 8, no. 2, pp. 300–304, June 1960. 143, 148

[161] R. Rivest, A. Shamir, and L. Adleman, "A method for obtaining digital signature and public-key cryptosystems," *Communications of the ACM*, vol. 21, no. 2, pp. 120–126, 1978. 52, 150

[162] R. L. Rivest, "Unconditionally secure commitment and oblivious transfer schemes using private channels and a trusted initializer," Tech. Rep., 1999, http://people.csail.mit.edu/rivest/pubs/Riv99d.pdf. 98

[163] B. Rosenblatt, "DRM, law and technology: an american perspective," *Online Information Review*, vol. 31, no. 1, pp. 73–84, 2007. 12

[164] N. Rumps, "Definition, aspects, and overview," in *Digital Rights Management: Technological, Economic, Legal and Political Aspects*, E. Becker, W. Buhse, D. Gunnewig, and N. Rump, Eds., vol. LNCS 2770. Berlin Heidelberg: Springer-Verlag, 2003, pp. 3–15. 12

[165] A. Russ, "Digital right management overview," *Security Essential*, vol. 1, no. 2e, pp. 1–11, July 2001. 34

[166] R. Safavi-Naini, N. Sheppard, and T. Uehara, "Import export in digital rights management," in *ACM Workshop on Digital Rights Management*, ACM 2004. Research Online the University of Wollongong, 2004, pp. 99–110. [Online]. Available: http://ro.uow.edu.au/infopapers/442 39

[167] R. Safavi-Naini and Y. Wang, "Sequential traitor tracing," *IEEE Transaction on Information Theory*, vol. 49, no. 5, pp. 1319–1326, May 2003. 137, 142, 143

[168] SafeNet, "Understanding white box cryptography," 2012. [Online]. Available: www.safenet-inc.com 72

[169] A. Saxena, B. Wyseur, and B. Preneel, "Towards security notions for white-box cryptography," in *Information Security ISC 2009*, e. P. Samarati, Ed., vol. LNCS 5735. Berlin Heidelberg: Springer-Verlag, 2009, pp. 49–58. 71

[170] B. Schneier, "Dvd encryption broken," November 1999. [Online]. Available: https://www.schneier.com/essay-193.html 16

[171] R. Schultz, "The many facades of DRM," *MISC HS 5 Magazine*, pp. 58–64, April 2012. xxi, 68, 70, 72, 80

[172] D. Seng, *Copyright Law, Digital Content and the Internet in the Asia-Pacific*. Sydney University Press, March 2008, ch. Regulation of the Interactive Digital Media Industry in Singapore, pp. 67–118. 3

[173] C. Serrao, M. Dias, J. Marques, and J. Delgado, "Open source software as a driver for digital content e-commerce and DRM interoperability," in *Europe-Chine Conference on Intellectual Property in Digital Media*, vol. IPDM'06, 2006. xxi, 24

[174] A. Shamir, "How to share a secret," *Communications of the ACM*, vol. 22, no. 11, pp. 612–613, 1979. 84, 87, 89, 93

[175] S. E. Siwek, "The true cost of motion picture piracy to the U.S. economy," Institute for Policy Innovation, Policy Report 186, September 2006. 2

[176] M. Soriano, S. Flake, J. Tacken, F. Bormann, and J. Tomas, "Mobile digital rights management: Security requirements and copy detection mechanisms," in *Proceedings of the 16th International Workshop on Database and Expert Systems Applications*, vol. DEXA'05. IEEE - Computer Society, 2005. 23

[177] G. Spenger, "Components of DRM systems: Authentication, identification techniques, and secure containers - baseline technologies," in *Digital Rights Management: Technological, Economic, Legal and Political Aspects*, E. Becker, W. Buhse, D. Gunnewig, and N. Rump, Eds., vol. LNCS 2770. Berlin Heidelberg: Springer-Verlag, 2003, pp. 62–80. 4

[178] M. Stamp, "Digital rights management: For better or for worse?" May 2003. [Online]. Available: www.eweek.com 34

[179] D. R. Stinson, *Chryptography, Theory and Practice*, K. Rosen, Ed. CRC Press LLC, 1995. 84, 85

[180] R. Suehle, "The DRM graveyard: A brief history of digital rights management in music," November 2011. [Online]. Available: http://opensource.com/life/11/11/drm-graveyard-brief-history-digital-rights-management-music 18, 19

[181] B. Surekha and G. Swamy, "A spatial domain public image watermarking," *International Journal of Security and Its Applications*, vol. 5, no. 1, pp. 1–11, January 2011. 102

[182] S. A. Taylor, C. Ishida, and D. W. Wallace, "Intention to engage in digital piracy: A conceptual model and empirical test," *Journal of Service Research*, vol. 11, no. 3, pp. 246–262, February 2009. 35

[183] T. Thomas, S. Emmanuel, A. Das, and M. S. Kankanhalli, "A CRT based watermark for multiparty multilevel DRM architecture," in *IEEE International Conference on Multimedia and Expo*. IEEE Xplore Digital Library, 2009, pp. 1010–1013. 37

[184] ——, "Secure multimedia content delivery with multiparty multilevel DRM architecture," in *Proceedings of the 18th International Workshop on Network and Operating System Support for Digital Audio and Video*, vol. 09, ACM. ACM Digital Library, 2009, pp. 85–90. 37

[185] T. Thomas, S. Emmanuel, A. Subramanyam, and M. S. Kankanhalli, "Joint watermarking scheme for multiparty multilevel DRM architecture," *IEEE Transactions on Information Forensics and Security*, vol. 4, no. 4, pp. 758–767, December 2009. 37

[186] S. Tibken, "Apple's iPod antitrust class action suit: All you need to know," December 2014. [Online]. Available: http://www.cnet.com/au/news/apples-ipod-antitrust-class-action-suit-all-you-need-to-know-faq/ 19

[187] U. Topkara, M. Topkara, and M. J. Atallah, "The hiding virtues of ambiguity: Quantifiably resilient watermarking of natural language text through synonym substitution," in *Proceeding of ACM Multimedia and Security Conference*, Geneva, 2006. 101

[188] J. Trevathan, H. Ghodosi, and B. Litow, "Overview of traitor tracing schemes," *Communications of CCISA, Selected Topics of Cryptography and Information Security*, vol. 9, no. 4, pp. 51–63, 2003. 138

[189] W.-G. Tzeng, "Efficient 1-out-n oblivious transfer schemes," in *PKC 2002*, D. Naccache and P. Paillier, Eds., vol. LNCS 2274, 2002, pp. 159–171. 83

[190] ——, "Efficient 1-out-of-n oblivious transfer schemes with universally usable parameters," *IEEE Transactions on Computers*, vol. 53, no. 2, pp. 232–240, 2004. 83

[191] W.-G. Tzeng and Z.-J. Tzeng, "A public-key traitor tracing scheme with revocation using dynamic shares," in *Public Key Cryptography (PKC 2001)*, K. Kim, Ed., vol. LNCS 1992. Springer-Verlag, 2001, pp. 207–224. 139

[192] H. Wang, C. He, and K. Ding, "Public watermarking based on chaotic map," *IEICE Trans. Fundamentals*, vol. E87-A, no. 8, pp. 2045–2047, August 2004. 110, 131

[193] Q. Wang, *Copyright Law, Digital Content and the Internet in the Asia-Pacific*. Sydney University Press, March 2008, ch. The New Right of Communication Through the Information Network in the People's Republic of China, pp. 187–217. 3

[194] X. Wang, "MPEG-21 rights expression language: enabling interoperable digital rights management," *Multimedia, IEEE*, vol. 11, no. 4, pp. 84 – 87, oct.-dec. 2004. 24

[195] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, "Image quality assessment: From error visibility to structural similarity," *IEEE Transactions on Image Processing*, vol. 13, no. 4, pp. 600–612, April 2004. 106

[196] P. Wolf, M. Steinbach, and K. Diener, "Complementing DRM with digital watermarking: Mark, search, retrieve," *Online Information Review*, vol. 31, no. 1, pp. 10–21, 2007. 34

[197] C.-S. Woo, "Digital image watermarking methods for copyright protection and authentication," Ph.D. dissertation, Queensland University of Technology, March 2007. 36

[198] C.-S. Woo, J. Du, and B. L. Pham, "Geometrically robust digital image watermarking using scale normalisation and flowline curvature," in *Proceedings Security, Steganography and Watermarking of Multimedia Contents*, E. Delp, P. Wong, and A. Wendelborn, Eds., vol. SPIE2005, no. 5681, San Jose, California, 2005, pp. 528–538. 104

[199] X. Wu and Z.-H. Guan, "A novel digital watermark algorithm based on chaotic maps," *Physics Letters A*, vol. 365, pp. 403–406, 2007. 111, 113, 115, 131

[200] X. Wu, Z.-H. Guan, and Z. Wu, "A chaos based robust spatial domain watermarking algorithm," in *ISNN 2007, Part II*, D. Liu, Ed., vol. LNCS 4492. Berlin Heidelberg: Springer-Verlag, 2007, pp. 113–119. 111, 113, 115, 131

[201] B. Wyseur, "White-box cryptography," PhD Thesis, Katholieke Universitiet Leuven, March 2009. 68

[202] ——, "White-box cryptography: Hiding keys in software," *MISC HS 5 Magazine*, pp. 65–72, April 2012. xxi, 68, 71, 72

[203] C. Xu, *Copyright Law, Digital Content and the Internet in the Asia-Pacific*. Sydney University Press, March 2008, ch. Copyright Protection in the People's Republic of China, pp. 359–366. 3

[204] J. Yoo, H. Jeong, and D. Won, "A method for secure and efficient block cipher using white-box cryptography," in *6th International Conference on Ubiquitos Information Management and Communication*, vol. ICUIMC'12, 2012. xxvii, 81, 82

[205] X. Zhang, "A survey of digital rights management technology," 2011. [Online]. Available: http://www.cse.wustl.edu/~jain/cse571-11/ftp/drm.pdf 34

[206] Z. Zhang, "Digital rights management ecosystem and its usage controls: A survey," *International Journal of Digital Content Technology and its Applications*, vol. 5, no. 3, pp. 255–272, March 2011. 4

[207] Z. Zhang, Q. Pei, J. Ma, and L. Yang, "Security and trust in digital rights management: A survey," *International Journal of Network Security*, vol. 9, no. 3, pp. 247–263, 2009. 40

[208] B. Zhu, M. Feng, F. Liu, and L. Hu, "Analysis on AACS' traitor tracing against mix-and-match attacks," in *IEEE CCNC 2008 Proceedings*, 2008, pp. 1097–1101. 155