1

PM-1 3½"x4" PHOTOGRAPHIC MICROCOPY TARGET
NBS 1010a ANSI/ISO #2 EQUIVALENT

PRECISIONSM RESOLUTION TARGETS

THRESHOLD DESIGNATED CONFIRMER SIGNATURE

By

Frankie Ho Yin Kwok, B.C Sc.

A thesis submitted to the Faculty of
Graduate Studies and Research in partial
fulfillment of the requirements for the
degree of Master of Computer Science

School of Computer Science

Carleton University

Ottawa, Ontario

April 16, 1996

© copyright

1996, Frankie Ho Yin Kwok

National Library
of Canada

Bibliothèque nationale
du Canada

Acquisitions and
Bibliographic Services Branch

Direction des acquisitions et
des services bibliographiques

395 Wellington Street
Ottawa, Ontario
K1A 0N4

395, rue Wellington
Ottawa (Ontario)
K1A 0N4

Your file Votre référence

Our file Notre référence

The author has granted an irrevocable non-exclusive licence allowing the National Library of Canada to reproduce, loan, distribute or sell copies of his/her thesis by any means and in any form or format, making this thesis available to interested persons.

L'auteur a accordé une licence irrévocable et non exclusive permettant à la Bibliothèque nationale du Canada de reproduire, prêter, distribuer ou vendre des copies de sa thèse de quelque manière et sous quelque forme que ce soit pour mettre des exemplaires de cette thèse à la disposition des personnes intéressées.

The author retains ownership of the copyright in his/her thesis. Neither the thesis nor substantial extracts from it may be printed or otherwise reproduced without his/her permission.

L'auteur conserve la propriété du droit d'auteur qui protège sa thèse. Ni la thèse ni des extraits substantiels de celle-ci ne doivent être imprimés ou autrement reproduits sans son autorisation.

ISBN 0-612-13831-3

Canada

Name

Travis H. Van Lue

Dissertation Abstracts International is arranged by broad, general subject categories. Please select the one subject which most nearly describes the content of your dissertation. Enter the corresponding four-digit code in the spaces provided.

Computer Science

SUBJECT TERM

0319

U·M·I

SUBJECT CODE

Subject Categories

THE HUMANITIES AND SOCIAL SCIENCES

COMMUNICATIONS AND THE ARTS

Architecture 0729
 Art History 0377
 Cinema 0700
 Dance 0378
 Fine Arts 0357
 Information Science 0723
 Journalism 0391
 Library Science 0399
 Mass Communications 0708
 Music 0413
 Speech Communication 0459
 Theater 0465

EDUCATION

General 0515
 Administration 0514
 Adult and Continuing 0516
 Agricultural 0517
 Art 0273
 Bilingual and Multicultural 0282
 Business 0688
 Community College 0275
 Curriculum and Instruction 0727
 Early Childhood 0518
 Elementary 0524
 Finance 0277
 Guidance and Counseling 0519
 Health 0680
 Higher 0745
 History of 0520
 Home Economics 0278
 Industrial 0521
 Language and Literature 0279
 Mathematics 0280
 Music 0522
 Philosophy of 0998
 Physical 0523

Psychology 0525
 Reading 0535
 Religious 0527
 Sciences 0714
 Secondary 0533
 Social Sciences 0534
 Sociology of 0340
 Special 0529
 Teacher Training 0530
 Technology 0710
 Tests and Measurements 0288
 Vocational 0747

LANGUAGE, LITERATURE AND LINGUISTICS

Language 0679
 General 0289
 Ancient 0290
 Linguistics 0291
 Modern 0401
 Literature 0294
 General 0295
 Classical 0297
 Comparative 0298
 Medieval 0298
 Modern 0316
 African 0591
 American 0305
 Asian 0352
 Canadian (English) 0355
 Canadian (French) 0593
 English 0311
 Germanic 0312
 Latin American 0315
 Middle Eastern 0313
 Romance 0314
 Slavic and East European

PHILOSOPHY, RELIGION AND THEOLOGY

Philosophy 0422
 Religion 0318
 General 0321
 Biblical Studies 0319
 Clergy 0320
 History of 0322
 Philosophy of 0469
 Theology

SOCIAL SCIENCES

American Studies 0323
 Anthropology 0324
 Archaeology 0326
 Cultural 0327
 Physical 0310
 Business Administration 0272
 General 0770
 Accounting 0454
 Banking 0338
 Management 0385
 Canadian Studies 0561
 Economics 0503
 General 0505
 Agricultural 0508
 Commerce-Business 0509
 Finance 0510
 History 0511
 Labor 0358
 Theory 0366
 Folklore 0351
 Geography 0578
 Gerontology 0578
 History 0578
 General

Ancient 0579
 Medieval 0581
 Modern 0328
 Black 0331
 African 0332
 Asia, Australia and Oceania 0334
 Canadian 0335
 European 0336
 Latin American 0337
 Middle Eastern 0585
 United States 0398
 History of Science 0615
 Law 0616
 Political Science 0617
 General 0814
 International Law and 0452
 Relations 0626
 Public Administration 0627
 Recreation 0938
 Social Work 0631
 Sociology 0628
 General 0629
 Criminology and Penology 0630
 Demography 0700
 Ethnic and Racial Studies 0344
 Individual and Family 0709
 Studies 0999
 Industrial and Labor 0453
 Relations 0537
 Public and Social Welfare 0539
 Social Structure and 0540
 Development 0541
 Theory and Methods 0542
 Transportation 0543
 Urban and Regional Planning 0544
 Women's Studies

THE SCIENCES AND ENGINEERING

BIOLOGICAL SCIENCES

Agriculture 0473
 General 0285
 Agronomy 0475
 Animal Culture and 0476
 Nutrition 0359
 Animal Pathology 0478
 Food Science and 0479
 Technology 0480
 Forestry and Wildlife 0817
 Plant Culture 0777
 Plant Pathology 0746
 Plant Physiology 0306
 Range Management 0287
 Wood Technology 0308
 Biology 0309
 General 0329
 Anatomy 0353
 Biostatistics 0369
 Botany 0793
 Cell 0410
 Ecology 0307
 Entomology 0317
 Genetics 0416
 Immunology 0433
 Limnology 0821
 Microbiology 0778
 Molecular 0472
 Neurosciences 0786
 Oceanography 0760
 Physiology 0425
 Radiation 0996
 Veterinary Science 0425
 Zoology 0996
 Biophysics 0425
 General 0996
 Medical 0425
 Earth Sciences 0425
 Biogeochemistry 0996
 Geochemistry

Geology 0370
 Geology 0372
 Geophysics 0373
 Hydrology 0388
 Mineralogy 0411
 Paleobotany 0345
 Paleoclimatology 0426
 Paleogeography 0418
 Paleontology 0985
 Palynology 0427
 Physical Geography 0368
 Physical Oceanography 0415

HEALTH AND ENVIRONMENTAL SCIENCES

Environmental Sciences 0768
 Health Sciences 0566
 General 0300
 Audiology 0992
 Chemotherapy 0567
 Dentistry 0350
 Education 0769
 Hospital Management 0758
 Human Development 0982
 Immunology 0564
 Medicine and Surgery 0347
 Mental Health 0569
 Nursing 0570
 Nutrition 0380
 Obstetrics and Gynecology 0354
 Occupational Health and 0381
 Therapy 0571
 Ophthalmology 0419
 Pathology 0572
 Pharmacology 0382
 Pharmacy 0573
 Physical Therapy 0574
 Public Health 0575
 Radiology 0575
 Recreation

Speech Pathology 0460
 Toxicology 0383
 Home Economics 0386

PHYSICAL SCIENCES

Pure Sciences 0485
 Chemistry 0749
 General 0486
 Agricultural 0487
 Analytical 0488
 Biochemistry 0738
 Inorganic 0490
 Nuclear 0491
 Organic 0494
 Pharmaceutical 0495
 Physical 0754
 Polymer 0405
 Radiation 0605
 Mathematics 0986
 Physics 0606
 General 0608
 Acoustics 0748
 Astronomy and 0607
 Astrophysics 0798
 Atmospheric Science 0759
 Atomic 0609
 Electronics and Electricity 0610
 Elementary Particles and 0752
 High Energy 0756
 Fluid and Plasma 0611
 Molecular 0463
 Nuclear 0346
 Optics 0984
 Radiation 0346
 Solid State 0984
 Statistics 0346
 Applied Sciences 0346
 Applied Mechanics 0984
 Computer Science

Engineering 0537
 General 0538
 Aerospace 0539
 Agricultural 0540
 Automotive 0541
 Biomedical 0542
 Chemical 0543
 Civil 0544
 Electronics and Electrical 0348
 Heat and Thermodynamics 0545
 Hydraulic 0546
 Industrial 0547
 Marine 0794
 Materials Science 0548
 Mechanical 0743
 Metallurgy 0551
 Mining 0552
 Nuclear 0549
 Packaging 0765
 Petroleum 0554
 Sanitary and Municipal 0790
 System Science 0428
 Geotechnology 0796
 Operations Research 0795
 Plastics Technology 0994
 Textile Technology

PSYCHOLOGY

General 0621
 Behavioral 0384
 Clinical 0622
 Developmental 0620
 Experimental 0623
 Industrial 0624
 Personality 0625
 Physiological 0989
 Psychobiology 0349
 Psychometrics 0632
 Social 0451

The undersigned recommend to
the Faculty of Graduate Studies and Research
acceptance of the thesis.

"Threshold Designated Confirmer Signature"

submitted by

Frankie Ho Yin Kwok, B.C.S.

in partial fulfillment of the requirements
for the degree of Master of Computer Science

A handwritten signature in black ink, appearing to be 'E. Kwok', written over a horizontal line.

Thesis Co-Supervisor

A large, stylized handwritten signature in black ink, written over a horizontal line.

Thesis Co-Supervisor

A handwritten signature in black ink, appearing to be 'E. Kwok', written over a horizontal line.

Director, School of Computer Science

Carleton University, Ottawa, Ontario

May 14, 1996

ABSTRACT

Since its introduction in 1978, the digital signature has become an area of growing research in the field of modern cryptography. With its inherent self-authenticity property, however, any illegitimate user who has knowledge of the signer's public key will be able to validate the signature.

This shortcoming can be resolved by the Designated Confirmer Signature Scheme, wherein any signature created by the signer can be verified only with the cooperation of a confirmer. We will propose the $(1,k)$ threshold designated confirmer signature, which consists of k confirmers, such that the signature can be verified by any 1 of the k confirmers. Additionally, we will discuss a variant of the $(1,k)$ designated confirmer signature that allows the verifier to confirm the signature with either the signer or any 1 out of the k confirmers.

ACKNOWLEDGMENTS

First and foremost, I wish to express my gratitude to my supervisor, Evangelos Kranakis, and my co-supervisor, Danny Krizanc. Thank you for having faith in me; taking me in, pointing out several interesting possible topics, the idea of zero-knowledge proof, and generally just finding time to help me when time didn't exist.

I also wish to thank my readers, Darren Michaud and John Clarke. Their aid in proof reading this paper was invaluable.

Lastly, I would like to express my sincerest gratitude to Paul Van Oorschot, the instructor of my first cryptography course. I could not have even attempted a thesis on this topic without all the background I learned from you.

DEDICATION

This thesis is dedicated to my parents. Thanks for all the love and support throughout my life. Thanks for always being there .. even when I am thousands miles away. Thanks for everything.

Love,

Frankie Ho Yin Kwok

TABLE OF CONTENTS

1. INTRODUCTION	1
1.1 Message Authentication	5
1.2 Public Key Cryptosystem (PKCS)	6
2. DIGITAL SIGNATURE	8
2.1 The Problem of Disputes	8
2.1.1 Properties of a Handwritten Signature	9
2.1.2 Properties of Digital Signature	10
2.2 The RSA Signature Scheme	14
2.2.1 Security of RSA System	20
3. UNDENIABLE DIGITAL SIGNATURE SCHEME	22
3.1 Undeniable Signature	25
3.1.1 Confirmation Protocol for the Undeniable Signature	26
3.1.2 Disavowal Protocol for the Undeniable Signature	29
3.1.3 Creating the Fake Transcript	34

3.1.4 Weakness of Undeniable Signature ?	35
3.2 Zero-Knowledge Undeniable Signature	37
3.2.1 Verification Protocol for the Zero-Knowledge Undeniable Signature	37
3.2.2 Disavowal Protocol for the Zero-Knowledge Undeniable Signature	41
4. DESIGNATED CONFIRMER SIGNATURE	16
4.1 (1,1) Threshold Confirmer Designated Signature	49
4.1.1 Signing Protocol for the (1,1) Designated Confirmer Signature.	52
4.1.2 Confirmation Protocol for the (1,1) Designated Confirmer Signature	58
4.1.3 Unforgability for the (1,1) Designated Confirmer Signature	62
4.1.4 Conversion Protocol.....	65
4.2 The (1,k) Threshold Designated Confirmer Signature	65
4.2.1 Signing Protocol for the (1,k) Designated Confirmer Signature.....	68
4.2.2 Confirmation Protocol	74
4.2.3 Unforgability of the (1,k) Threshold Designated Confirmer Signature	77
4.2.4 Conversion Protocol.....	81
4.3 The (1,1)* Designated Confirmer Signature.....	82
4.3.1 Security of the (1,1)* Designated Confirmer Signature	87

5. CONCLUSION	88
APPENDIX A. ZERO-KNOWLEDGE PROOF	94
APPENDIX B. BIT COMMITMENT SCHEME	104
APPENDIX C. SHAMIR SECRET SHARING SCHEME AND SECURE MESSAGE BROADCASTING	110
APPENDIX D. SECURE MESSAGE BROADCASTING	114
BIBLIOGRAPHY	119

LIST OF FIGURES

Figure 1.1. Principle of Message Authentication.....	6
Figure 1.2. Differences Between Symmetric-Key and Public-Key Cryptosystems.....	7
Figure 2.1. Digital Signature System.....	14
Figure 3.1. The Confirmation Protocol for Undeniable Signature.....	27
Figure 3.2. The Disavowal Protocol for Undeniable Signature.....	30
Figure 3.3. The Fake Transcript of the Undeniable Signature.....	34
Figure 3.4. Confirmation Protocol for the Zero-Knowledge Undeniable Signature.....	38
Figure 3.5. The Disavowal Protocol for the Zero-Knowledge Undeniable Signature.....	42
Figure 4.1. Designated Confirmer Signature.....	47
Figure 4.2. Signing Protocol for the (1,1) Designated Confirmer Signature.....	54
Figure 4.3. Confirmation Protocol for the (1,1) Designated Confirmer Signature.....	59
Figure 4.4. Signing Protocol for the (1,k) Designated Confirmer Signature.....	69
Figure 4.5. Confirmation Protocol for the (1,k) Designated Confirmer Signature.....	74
Figure 4.6. The Forged Transcript for the (1,k) Designated Confirmer Signature.....	80
Figure 4.7. Signer's Confirmation Protocol for the (1,1)* Designated Confirmer Signature.....	84
Figure 4.8. Confirmer's Confirmation Protocol for the (1,1)* Designated Confirmer Signature.....	86
Figure A1 Graph Isomorphism.....	93

Figure A2 Zero-Knowledge Proof System for the Graph Isomorphism Problem	95
Figure A3 The Polynomial-time Forging Algorithm for the Transcript for the Graph Isomorphism Proof System.....	101

1. Introduction

The importance of message authenticity has dramatically increased since the adoption of computer and communication technology in recent decades. Message authenticity implies that the message has never been altered since it left the sender. Additionally, the receiver must ensure that the message has originated from its designated source.

A simple method of achieving the message authenticity is to send the message directly through a physically protected channel. However, the high cost of producing such a channel usually makes this option an impossibility. An alternative is to use a message authentication code (details of the authentication code can be found in the following sections), which is a function of the message and a key shared by the sender and receiver. In this case any modification of the message will be detected unless an outsider knows the shared key. In addition, under the assumption that the key was shared between the sender and receiver, the receiver can ensure that the message originated from the designated sender.

Another stronger form of message authenticity can be achieved by the digital signature [DH76], wherein the signature is only computable by the signer who possesses a private signature key. Informally, the digital signature maintains protocols, similar to the traditional method of manually signing a document. The major differences between a digital and the handwritten signature is that the former depends on private information

from the signer and the document which it is affixed to. The traditional handwritten signature was seen as physical trait exclusively available to the signer. This signature could, therefore vary depending upon the signer's health, mental state, etc. A digital signature however, remains a constant, linked to personal information provided by the signer. This constancy is an inherent weakness in the digital signature scheme, one which is not present in the highly variable handwritten signature.

To lend the digital signature a uniqueness similar to its handwritten predecessor, a timestamp can be added. With this timestamp, the signature use can be tracked based upon the time and date affixed to the document.

Moreover, the signature is verifiable by public information, known as the public key. This property is similar to the handwritten signature, where signature verification is done by visually comparing the signature with an "authenticate signature".

The idea of the digital signature is strongly related to the public-key-cryptosystem. We will discuss details of the digital signature in chapter 2, and one of the most widely used digital signature schemes, known as the RSA signature scheme [RSA78], in section 2.1.

One of the most important properties of the digital signature is that signature verification does not require knowledge of the signature key. This means that verification based on the public verification key is designed such that any verifier (including the illegal users) is qualified for the verification. This property is known as the self-authenticity

property of the digital signature. As we will show in section 3.1, the self-authenticity property is too strong in many applications. That problem can be overcome using the undeniable signature [BCD90] [Cha90] [CA89], a variant of the digital signature in which the signature can only be confirmed with the cooperation of the signer. In this case the signer can prevent all illegal users from verifying the signature and thereby provide stronger protection for the signer. Signature verification is done by means of an interactive proof system which allows the signer to convince the verifier of the signature's validity without revealing the signature key. One major consequence of the undeniable signature scheme is that even if the verifier is convinced of the signature's validity, he is unable to prove it to others, as he does not know the signature key. Details of the undeniable signature can be found in chapter 3.

One major drawback of the undeniable signature is that if the signer is unavailable or refuses to cooperate, then the signature is useless to the verifier. This problem can be solved by the designated confirmer signature [Cha94], in which the responsibilities of signing and verification are separated into two distinct parties: the signer and the confirmer. We emphasize that in a designated confirmer signature, even if the signature was created by the signer, it is not possible to prove the signature's validity to any verifier. Informally, a (t,k) threshold designated confirmer signature consists of a signer and k confirmers, such that any subset of t confirmers are able to prove the signature's validity to the verifier. The first designated confirmer signature proposed by David Chaum, is essentially a $(1,1)$ designated confirmer signature scheme. This means there exists only one

confirmer and the signature can be confirmed only with the cooperation of the that confirmer. Details of the $(1,1)$ designated confirmer will be discussed in section 4.1.

However, with separation of the signing and verification responsibilities in the $(1,1)$ designated confirmer signature, the signature scheme suffers also from the same problem as the undeniable signature mentioned previously. If the confirmer is unavailable for the signature's verification, then none of the verifiers can make use of the signature. It is exactly the same problem as stated earlier, with the confirmer as the uncooperative party rather than the signer. In section 4.2, we will propose the $(1,k)$ designated confirmer signature scheme, in which a verifier can validate the signature with the cooperation of any one of k confirmers. The $(1,k)$ designated confirmer signature is distinctly more desirable than the $(1,1)$ designated confirmer signature, since verification no longer depends on a single party. Details of the $(1,k)$ designated confirmer signature can be found in section 4.2.

In section 4.3, we will also propose a variant of the $(1,1)$ designated signature known as the $(1,1)^*$ designated signature. As we mentioned before, the $(1,1)$ designated signature does not allow the signer to prove the signature validity to any verifier, even if the signature is created by the signer. With the proposed $(1,1)^*$ designated signature scheme, the signer is now able to prove the validity of the signature to any verifier. An advantage of such a signature scheme is that the signer is now responsible for the signature verification, and the confirmer is only involved whenever the signer is

unavailable. The properties are particularly important when there exist many signers and only a few confirmers.

What follows is a brief outline of this thesis. In chapter 2, we will discuss the details of the digital signature as well as the RSA signature scheme. The undeniable signature scheme will then be discussed in chapter 3. Finally, we will discuss the (1,1) designated confirmation scheme, followed by the newly proposed (1,k) and (1,1)* designated confirmer signature in chapter 4.

The following sections provides some necessary background for this thesis.

1.1 Message Authentication

A message, document or file is considered 'authenticated' if it has never been modified since it left the sender. There are two forms of 'unmodified message': it could be damaged due to the transmission; or some active attackers intentionally modify the message. For example, if a bank cheque was sent through a digital communication network, the receiver could take advantage of the sender by increasing the amount of the payment. In this case, nobody can judge the correctness of the message since it was totally unprotected over the communication channel.

A message authentication protocol ensures that the message which is received is identical to that which was initially sent, by attaching a field called an **authenticator**. Generally, the authenticator requires the sender and receiver to both agree on an authentication key, k , and an authentication function $F(\cdot, \cdot)$ prior to the message delivery.

Assuming that the sender wants to transmit a message m to the receiver, the sender has to attach the authenticator $a=F(m,k)$ to the message and the receiver will accept the incoming message m' if and only if $a=F(m',k)$. If authentication fails, the sender has to re-send the message or appropriate legal action should be taken.

Figure 1.1 Principle of Message Authentication

1.2 Public Key Cryptosystem (PKCS)

Consider the situation where A(lice) wants to send a secret s to B(ob) via some communication network. To avoid eavesdropping by opponents, A can send the message over a private channel, that is physically protected against the opponents' listening. The cost of constructing such a channel often makes this option an impossibility. Instead of sending the secret s , the sender might first encrypt the message s and send only the encrypted message to the receiver over an insecure channel. In this case, any message obtained by opponents would be encrypted to appear as random information.

Cryptosystems have been used for years to protect information from eavesdroppers. The strength of any cipher should depend on the privacy of the key, rather than the encryption/decryption algorithm. Conventional cryptosystems (symmetric-key

cryptosystems) require that both encryption and decryption employ the same key. In 1976, W.Diffie and M.E.Hellman [DH76] introduced the idea of a public-key cryptosystem, which requires different but related keys, one for encryption and one for decryption. The receiver is able to publish the encryption key while keeping the decryption key private. Senders who wish to send a private message m to the receiver can encrypt the message by the public encryption key of the receiver. The message is protected since the encrypted message can only be decrypted by the designated receiver. A major requirement of public key cryptosystems is that the availability of the public key does not reveal any partial information about the private key. Otherwise attackers can compute the private key from the public key, thus effectively decrypting any message for the receiver. The following diagrams demonstrate the differences between symmetric-key cryptosystems and public key cryptosystems.

Figure 1.2 Differences Between Symmetric-Key and Public-Key Cryptosystems

2. Digital Signature

This chapter is organized as follows: In section 2.1 we will first discuss some drawbacks of conventional message authentication and the advantages of digital signature over message authentication. In section 2.2, we will discuss one of the most widely used digital signature known as the RSA signature, proposed by R.L.Rivest, A.Shamir, and L Adleman [RSA78].

2.1 The Problem of Disputes

Message authentication can ensure that the receiver's incoming message is identical to its original one. If both the sender, Simon, and receiver, Rita, are honest then the message authentication code is highly effective for them. However, recall that the authentication key is shared by both the sender and the receiver. Therefore it is possible for Rita to compute the authenticator for any message m . Rita can later claim that the message m originated with Simon and it would be impossible to differentiate this. Similarly, Simon could carry out the same deception on Rita, since they are sharing the same authentication key.

The weakness of the message authentication code is due to the shared information of the authentication key. In this chapter, we are going to discuss a more powerful protocol, called a digital signature, which can solve the aforementioned problem. Informally, a digital signature is an "authenticator" that is computed by the originator in such a way that only the originator can create the authenticator. With that authenticator,

the verifier is able to verify the correctness of the authenticator without the knowledge of the private “authentication key”. The properties of the protocol are similar to the handwritten signature and will be discussed in the next section.

2.1.1 Properties of a Handwritten Signature

Handwritten signatures have been used for hundreds of years as legal handwritten closure between different parties. The signature traditionally guaranteed that the signer would adhere to, and respect all aspects of the contract, accepting responsibility for any breach of that agreement. Before discussing digital signatures, we will describe the handwritten signature by dividing it into two sub-protocols : **Signing and Verification protocols.**

The Signing Protocol :

The signing protocol is the process of creating the “personalized-image” (signature) on a document. Once the document is signed, the signer has to accept responsibility for the content of it. He cannot deny responsibility for articles in the document since forging of the handwritten signature is considered to be extremely difficult. The handwritten signature is considered as unforgable because the signing process is a physical trait of the signer, not subject to deliberate muscular control. In addition to the unforgability property, signatures also have the **non-**

transferability property: once the handwritten signature is permanently attached to the document, the signature itself becomes part of the official document and is non-transferable. Though the forger can “copy” the signature by physically cutting and re-attaching the signature to the forged document (it can be further enhanced by means of image processing), the forged document will lack legal validity since all legal institutions accept only the original signature.

The Verification Protocol:

The verification protocol is carried out by the verifier with a signed document. The verifier’s task is to judge the accuracy of the signature by visually inspecting the signature image. Notice that the verifier need not be able to create the signature, but he can easily validate the correctness of it. We will call this the **public verification property** of the signature.

2.1.2 Properties of Digital Signature

As computerized communication volume increases, the applicability of the handwritten signature as an authentication method has been drastically reduced. The digital signature is a way to maintain the ease of the conventional signature in a digital

manner. Like the conventional signature, any digital scheme can be divided into two sub protocols:

(P1) Signing Protocol :

The signature should depend on some private information that is only known by the signer. Informally, a digital signature is a binary string or a number, which appends to the message m such that it is infeasible to create a valid signature without the private key. Obviously, the signature should depend on the private key of the signer.

Additionally, we would like to maintain the non-transferability and unforgability properties for the digital signature scheme.

(C1) Non-transferability:

The signature should depend on the message m being signed. Otherwise the signature can be easily moved from document to document by cut and paste operations. The non-transferability property implies that the signature is a function of the message.

(C2) Unforgability:

Unforgability suggests that it was computationally infeasible for the attacker to compute any valid message-signature pair other than the previously computed message-

signature pairs. Formally, given k message-signature pairs, $S = \{(m_1, s_1), \dots, (m_k, s_k)\}$, it is computationally infeasible to compute any valid message-signature pair $(m', s') \notin S$ without the knowledge of the private key.

P1 and **C1** imply that the digital signature, s , corresponding to the message, m , should depend on both the message m and the private key.

Definition 2.1. *Digital signature scheme* consists of K_s , K_v , M , S where K_s , K_v , M and S are the private key, public key, message and signature spaces respectively.

Definition 2.2. The *signing function* $\text{Sig}: K_s \times M \rightarrow S$ accepts the private key of the signer, k_s , and the message, m , and returns a signature s .

Definition 2.3 A signature $s \in S$ (corresponding to message m) is *valid* if and only if $\text{Sig}(k_s, m) = s$, where k_s is the private key of the signer.

(P2) Verification Protocol:

The signature verification should be easy and feasible even without revealing the secret information from the signer. Informally, the signature should be verifiable by the signed message and some information publicly available to the verifier.

Definition 2.4 The verifying function $\text{Ver}: K_v \times M \times S \rightarrow \{0, 1\}$ returns 1 if and only if the signature is valid (definition 2.3).

Another requirement of the digital signature system is that the private and public information should be somehow related; however, the public information should not reveal any partial information about the private key. Figure 2.1 illustrates the idea of a digital signature scheme. Like the PKCS discussed earlier, it requires a key-generation key and key-generation algorithm for generating the signature and verification key.

Figure 2.1 Digital Signature System.

In the following sections, we will discuss one of the most widely used digital signature schemes, known as the RSA signature scheme [RSA78].

2.2 The RSA Signature Scheme

In 1978, R.L.Rivest, A.Shamir, and L.Adleman [RSA78] proposed the first realization of the PKCS system, known as the RSA cryptosystem. Their system is both a PKCS and digital signature scheme that is widely accepted and prevalent in industry. The

system is based on the intractability of the discrete logarithm and of factoring the product of two large primes.

Definition 2.5. Let Z_n be the field $\{0, \dots, n-1\}$ under the operation $(+, \times)$ and Z_n^* be the group $\{x | x \in Z_n, \gcd(x, n) = 1\}$ under the operation $(\times)$.

Definition 2.6. Totient function $\Phi(n)$ is given by $|Z_n^*|$.

Additionally, we will state the following theorems with the proofs omitted for the sake of simplicity. [Kra86]

Theorem 2.1. If $n=pq$, where p and q are prime, then $\Phi(n)=(p-1)(q-1)$.

Theorem 2.2. (Euler-Fermat Theorem). If $x \in Z_n^*$, then $x^{\Phi(n)} = 1 \pmod{n}$.

With the above definitions and theorems, we can now discuss the RSA signature scheme. Firstly, let $n=pq$ be the product of two large primes such that p and q are kept secret. Additionally, select $d \in Z_n^*$ as the private key and we can uniquely determine the public key e in such a way that $d \cdot e = 1 \pmod{\Phi(n)}$ by the Extended Euclidean Algorithm. The following table summarizes the private and public key of the RSA signature.

Private Key: d, p, q , where $d \in Z_{\Phi(n)}^*$

Public Key: n, e , where $e \in Z_{\Phi(n)}^*, n=pq$ and $d \cdot e = 1 \pmod{\Phi(n)}$

The message, signature, private and public key spaces are defined as Z_n, Z_n, Z_n^* and Z_n^* respectively.

To sign a message $m \in Z_n$, the signature is computed by the following signature function $\text{Sig}: Z_n^* \times Z_n \rightarrow Z_n$

$$\text{Sig}(n, d, m) = m^d \pmod{n}$$

To verify a signature $s = \text{Sig}(n, d, m)$ (corresponding to message m), we check the following condition:

$$m \stackrel{?}{=} s^e \pmod{n}$$

or equivalently, we can define the verification function as follows:

Definition 2.7. $\text{Ver}: Z_n^* \times Z_n \times Z_n \times Z \rightarrow \{0, 1\}$ where $\text{Ver}(e, m, s, n) = 1$ if and only if $m = s^e \pmod{n}$. Additionally, a signature is said to be valid if and only if $\text{Ver}(e, m, s, n) = 1$.

Correctness of the RSA Signature.

In this section, we will discuss the correctness of the RSA signature. In all of the following theorems and lemmas, we assume m , (p,q,d) and e are the message, the private and the public key respectively. Additionally, let $s=m^d \pmod n$ be the signature of message m .

Theorem 2.3. In a RSA signature scheme, if $m \in Z_n^*$, then $\text{Ver}(e,m,s,n)=1$.

Proof. It is sufficient to show that $m=s^e \pmod n$, $m \in Z_n^*$. Since d and e were selected in such a way that $d \cdot e \equiv 1 \pmod{\Phi(n)}$, we have $d \cdot e = 1 + C \cdot \Phi(n)$ for some integer C . Therefore,

$$\begin{aligned} s^e &= m^{de} \pmod n \\ &= m \cdot m^{C \cdot \Phi(n)} \pmod n \\ &= m \cdot 1 \pmod n && \text{(Euler-Fermat Theorem)} \\ &= m. \end{aligned}$$

This concludes that $\text{Ver}(e, m, s, n)=1$ (definition 2.7). $\square$

Lemma 2.1. if $n=pq$ and $i \in \mathbb{Z}_q$, where p and q are primes, then $(p \cdot i)^{\Phi(n)} = 1 \pmod{n}$

Proof. By the Euler-Fermat theorem, we have

$$(p \cdot i)^{q-1} = 1 \pmod{q}$$

$$\text{so } (p \cdot i)^{q-1} - 1 = kq, \quad \text{for some integer } k$$

$$(p \cdot i)^q - (p \cdot i) = (k \cdot i)pq$$

$$(p \cdot i)^q = (p \cdot i) \pmod{n}$$

$$(p \cdot i)^{q-1} = 1 \pmod{n}$$

$$(p \cdot i)^{(q-1)(p-1)} = 1 \pmod{n}$$

$$(p \cdot i)^{\Phi(n)} = 1 \pmod{n}. \quad \square$$

Lemma 2.2. If $n=pq$ and $i \in \mathbb{Z}_p$, where p and q are primes, then $(q \cdot i)^{\Phi(n)} = 1 \pmod{n}$

Proof. It can be proved by a similar technique to the one shown in the proof of lemma 2.1. $\square$

Theorem 2.4. In a RSA signature scheme, if $m \in Z_n/Z_n^*$, then $\text{Ver}(e, m, s, n) = 1$.

Proof. It is sufficient to show that $m = m^{d \cdot e} \pmod{n}$ for all $m \in Z_n/Z_n^*$. Since $m \in Z_n/Z_n^*$ implies (i) $m = p \cdot i$ or (ii) $m = q \cdot j$, where $i \in Z_q$ and $j \in Z_p$. First consider $m = p \cdot i$,

$$\begin{aligned} m^{d \cdot e} \pmod{n} &= (p \cdot i)^{d \cdot e} \pmod{n} \\ &= (p \cdot i)^{K \cdot \Phi(n) + 1} \pmod{n}, \text{ for some integer } K \\ &= ((p \cdot i)^{\Phi(n)})^K (p \cdot i) \pmod{n}, \end{aligned}$$

By lemma 2.1, we have

$$\begin{aligned} m^{d \cdot e} \pmod{n} &= 1^K (p \cdot i) \pmod{n} \\ &= m \pmod{n}. \end{aligned}$$

Similarly, by lemma 2.2, we can also prove $m = m^{d \cdot e} \pmod{n}$ if $m = q \cdot j$ where $j \in Z_p$.

Thus, this concludes that $\text{Ver}(e, m, s, n) = 1$ for all $m \in Z_n/Z_n^*$. $\square$

Theorem 2.5. For all messages m in Z_n , $m^{d \cdot e} \pmod{n} = m$, where d and e are the private and public keys of the RSA signature scheme respectively.

Proof. Theorem 2.3 implies $m^{d \cdot e} \pmod{n} = m$ for all messages m in Z_n^* and theorem 2.4 implies $m^{d \cdot e} \pmod{n} = m$ for all the messages in Z_n/Z_n^* . This concludes that for all messages in Z_n , $m^{d \cdot e} \pmod{n} = m$. $\square$

2.2.1 Security of RSA System

In this section, we will consider the security of the RSA signature system. Obviously, there are two options for the opponent, Oscar, who wants to break the system. The first choice is to find d by determining the discrete logarithm of $s = m^d \pmod{n}$ to the base of m . However, this option is infeasible since there is no known algorithm that can efficiently solve the problem. The second option is that Oscar has to solve the **RSA problem**, defined as follows:

Definition 2.8: Given n and $e \in Z_n^*$ such that $\gcd(e, n) = 1$, where n is the product of two large primes p and q . Then the RSA problem $RSA(e, n)$ is to find $d \in Z_{\Phi(n)}^*$ in such a way that $d \cdot e \equiv 1 \pmod{\Phi(n)}$.

A simple solution for solving the RSA problem is to determine the factorization of $n = pq$. Assuming that Oscar can find p and q , then he is now able to find $d \in Z_n^*$ such that $d \cdot e \equiv 1 \pmod{\Phi(n)}$ by the Extended Euclidean Algorithm. Therefore, we conclude that the RSA signature is breakable (in polynomial-time) if there exists a polynomial-time algorithm that requires $O(\log n)$ time to factor $n = pq$.

Definition 2.9. The problem $Factor(n)$ is the problem of factoring the integer n where n is the product of two large primes p and q .

Although we have already shown that solving the $Factor(n)$ problem will solve the $RSA(e,n)$ problem, it is an open problem whether the $Factor(n)$ and $RSA(e,n)$ problems are equivalent. This means that even if we have an efficient algorithm to solve the $RSA(e,n)$ problem, it does not imply the existence of an efficient algorithm for solving the $Factor(n)$ problem.

In addition to the private keys p and q , the totient function $\Phi(n)$ should also be kept secret; otherwise the RSA signature scheme is breakable. To see why this is so,

$$\text{let } \Phi(n) = (p-1)(q-1) \quad (1)$$

then substitute $q = n/p$ into (1) yields

$$p^2 - (n - \Phi(n) + 1)p + n = 0.$$

The above represents a quadratic equation with the unknown p ; therefore, the opponent can find p in $O(1)$ time if that $\Phi(n)$ is known. The above attack implies that the computation of the totient function $\Phi(n)$ is no easier than factoring n .

3. Undeniable Digital Signature Scheme

In this chapter, we will discuss a digital signature scheme where the signature can only be confirmed with the cooperation of the original signer. The digital signature scheme with this property is known as the undeniable signature and was originally proposed by David Chuam and Hans van Antwerpen [CA89]. Details of the undeniable signature will be discussed in section 3.1. In addition, we will also discuss the zero-knowledge undeniable signature [Cha90] in section 3.2, which is the first undeniable signature where the signature confirmation is based on a zero-knowledge interactive proof system [GMR89]. Advantages of the zero-knowledge undeniable signature [Cha90] over the undeniable signature [CA89] is that the former will give the verifier absolutely no extra information other than the fact "the signature is valid". Details of the zero-knowledge proof system [GMR89] can be found in Appendix A.

The digital signature scheme discussed in the last chapter allows a verifier to validate the signature using the verification key. One major problem is that once the recipient, Rita, obtains the signature, she is free to copy it and distribute it anywhere she wants, which could include to enemies of the signer. Consider the case where the message is extremely personal or commercially sensitive so that the signer does not wish his enemies to verify the correctness of the message. Unfortunately, the conventional digital signature scheme does not allow the signer to prohibit his enemies from verification, since the verification key is publicly available.

As stated in the above example, the self-authenticity property of the conventional digital signature is too strong for many applications. In 1989, David Chaum & Hans Van Antwerpen [CA89] introduced the idea of **undeniable signature**, which solves the above problem. The undeniable signature allows the signer to create a verifiable signature. However, unlike the conventional digital signature, the undeniable signature can only be confirmed with the interaction and cooperation of the original signer. Another important property of the undeniable signature is that once the verifier is convinced that the signature is valid, he is not able to further convince anybody else of the validity of the signature, as he does not know the private key.

Another application for the undeniable signature is, as stated by [Cha90]

'Consider, for example, the signature a software supplier may issue on its software, allowing customers to check that the software is genuine and unmodified. With undeniable signatures, only paying customers are able to verify the signature, and they are ensured that the supplier remains accountable for the software'

In section 3.1, we are going to discuss the first undeniable signature proposed by [CA89]. Additionally, we will also discuss the zero-knowledge undeniable signature in section 3.2, which give the verifier absolutely no extra information other than the validity of the signature. Both of the signature schemes consist of three protocols: signing, confirmation and disavowal protocol. The signing protocol allows the signer to create a

signature that corresponds to a particular message. However, the recipient can only confirm the signature's validity with the cooperation of the signer by the confirmation protocol. In case of the confirmation protocol failing to prove the validity of the signature, the disavowal protocol can be carried out by the signer and the verifier to determine the invalidity of the signature.

Before discussing the undeniable signature scheme, we need to define the zero-knowledge proof as follows ([Stin95] pp.393). Further information of the zero-knowledge proof system can be found in Appendix A.

Definition A7 Zero-knowledge interactive proof system. Suppose that we have a polynomial-time interactive proof system for a decision problem Π , and a polynomial-time simulator S . Denote the set of all possible transcripts that could be produced as a result of Peggy and Vicky carrying out the interactive proof with a yes-instance x by $T(x)$, and denote the set of all possible forged transcripts that could be produced by S by $F(x)$. For any transcript $t \in T(x)$, let $p_T(t)$ denote the probability that t is the transcript produced from the interactive proof. Similarly, for $t \in F(x)$, let $p_F(t)$ denote the probability that t is the (forged) transcript produced by S . Suppose that $T(x) = F(x)$, and for any $t \in T(x)$, suppose that $p_T(t) = p_F(t)$. (In other words, the set of real transcripts is identical to the set of forged transcripts, and the two probability distributions are identical) then we define the interactive proof system to be perfect zero-knowledge for Vicky.

3.1 Undeniable Signature

As mentioned above, the first undeniable signature was proposed by [CA89]. The signer possesses a private and a public key while the verifier possesses neither of them. The private key and public key for the signer are selected as follows: q and k are primes, such that $p=qk+1$ is also prime. In addition, select any $g \in \mathbb{Z}_p$ such that $g^k \neq 1 \pmod{p}$. We define the group $G=\langle g \rangle$. In this case, the group G will have a prime order q , and g is a primitive element of G . The private and public keys of the signer are summarized as follows:

Signer's Private key: $x \in \mathbb{Z}_q$

Signer's Public key: $h = g^x \pmod{p} \in G$

Additionally, we assume that the discrete logarithm in the group G is intractable; otherwise the opponent is able to recover the private key from the public key.

Signing Protocol:

To sign message $m \in G$, the signer computes the signature $z \in G$ as follows:

$$z = m^x \pmod{p} \quad \text{where } x \text{ is the private key of the signer.}$$

Clearly, forging a signature requires the adversary to recover the signer's secret key. This can be done by solving the discrete logarithm problem $z = m^x \pmod{p}$ to the base of m . However, this solution is infeasible since no efficient algorithm is known for solving the discrete logarithm problem.

Before discussion of the confirmation protocol and disavowal protocol, we will formally define the necessary conditions for a valid signature.

Definition 3.1 Function $Validate(m, z)$ will return 1 if and only if $z = m^x \pmod{p}$, where x is the private key of the signer, otherwise $Validate(m, z) = 0$.

Definition 3.2. A signature $z \in G$ (corresponding to the message $m \in G$) is *valid* if and only if $Validate(m, z) = 1$.

3.1.1 Confirmation Protocol for the Undeniable Signature

The confirmation protocol allows the signer to validate the signature $z = m^x \pmod{p}$. The verifier will accept the signature as genuine if $r = m^a g^b \pmod{p}$ holds at the end of the protocol. Details of the confirmation protocol are as follows:

Figure 3.1 The Confirmation Protocol for Undeniable Signature

Correctness of the Confirmation Protocol

Assuming that all the parties that participate in the confirmation protocol honestly follow the protocol and that the signature is valid, i.e. $z = m^x \pmod{p}$, then

$$\begin{aligned}
 r &= (z^a h^b)^{1/x} \pmod{p} \\
 &= (m^{xa} g^{xb})^{1/x} \pmod{p} \\
 &= m^a g^b \pmod{p}.
 \end{aligned}$$

Therefore, the verifier will be convinced of the validity of the signature.

Security of the Confirmation Protocol

Definition 3.3. Function $Confirm(m, r, a, b)$ will return 1 where 1 if and only if $r = m^a g^b \pmod{p}$, otherwise $Confirm(m, r, a, b) = 0$

Definition 3.4: In the confirmation protocol, the verifier will accept the signature $z \in G$ (corresponding to the message $m \in G$) as genuine if and only if $\text{Confirm}(m, r, a, b) = 1$, where r is the response from the signer and (a, b) are the random values chosen by the verifier.

Theorem 3.1. In the confirmation protocol as stated in figure 3.1, given an invalid signature $z \in G$ (definition 3.2), the probability of the verifier accepting z as genuine (definition 3.4) is given by $1/q$.

Proof: Assume the signer wants to convince the verifier to accept an invalid signature, i.e. $z \neq m^x \pmod{p}$. Clearly, his task is to return r in such a way that $r = m^a g^b \pmod{p}$. However, this is infeasible for the signer to solve a and b from the challenge $c = z^a h^b \pmod{p}$. Thus, he has to guess the correct value of $r \in G$ with probability $1/q$. This implies that the signer can convince the verifier to accept an invalid signature with probability $1/q$. $\square$

Theorem 3.2. The confirmation protocol discussed in section 3.1.1 is *not* zero-knowledge.

Proof. Assume the random values (a,b) are uniformly chosen over Z_q^2 . Consider the scenario whereby we wish to create a forged transcript for $z \neq m^x \pmod{p}$. The forger first selects $(a,b) \in_R Z_q^2$, and let $c = z^a h^b \pmod{p}$. In this case the forger must compute $r = c^{1/x} \pmod{p}$ in order to create an indistinguishable transcript. However the forger has no better chance of guessing the correct value of r with probability $1/q$ since he does not know the private key x . This implies that the probability distribution of the forged transcript is given by $1/q^3$ for each of the instance $(a,b,c,r) \in Z_q^2 \times G^2$. Now consider the probability distribution of the real transcript. Every pair of $(a,b) \in Z_q^2$ will uniquely determine a pair $(c,r) \in G$ and therefore, the probability distribution of the real transcript is given by $1/q^2$. Since the probability distribution of the forged transcript is different from the probability distribution of the real one, this concludes that the above confirmation is *not* zero-knowledge (Definition A7, appendix A). $\square$

3.1.2 Disavowal Protocol for the Undeniable Signature

The confirmation allows the verifier to validate the signature. However, the signer can easily deny a signature by returning some random response instead of raising the challenge $c = z^a h^b \pmod{p}$ to the power of x^{-1} . In this case the equality test $r \stackrel{?}{=} m^a g^b$ will be unlikely to hold. Unfortunately, the confirmation protocol does not allow the verifier to differentiate the cases: whether the signature is invalid or the signer did not respond properly.

In case the confirmation protocol fails to prove the validity of the signature the signer and the verifier can execute the following disavowal protocol, which allows the signer to interactively prove the invalidity of the signature. The verifier will be convinced that the signature is invalid if $(r_1 g^{-b_1})^{a_2} \not\equiv (r_2 g^{-b_2})^{a_1} \pmod{p}$ holds at the end of the disavowal protocol

Signer**Verifier**

$$a_1, b_1, a_2, b_2 \in_{\mathbb{R}} \mathbb{Z}_q$$

$$\xleftarrow{c_1}$$

$$c_1 = z^{a_1} h^{b_1} \pmod{p}$$

$$r_1 = c_1^{1/a_1} \pmod{p}$$

$$\xrightarrow{r_1}$$

$$\text{if } r_1 = m^{a_1} g^{b_1} \pmod{p}$$

then stop (the signature is valid)

$$\xleftarrow{c_2}$$

$$c_2 = z^{a_2} h^{b_2} \pmod{p}$$

$$r_2 = c_2^{1/a_2} \pmod{p}$$

$$\xrightarrow{r_2}$$

$$(r_1 g^{-b_1})^{a_2} \stackrel{?}{\equiv} (r_2 g^{-b_2})^{a_1} \pmod{p}$$

Figure 3.2 The Disavowal Protocol for Undeniable Signature

Correctness of the Disavowal Protocol

Notice that the disavowal protocol is actually performing the confirmation protocol twice, between the signer and the verifier. If the condition $r_1 = m^a g^{b_1} \pmod{p}$ does not hold at the end of the confirmation protocol, then the verifier should raise another confirmation request, in order to complete the disavowal protocol.

Assuming that the signer honestly replies r_1 and r_2 by raising c_1 and c_2 to the power of $1/x$ respectively, then it is easy to see that $(r_1 g^{-b_1})^{a_2} \stackrel{?}{=} (r_2 g^{-b_2})^{a_1} \pmod{p}$ holds.

Since

$$(r_1 g^{-b_1})^{a_2} = ((m^{x a_1} g^{x b_1})^{1/x} g^{-b_1})^{a_2} = m^{a_1 a_2} \pmod{p}$$

and

$$(r_2 g^{-b_2})^{a_1} = ((m^{x a_2} g^{x b_2})^{1/x} g^{-b_2})^{a_1} = m^{a_1 a_2} \pmod{p}$$

Thus,
$$(r_1 g^{-b_1})^{a_2} = (r_2 g^{-b_2})^{a_1} \pmod{p}$$

Once the signer returns r_1 to the verifier, the protocol will only be continued if $r_1 \neq m^{a_1} g^{b_1} \pmod{p}$, which means either the signature is invalid or the signer did not properly respond to the challenge c_1 . If the condition $(r_1 g^{-b_1})^{a_2} \stackrel{?}{=} (r_2 g^{-b_2})^{a_1} \pmod{p}$ holds, then (i) the signer fails to prove the validity of the signature, and (ii) the signer has responded honestly to both the challenges c_1 and c_2 . These two facts imply that the signer did properly respond to the signer; however, it fails to prove to the verifier that the

signature is valid. This implies that the signature is invalid if the condition $(r_1 g^{-b_1})^{a_2} = (r_2 g^{-b_2})^{a_1} \pmod{p}$ holds at the end of the disavowal protocol.

Now consider the case that $(r_1 g^{-b_1})^{a_2} \neq (r_2 g^{-b_2})^{a_1} \pmod{p}$ does not hold at the end of the disavowal protocol. This implies that (i) the signer fails to prove the validity of the signature, and (ii) the signer did not respond properly to both challenges c_1 and c_2 . In this case, the verifier will be convinced that the signer did not respond properly to the challenges and is trying to deny a valid signature. Further, this implies that the signer wishes to deny the signature (valid or invalid), if $(r_1 g^{-b_1})^{a_2} = (r_2 g^{-b_2})^{a_1} \pmod{p}$ does not hold at the end of the disavowal protocol.

Security of the Disavowal Protocol

Recall that the necessary condition for the signer to deny a valid signature is by returning r_1 and r_2 in such a way that C3.1.1 holds.

$$\text{C3.1.1} \quad (r_1 g^{-b_1})^{a_2} = (r_2 g^{-b_2})^{a_1} \pmod{p}$$

Assuming that the verifier initializes the challenge, $c_1 = z^{a_1} h^{b_1} \pmod{p}$ and $c_2 = z^{a_2} h^{b_2} \pmod{p}$, the signer then computes $r_1 = c_1^{x_1^{-1}}$ and $r_2 = c_2^{x_2^{-1}}$, such that

$(x_1, x_2) \neq (x, x)^{**}$. Now the signer's task is to reply (r_1, r_2) , in such a way that C3.1.1 holds, if the signer wants to deny the signature. Therefore, we define the following:

Definition 3.5. In the disavowal protocol, as stated in figure 3.2, the signer can *deny a valid signature* if and only if the signer can reply r_1 and r_2 to the verifier in such a way that $(r_1 g^{-b_1})^{a_2} = (r_2 g^{-b_2})^{a_1} \pmod{p}$, or equivalently $r_1 = (r_2 g^{-b_2})^{a_1 a_2^{-1}} g^{b_1} \pmod{p}$.

Theorem 3.3: In the disavowal protocol as stated in figure 3.2, the probability that a signer can deny a valid signature (definition 3.5) is given by $1/q$.

Proof: The signer must return $r_1 \in G$ and $r_2 \in G$ in such a way that $r_1 = (r_2 g^{-b_2})^{a_1 a_2^{-1}} g^{b_1} \pmod{p}$. However, with a_1, a_2, b_1 and b_2 known, the signer can only guess the correct value of r_1 with probability $1/q$. This concludes the proof that the probability that a signer can deny a valid signature is given by $1/q$. $\square$

Theorem 3.4. The disavowal protocol discussed in section 3.1.2 is *not* zero-knowledge (See Appendix A).

^{**} if $(x_1, x_2) \neq (x, x)$, then the signer is honest, which contradicts with our assumption

Proof. Again, the disavowal protocol is actually executing the confirmation protocol, twice. We have already prove that the confirmation protocol is *not* zero-knowledge. Thus this implies the disavowal protocol must not be zero-knowledge. $\square$

3.1.3 Creating the Fake Transcript.

Assume Veronica is convinced of the signature validation. Can Veronica show the transcript to another verifier, Vicky, and try to convince her that the signature is correct? The answer is absolutely no, because Vicky should not trust any transcript unless she is actually involved in the verification protocol. This is because Veronica (or anybody else) can easily create a fake transcript. The following section demonstrates how an adversary, Alice, can create a fake transcript.

Clearly, any transcript from the confirmation protocol takes the following form.

Signer

Verifier

$$a, b \in_R \mathbb{Z}_q \quad (1)$$

???

$\xleftarrow{c}$

$$c = z^a h^b \pmod{p} \quad (2)$$

???

$\xrightarrow{r}$

$$r = m^a g^b \pmod{p} \quad (3)$$

Figure 3.3. The Fake Transcript of the Undeniable Signature

Now, Alice can randomly select (a,b) and compute the “correct” response, r , as well as the challenge, c . Since x is known only to the signer, Vicky has no way of determining whether r is computed by raising the challenge to the power of $1/x$ or directly computed from $m^a g^b \pmod{p}$. Therefore, the above transcript seems to be “correct” in Vicky’s view.

3.1.4 Weakness of Undeniable Signature ?

Shortly after David Chaum introduced the idea of “undeniable signature” [CA89], Y.Desmedt and M.Yung pointed out some “weaknesses” of the undeniable signature [DY91]. They claimed that in any undeniable signature scheme, the signer has absolutely no control over the number of verifiers being convinced by the confirmation protocol. More precisely, when the signer believes that he is verifying a signature with a single user, he may, in fact, be verifying the signature for a large group of people. Desmedt and Yung [DY91] proposed the following attack plan.

Assume there is a group of verifiers $\{v_1, \dots, v_n\}$ where each of them wants to verify a signature z (corresponding to a message m). Note that those verifiers need not trust each other, but they all trust a single party, v , which will execute the confirmation protocol with the signer. Now all the verifiers jointly compute two random numbers (a,b) . For example, the random number a, b can be calculated in the following way: suppose each of the verifiers v_i randomly choose $(a_i, b_i) \in \mathbb{Z}_q^2$, and compute $a = \sum_{i=1}^n a_i \pmod{q}$, $b = \sum_{i=1}^n b_i \pmod{q}$.

Verifier v now can execute the confirmation protocol with the signer, using $c=z^a h^b \pmod{p}$ as the initial challenge. Once the signature has been successfully validated, verifier v will privately send the transcript to all verifiers $\{v_1, \dots, v_n\}$. Therefore, all of the verifiers $\{v_1, \dots, v_n\}$ will be convinced of the signature's validity, since the challenge was jointly computed by them.

Shortly after [DY91] pointed their result, [Cha91] claimed that this kind of attack is irrelevant to the undeniable signature. The major reason is that the verifier v can always create a fake transcript to convince all the verifiers $v_1, \dots, v_n$ to accept the signature, even though he has no control of the selection of (a, b) . To illustrate this, first assume $v_1, \dots, v_n$ jointly computed a and b . Thus all of the verifiers $\{v_1, \dots, v_n\}$ are expecting v to confirm the signature by the challenge $c=z^a h^b \pmod{p}$. However, it is easy for the verifier v to create the following transcript by computing $r=m^a g^b \pmod{p}$.

In conclusion, the attack proposed by [DY91] is not applicable to the undeniable signature, since it is easy for the verifier v to create a fake transcript.

3.2 Zero-Knowledge Undeniable Signature

The idea of undeniable signature proposed by [CA89] is particularly suitable for a wide range of applications, as we stated in the last section. David Chaum later proposed the zero-knowledge undeniable signature [Cha90], which is the first undeniable signature scheme based on the zero-knowledge interactive proof system. The signature scheme consists of three sub-protocols: signing, confirmation and disavowal protocol. In addition, the cryptographic setting is identical to that of the undeniable signature described in Section 3.1. We briefly recall the model setting: let G be a multiplicative Group of prime order q , such that g is a primitive element in G .

Signer's Private Key: $x \in \mathbb{Z}_q$

Signer's Public Key: $h = g^x \pmod{p} \in G$

Signing Protocol:

To sign a message $m \in G$, the signer computes the signature $z \in G$ as follows: $z = m^x \pmod{p}$

3.2.1 Verification Protocol for the Zero-Knowledge Undeniable Signature

The verifier can confirm a signature by executing the confirmation protocol with the signer.

Signer**Verifier**

$$a, b \in_R Z_q$$

$$u \in_R Z_q$$

$$\xleftarrow{c}$$

$$c = m^a g^b \pmod{p}$$

$$r_1 = cg^u \pmod{p}$$

$$r_2 = r_1^x \pmod{p}$$

$$\xrightarrow{r_1, r_2}$$

$$c \stackrel{?}{=} m^a g^b \pmod{p}$$

$$\xleftarrow{a, b}$$

$$\xrightarrow{u}$$

$$r_1 \stackrel{?}{=} cg^u \pmod{p} \quad (1)$$

$$r_2 \stackrel{?}{=} z^a h^{b+u} \pmod{p} \quad (2)$$

Figure 3.4. Confirmation Protocol for the Zero-Knowledge Undeniable Signature

The verifier accepts the signature as genuine if and only if conditions (1) and (2) hold at the end of the confirmation protocol. The major difference between the zero-knowledge undeniable and the undeniable signature is that the oracle for computing $c^{1/x} \pmod{p}$ is no longer available to the verifier.

Correctness for the Confirmation Protocol

Assuming that both the signer and the verifier honestly follow the confirmation protocol, then it is trivial to see $r_1 = cg^u \pmod{p}$. Additionally,

$$r_2 = r_1^x \pmod{p}$$

$$= (m^a g^b g^u)^x \pmod{p}$$

$$= (m^x)^a (g^x)^{b+u} \pmod{p}.$$

Recall that $z = m^x \pmod{p}$ and $h = g^x \pmod{p}$, so we have

$$r_2 = z^a h^{b+u} \pmod{p}.$$

Security of the Confirmation Protocol

Definition 3.6. Function $\text{Confirm}(r_1, r_2, c, z, a, b, u)$ will return 1 if and only if all of the following conditions hold, otherwise $\text{Confirm}(r_1, r_2, c, z, a, b, u) = 0$.

$$\text{C3.2.1} \quad r_1 = c g^u \pmod{p}$$

$$\text{C3.2.2} \quad r_2 = z^a h^{b+u} \pmod{p}$$

Definition 3.7. At the end of the confirmation protocol, the signer *will accept the signature z (corresponding to the message m) as genuine* if and only if $\text{confirm}(r_1, r_2, c, z, a, b, u) = 1$, where a, b, u, c, z, r_1 and r_2 are computed as stated in figure 3.4.

Theorem 3.5. Given an invalid signature $z \in G$ (definition 3.2), the probability of the verifier accepting z as genuine (definition 3.7) is $1/q$.

Proof: Assume that the signer wants to convince the verifier to accept the invalid signature, i.e. $z \neq m^x \pmod{p}$. Clearly, his task is to return r_2 in such a way that $r_2 = z^a h^{b \cdot u} \pmod{p}$, with a and b unknown. However, this is infeasible for him to solve (a, b) from the challenge $c = m^a g^b \pmod{p}$, thus the signer has to guess the correct value of $r_2 \in G$ with probability $1/q$. This implies the probability that the signer can convince the verifier to accept an invalid signature as given by $1/q$. $\square$

Theorem 3.6. The confirmation protocol as stated in figure 3.4 is zero-knowledge.

Proof. It is sufficient to show that the probability distribution of the real transcript is identical to the probability distribution of the forged transcript (Appendix A, Definition A7). Assume that $z \neq m^x \pmod{p}$ and (a, b, u) are uniformly chosen over Z_q^3 . First, in considering a real transcript, it is easy to see that (a, b, u) uniquely determines (c, r_1, r_2) . This implies that all the tuples (a, b, u, c, r_1, r_2) have an equal probability of $1/q^3$. Now we consider the probability distribution of the fake transcript, assuming that a forger first selects (a, b, u) from Z_q^3 and computes $c = m^a g^b \pmod{p}$, $r_1 = g^u \pmod{p}$ and $r_2 = z^a h^{b \cdot u} \pmod{p}$. In this case, the fake transcript will be indistinguishable from the real transcript. Notice that the

condition $z = m^x \pmod{p}$ does not necessary hold in the fake transcript. Additionally, it is easy to see that all the fake transcripts are equally probable of $1/q^3$. Therefore the probability distribution of the fake transcript is identical to the probability distribution of the real transcript. Therefore the confirmation protocol mentioned in figure 3.4 is zero-knowledge. $\square$

3.2.2 Disavowal Protocol for the Zero-Knowledge Undeniable Signature

The confirmation protocol allows the verifier to confirm the signature with the signer. However, in case condition C3.2.1 or C3.2.2 did not hold, it is implied that either the signature is invalid or the signer did not response properly to the challenge. In this case, the verifier is able to differentiate between the cases by executing the disavowal protocol.

In the following disavowal protocol, we assume $z = m^x$ and that z is a valid signature if and only if $x' = x$. Let $k < p$ be a small integer, such that both the signer and the verifier are agreed. The following protocol allows the signer to deny a valid signature with probability $1/k$.

SignerVerifier

$$s \in_R \{1, \dots, k\}, a \in_R Z_q$$

$$c_1 = m^s g^a \pmod{p}$$

$$c_2 = z^s h^a \pmod{p}$$

$$r \in_R Z_q$$

$$\xleftarrow{c_1, c_2}$$

$$\xrightarrow{\text{blob}(r, s')}$$

$$c_1 \stackrel{?}{=} m^s g^a \pmod{p}$$

$$\xleftarrow{a}$$

$$c_2 \stackrel{?}{=} z^s h^a \pmod{p}$$

$$\xrightarrow{r}$$

recover s' from $\text{blob}(r, s')$

check $s \stackrel{?}{=} s'$

where **blob** is some bit commitment scheme. (See Appendix B for more information about the bit commitment scheme).

Figure 3.5 The Disavowal Protocol for the Zero-Knowledge Undeniable Signature

Correctness and Security of the Disavowal Protocol

Assume that $z = m^{x'} \pmod{p}$ for some x' in Z_q . Notice that the signature is valid if and only if $x' = x$ where x is the private key of the signer. The protocol is first initialized by generating the message $c_1 = m^s g^a \pmod{p}$ where $s \in_R \{1, \dots, k\}$ and $a \in_R Z_p$. Now the verifier determines the value of s by trial and error (that's the reason that we require k to be a "small" integer) based on the following algorithm.

$$\text{let } y_1 = c_1^x = m^{xs} g^{xa} \pmod{p}$$

$$\text{and } y_2 = y_1 / c_2 \pmod{p}$$

Therefore,

$$y_2 = m^{xs} g^{xa} / (m^x)^s (g^x)^a \pmod{p}$$

$$= (m^x / m^{x'})^s \pmod{p}$$

Notices that $x' \neq x$ if signature is invalid. Therefore $(m^x / m^{x'}) \neq 1 \pmod{p}$ and let $k_1 = (m^x / z) \pmod{p}$. Thus we have

$$y_2 = k_1^s, \text{ for some } s \in \{1, \dots, k-1\} \quad (1)$$

Now the signer can take $O(k)$ time to compute s by trial and error. Let s' be the value that the signer has recovered. Notices that if the signature z is genuine, that is $z = m^x$,

then we have $y_2 = 1^s = 1$. In this case all of the $s' \in \{1, \dots, k-1\}$ are equally probable. This implies the signer has to guess the correct value of s' with probability $1/k$.

The above algorithm allows the signer to find s' by trial and error, provided that the signer possesses the private key x and the signature is invalid. The signer then commits a random values r and s' to the verifier by some zero-knowledge bit commitment scheme. Once the signer reveals the random value x , the verifier is able to determine s' from the commitment, $\text{blob}(r, s')$.

The above protocol is repeated n times to give the signer a probability of $1/k^n$ to deny a valid signature.

Theorem 3.7. The disavowal protocol stated in figure 3.5 is zero-knowledge.

Proof. Consider the following procedure to create a forged transcript: the forger first selects $s \in_R \{1, \dots, k\}$ and $(a, r) \in \mathbb{Z}_q^2$. Let c_1, c_2 be $m^s g^s \pmod{p}$ and $z^s h^s \pmod{p}$ respectively. Additionally, by letting $s' = s$ we can create the following indistinguishable transcript:

Signer**Verifier**

$$s \in_R \{1, \dots, k\}, a \in_R Z_q$$

$$c_1 = m^s g^a \pmod{p}$$

$$\xleftarrow{c_1, c_2}$$

$$c_2 = z^s h^a \pmod{p}$$

$$\xrightarrow{\text{blob}(r, s')}$$

$$\xleftarrow{a}$$

$$\xrightarrow{r}$$

recover s' from $\text{blob}(r, s')$

check $s \stackrel{?}{=} s'$

In the above transcript, we will, of course, recover $s'=s$, and the resulting transcript is indistinguishable from the real one. It is easy to see that the probability distribution for the real transcript is give by $1/(kq)$, and that it is identical to the probability distribution of the forged transcript. This concludes that the above disavowal protocol is zero-knowledge. $\square$

4. Designated Confirmer Signature

In this chapter, we will discuss some drawbacks of the undeniable signature [BCD90] [Cha90] [CA89]. In section 4.1, we will present an enhancement of the undeniable signature known as the designated confirmer signature in which a party known as the signer is responsible for signing and another party known as the confirmer is responsible for signature confirmation. The above scheme is known as the $(1,1)$ threshold designated confirmer signature. In section 4.2, we will generalize the idea of $(1,k)$ threshold designated confirmer signature in which the signature confirmation can be done by any one of k confirmers. Lastly, we will propose the $(1,1)^*$ threshold designated confirmer signature in section 4.3, in which the signature can either be confirmed with the cooperation of the signer or the confirmer. The $(1,k)^*$ threshold designated confirmer signature is a direct result of our $(1,1)^*$ threshold designated confirmer signature.

The undeniable signature [BCD90] [Cha90] [CA89] allows the signer to create a signature in such a way that cooperation of the originator is necessary for signature confirmation. Advantages of the undeniable signature over the conventional digital signature have already been discussed in chapter 3. However, with the undeniable signature another problem arises: If the signer is not available, or refuses to cooperate with the verifier, then the signature itself is totally useless to the verifier. A typical example of this when a software vendor provides an undeniable signature which allows all the legitimate users to validate the software authenticity. If the company goes bankrupt later, then none of the legitimate users can validate the signature anymore.

To overcome this problem, David Chaum recently proposed the idea of a designated confirmer signature [Cha94], which separates the responsibility of signing and verification into two distinct parties - signer Simon and confirmer Colin. Similar to the undeniable signature, the verification protocol can not be completed without the cooperation of Colin. A major advantage of the designated confirmer signature over the undeniable signature is that two distinct parties are responsible for the signing and verification process. This greatly enhances system security.

We define the **original receiver** as 'the one whom the signer intentionally sent the signature to', and those who receive the signature from the original receiver, directly or otherwise, are referred to as **indirect receivers**. For example, assume that Simon sent a signature to Rita and redirected the signature to Veronica. We refer to Simon, Rita and Veronica as the signer, original receiver and indirect receiver respectively. It should be noted that the receiver can also assume the role of the verifier in some cases. Additionally we might have numerous indirect receivers but no original receiver. A typical example involves the software vendor we mentioned before. The vendors publish the signature without any original receiver and all the paid users act as the indirect receivers.

Figure 4.1 Designated Confirmer Signature

(t,k) and (t,k)* Threshold Designated Confirmer Signature

Before discussing the designated confirmer signature proposed by [Cha94] we should define the (t,n) and (t,n)* threshold designated confirmer signature as follows:

Definition 4.1. *(t,k) Threshold Designated Confirmer Signature:* assume $t, k \in \mathbb{Z}$ such that $t \leq k$. A (t,k) threshold designated confirmer signature, or (t,k)-signature for short, is a signature scheme which involves a signer and a set of k confirmers $C = \{C_1, \dots, C_k\}$. The (t,k) threshold designated confirmer signature allows the signer to create a signature in such a way that it can be confirmed by any subset of the confirmers $C' \subseteq C$, provided that $|C'| \geq t$. We emphasize the fact that in the (t,k)-signature scheme, there is no way for any indirect receiver to confirm the signature with the signer.

Definition 4.2. *(t,k)* threshold designated confirmer signature.* Also referred to as (t,k)*-signature, this is a (t,k)-signature such that the verifier is able to confirm the signature with the signer. Advantages of the (t,k)*-signature over the (t,k)-signature will be discussed in chapter 4.3.

The designated confirmer signature proposed by [Cha94] is essentially a $(1,1)$ threshold designated confirmer signature. This means there exists a confirmer, Colin, and any indirect verifier can only confirm the signature with his cooperation. The $(1,1)$ signature scheme will first be discussed in chapter 4.1 and we will propose the $(1,k)$ -signature and $(1,1)^*$ -signature in chapter 4.2 and 4.3 respectively. The $(1,k)^*$ -signature scheme follows as a direct result of the $(1,1)^*$ signature scheme.

4.1 $(1,1)$ Threshold Confirmer Designated Signature

The basic model of the proposed designated confirmer signature [Cha94] consists of two protocols: the signing and confirmation protocol.

The **signing protocol** allows the signer, Simon, to create a signature and distribute it to the original receiver Rita. Once the protocol is completed, the original receiver will be convinced that the signature from the signer is genuine. Additionally, the original receiver has to carry out the confirmation protocol if he wants to reconfirm the signature, after the completion of the signing protocol.

The **confirmation** protocol allows the indirect receiver, Veronica, and/or the original receiver to confirm the signature with the designated confirmer, Colin. The difficult part of the designated confirmer signature is that the signer has to create a signature which is somehow 'related' with Colin's private key. The resulting signature

sounds like it was originally created by Colin, thus allowing Colin to confirm the signature with the verifier. Note that the verifier may be the original receiver or any indirect receiver.

System Model

The (1,1)-signature scheme requires that both signer and confirmer possess a private-public key pair. The private key (k_s) and public key (n, k_v) for the signer is an instance of the RSA [RSA78] signature scheme previously mentioned in chapter 2.2.

The method of selection for the private and public key for Colin is as follows: Let k and q be distinct primes such that $p = kq+1 < n$ is also prime. We define the multiplicative group $G \subset Z_p^*$ as the cyclic group $\langle g \rangle$, where $g^k \neq 1 \pmod{p}$ and g is an element of order q . The private and public keys of the confirmer are then given by $z \in Z_q$ and $h = g^z \pmod{p}$ respectively. The private-public key of both the signer and confirmer are summarized as follows:

	Private Key	Public Key
Signer	$k_s \in Z_{\Phi(n)}^*$	$n \in Z$ and $k_v \in Z_{\Phi(n)}^*$
Confirmer	$z \in Z_q$	$h = g^z \pmod{p} \in G$

Additionally, we assume the following functions are publicly available to all of the parties involved.

Definition 4.3. Invertible function $H:Z_n \times Z_n \rightarrow Z_n$ is a surjective and invertible function which maps $x \in Z_n \times Z_n$ to Z_n . Define $C_0, \dots, C_{n-1}$ as the partition of $Z_n \times Z_n$ such that $C_i = \{x \in Z_n \times Z_n \mid H(x) = i\}$. When we refer to H as invertible, we mean there exists a probabilistic algorithm $A:Z_n \times \{0,1\}^* \rightarrow G \times G$ which on inputs $i \in Z_n$ and some random value $x \in \{0,1\}^*$, returns $a \in C_i$. Notice that the actual element a being returned is dependent on random input, but it is guaranteed that $a \in Z_n \times Z_n$. For example, we can define the function $H:Z_n \times Z_n \rightarrow Z_n$ as $H(a,b) = a+b \pmod{p}$ and $A:Z_n \times \{0,1\}^* \rightarrow Z_n \times Z_n$ as follows:

Input $c \in Z_n$ and $r \in \{0,1\}^*$

let $a = 1^r \pmod{p}$ (the unary representation of r)

let $b = c - a \pmod{p}$

return (a,b)

Definition 4.4. 1-Way Hash Function $F:\{0,1\}^* \rightarrow Z_n$ is a 1-way hash function which maps an arbitrary binary string to Z_n . Informally, a hash function h is one-way if, given a message digest z , it is computationally infeasible to find a message x such

that $h(x)=z$. Though there are many functions which are believed to be 1-way hash functions, the existence of the 1-way hash function is an open problem in modern cryptography.

The following notations will be used in the following sections:

Signer's public key: (k_v, n) , where $k_v \in Z_{\Phi(n)}^*$, $n \in Z$

Signer's private key: $k_s \in Z_{\Phi(n)}^*$, such that $k_v \cdot k_s = 1 \pmod{\Phi(n)}$

Confirmer's private key: $z \in Z_q$.

Confirmer's public key: $h = g^z \pmod{p} \in G$

Additionally, we assume H (Definition 4.3), F (Definition 4.4), g, p, q, n are public knowledge.

4.1.1 Signing Protocol for the (1,1) Designated Confirmer Signature

We will discuss the signing protocol for the (1,1) designated confirmer signature in this section. The signing protocol allows the signer, who possesses a private key k_s , to create a signature corresponding to a message m . The signature depends on the message

m , the signer's private key k_s , and the confirmer's public key $h=g^z \pmod{p}$. The resulting signature is verifiable by the confirmer upon request.

The signing protocol for the designated confirmer signature is as shown in figure 4.2. The protocol can be separated into two parts. In the first part, the signer creates the signature $(a,b,\alpha) \in G^2 \times Z_n$, corresponding to the message $m \in \{0,1\}^*$. This part is completed by the signer sending the signature (a,b,α) to the original receiver (in case there is no original receiver for the signature, the signer simply publishes the signature and terminates the protocol). The second part of the signing protocol is initiated by the original receiver who sends message (2) to the signer, and is completed by the original receiver who receives message (5) from the signer. The second part of the signing protocol allows the signer to interactively prove to the original receiver that the signature is valid. It can be bypassed if the original receiver does not wish to confirm the signature with the signer. It is easy to see that the second part of the signing protocol is applicable only if the signer can remember the random value x . Under the assumption that the signer does not keep track of the random value being used, he has no way to prove the signature's validity to any verifier once the signing protocol is completed. In section 4.3, we will propose a technique, the $(1,1)^*$ designated confirmer signature, which allows the signer to recall the random value being used for each particular message, without the aid of a database.

SignerOriginal Receiver

$$x \in_R \mathbb{Z}_q$$

$$a = g^x \pmod{p}$$

$$b = h^x \pmod{p}$$

$$\alpha = (H(a, b) \oplus F(m))^{k_1} \pmod{n}$$

part I

$$\xrightarrow{a, b, \alpha} s, t \in_R \mathbb{Z}_q \quad (1)$$

$$r \in_R \mathbb{Z}_q \quad \xleftarrow{c} \quad c = g^s h^t \pmod{p} \quad (2)$$

$$d = g^r \pmod{p}$$

$$e = (cd)^x \pmod{p} \quad \xrightarrow{d, e} \quad (3)$$

$$g^s h^t \stackrel{?}{=} c \pmod{p} \quad \xleftarrow{s, t} \quad (4)$$

part II

$$\xrightarrow{r} \quad d \stackrel{?}{=} g^r \pmod{p} \quad (5)$$

$$e/a^r \stackrel{?}{=} a^s b^t \pmod{p}$$

$$\alpha^{k_1} \stackrel{?}{=} H(a, b) \oplus F(m) \pmod{n}$$

Figure 4.2 Signing Protocol for the (1,1) Designated Confirmer Signature.

Before we analyze the signing protocol, we first state the necessary conditions for the valid signature in the (1,1) designated confirmer signature scheme.

Definition 4.5 Function $Validate_{(1,1)}(m,a,b,\alpha)$ will return 1 if and only if all of the following conditions hold, otherwise, $Validate_{(1,1)}(m,a,b,\alpha)=0$.

$$\text{C4.1.1} \quad b=a^2 \pmod{p}$$

$$\text{C4.1.2} \quad \alpha^{k_v} = H(a,b) \oplus F(m) \pmod{n}.$$

Definition 4.6. The signature $(a,b,\alpha) \in G \times G \times Z_q$ (corresponding to the message $m \in G$) is *valid* if and only if $Validate_{(1,1)}(m,a,b,\alpha) = 1$.

Correctness of the Signing Protocol

Assuming that all the parties are honestly following the signing protocol, then it is easy to see that $a = g^x \pmod{p}$ and $b = h^x \pmod{p}$. This implies that condition C4.1.1 holds, since $b = h^x = g^{zx} = a^z \pmod{p}$.

In addition, condition C4.1.2 also holds, since $\alpha = (H(a,b) \oplus F(m))^{k_v} \pmod{n}$. Therefore $\alpha^{k_v} = (H(a,b) \oplus F(m))^{k_v k_v} \pmod{n}$. Recall that (k_s, k_v) is an instance of the RSA signature key. Therefore, by theorem 2.5 we have $\alpha^{k_v} = H(a,b) \oplus F(m) \pmod{n}$. This implies that C4.1.2 holds.

Both the conditions C4.1.1 and C4.1.2 hold in the signing protocol, by definition 4.5, which implies that $\text{Validate}_{(1,1)}(m,a,b,\alpha) = 1$. Therefore we conclude that the original receiver will accept the signature (a,b,α) as genuine (definition 4.6).

Lastly, we have to prove that the conditions (i) $d = g^r \pmod{p}$, and (ii) $e/a^r = a^t b^t \pmod{p}$ hold. It is easy to see that $d = g^r \pmod{p}$ holds if both the signer and original receiver are honestly following the protocol. In addition, the condition $e/a^r = a^t b^t \pmod{p}$ also holds, since

$$\begin{aligned}
 e/a^r \pmod{p} &= (cd)^x/a^r \pmod{p} \\
 &= (g^t h^t g^r)^x/a^r \pmod{p} \\
 &= g^{x(s+r)} h^{xt}/a^r \pmod{p} \\
 &= a^{s+r} b^t/a^r \pmod{p} \\
 &= a^t b^t \pmod{p}.
 \end{aligned}$$

Security of the Signing Protocol

In this chapter, we discuss the security of the signing protocol in the (1,1) designated confirmer protocol. In the following sections, we assume that there exists a original receiver who wishes to confirm the signature's validity with the signer. That means the second part of the signing protocol has been completed by the signer and the verifier. Before the discussion of our result, we have to define the following.

Definition 4.7. Function $Sign_{(1,1)}(a,b,d,e,r,s,t,m,\alpha)$ will return 1 if and only if all of the conditions C4.1.3, C4.1.4 and C4.1.5 hold, otherwise $Sign_{(1,1)}(a,b,d,e,r,s,t,m,\alpha)=0$.

$$\text{C4.1.3} \quad d=g^r \pmod{p}$$

$$\text{C4.1.4} \quad e/a' = a^s b^t \pmod{p}$$

$$\text{C4.1.5} \quad \alpha^{k_v} = H(a,b) \oplus F(m) \pmod{n}$$

Definition 4.8. The original receiver will accept the signature $(a,b,\alpha) \in G \times G \times Z_n$ (corresponding to message $m \in G$) as *genuine* if and only if $Sign_{(1,1)}(a,b,d,e,r,s,t,m,\alpha) = 1$, where $(d,e,r) \in G \times G \times Z_q$ are responses from the signer and $(s,t) \in Z_q \times Z_q$ are random values chosen by the original receiver.

Theorem 4.1. Given that an invalid signature $(a,b,\alpha) \in G \times G \times Z_n$ corresponds to message $m \in G$ (definition 4.6), then the original receiver will accept (a,b,α) as genuine with probability $1/q$ (definition 4.8).

Proof. By definition 4.8, the signer must respond to the original receiver with $(d,e,r) \in G \times G \times Z_q$ such that $\text{Sign}_{(1,1)}(a,b,d,e,r,s,t,m,\alpha) = 1$. Recall that condition C4.1.4 must be fulfilled if $\text{Sign}_{(1,1)}(a,b,d,e,r,s,t,m,\alpha) = 1$. This means that the original receiver must receive $e \in G$ from the signer such that condition C.4.1.4 holds, where

$$\begin{aligned} e &= a^s b^t a^r \pmod{p} \\ &= a^{s+yt+r} \pmod{p} \text{ with unknown } s \text{ and } t. \end{aligned}$$

This implies that the signer has to guess the correct value of $s+yt+r \pmod{q}$ with a probability of $1/q$. This implies that the signer can convince the original receiver to accept the signature (a,b,α) as genuine with probability $1/q$. $\square$

4.1.2 Confirmation Protocol for the (1,1) Designated Confirmer Signature

The confirmation protocol allows the confirmer to convince the original receiver or all other indirect receivers that the signature $(a,b) \in G^2 \times Z^n$, which was created by the signer, is valid. In the following protocol, we denote the verifier as either the original or indirect receiver.

ConfirmerVerifier

$$u, v \in Z_q$$

$$t \in_R Z_q$$

$$\xleftarrow{k}$$

$$k = g^u a^v \pmod{p}$$

$$y = g^t \pmod{p}$$

$$s = (ky)^z \pmod{p}$$

$$\xrightarrow{y, s}$$

$$g^u a^v \stackrel{?}{=} k \pmod{p}$$

$$\xleftarrow{u, v}$$

$$\xrightarrow{t}$$

$$g^t \stackrel{?}{=} y \pmod{p}$$

$$s/h^t \stackrel{?}{=} h^u b^v \pmod{p}$$

$$\alpha^{h_v} \stackrel{?}{=} (H(a, b) \oplus F(m)) \pmod{n}$$

Figure 4.3 Confirmation Protocol for the (1,1) Designated Confirmer Signature

Correctness of the Confirmation Protocol

If both the confirmer and verifier honestly follow the confirmation protocol, it is easy to see that $g^t = y \pmod{p}$. In addition, consider

$$\begin{aligned}
s/h^t &= (g^u a^v g^t)^z / h^t \pmod{p} \\
&= g^{z(u+t)} a^{zv} / h^t \pmod{p} \\
&= h^{u+t} b^v / h^t \pmod{p} && (\text{since } b=a^z \pmod{p}) \\
&= h^u b^v \pmod{p}
\end{aligned}$$

This implies that with high probability, $b=a^z \pmod{p}$, and therefore condition C4.1.1 holds. In addition, we have already proven that $\alpha^{k_v} = H(a, b) \oplus F(m) \pmod{n}$ in section 4.1.1, which implies that condition C4.1.2 holds. Consequently, we conclude that $\text{Validate}_{(1,1)}(m, a, b, \alpha) = 1$ and therefore, the verifier will accept the signature as genuine (definition 4.8).

Security of the Confirmation Protocol

In this chapter, we discuss the security of the confirmation protocol in the (1,1) designated confirmer protocol.

Definition 4.9. Function $\text{Confirm}_{(1,1)}(a, b, s, y, u, v, t, m, \alpha)$ will return 1 if and only if all of the conditions C4.1.6, C4.1.7 and C4.1.8 hold, otherwise $\text{Confirm}_{(1,1)}(a, b, s, y, u, v, t, m, \alpha) = 0$.

$$\text{C4.1.6} \quad g^t = y \pmod{p}$$

$$\text{C4.1.7} \quad s/h^t = h^u b^v \pmod{p}$$

$$\text{C4.1.8} \quad \alpha^{h^t} \stackrel{?}{=} (H(a, b) \oplus F(m)) \pmod{n}$$

Definition 4.10. The verifier will accept the signature $(a, b, \alpha) \in G \times G \times Z_n$ (corresponding to message $m \in G$), as *genuine* if and only if $\text{Confirm}_{(1,1)}(a, b, s, y, u, v, t, m, \alpha) = 1$, where $(y, s, t) \in G \times G \times Z_q$ are responses from the signer and $(u, v) \in Z_q \times Z_q$ are random values chosen by the verifier.

Theorem 4.2. Given an invalid signature (a, b, α) corresponding to message $m \in G$ (definition 4.6), then the verifier will accept (a, b, α) as genuine with probability $1/q$. (definition 4.10)

Proof. By definition 4.10, the signer must respond to the original receiver with $(y, s, t) \in G \times G \times Z_q$ such that $\text{Confirm}_{(1,1)}(a, b, s, y, u, v, t, m, \alpha) = 1$. Recall that condition C4.1.7 must be fulfilled if $\text{Confirm}_{(1,1)}(a, b, s, y, u, v, t, m, \alpha) = 1$. This means the verifier have must received $s \in G$ from the signer such that condition C.4.1.7 holds, where

$$s = h^u b^v h^t \pmod{p}$$

$$= g^w \pmod{p} \quad \text{for some } w \in \mathbb{Z}_q \text{ and unknown } (u, v)$$

This implies that the signer has to guess the correct value of $w \pmod{q}$ with a probability of $1/q$. This implies that the signer can convince the receiver to accept the signature (a, b, α) as genuine with probability $1/q$. $\square$

4.1.3 Unforgability for the (1,1) Designated Confirmer Signature

First, observe that if the signature space is defined by $G \times G \times \mathbb{Z}_n$, forging the signature means Alice must find an ordered tuple $(a, b, \alpha) \in G \times G \times \mathbb{Z}_n$ such that $\text{Validate}_{(1,1)}(m, a, b, \alpha) = 1$, for some $m \in G$.

Assume that Alice has k pairs of message-signature tuples $\{(m_1, a_1, b_1, \alpha_1), \dots, (m_k, a_k, b_k, \alpha_k)\}$, and she wants to find a signature (a, b, α) for message m where $m \notin \{m_1, \dots, m_k\}$. If Alice computes $a = g^x \pmod{p}$ and $b = h^x \pmod{p}$ for some random $x \in \mathbb{Z}_q$, she must find $\alpha = (H(a, b) \oplus F(m))^{h^{-1}} \pmod{n}$, which means Alice has to break the RSA [RSA79] signature scheme in order to find k . This solution is impractical since there is no known efficient algorithm for breaking the RSA signature scheme [RSA79].

On the other hand, if she starts from any random α , then we have

$$\alpha^{h^{-1}} = H(a, b) \oplus F(m) \pmod{n}$$

Alice can assign $a = \prod_{i \in S} a_i \pmod{p}$ and $b = \prod_{i \in S} b_i \pmod{p}$ for some non-empty subset $S \subseteq \{1, \dots, k\}$. Hence, $\text{Validate}_{(1,1)}(m, a, b, \alpha) = 1$, for some random $m \in G$. This kind of attack is known as the existential forgery under an adaptive chosen plaintext attack [GMR88]. Although this attack allows Alice to find a signature (a, b, α) for some message m , this does not cause any serious problem in the $(1,1)$ signature scheme for two major reasons. First, it was hard for Alice to invert the 1-way hash function in order to recover message m . Secondly, the message corresponding to the signature is random message rather than a message chosen by Alice.

Similarly, if Alice first selects α and message m , then she has to find (a, b) such that $H(a, b) = K$ where $K = \alpha^{k_1} \oplus F(m)$. Recall that the function H is easily invertible and assumes $S = \{(a_1, b_1), \dots, (a_i, b_i)\}$ to be the largest subset of $G \times G$ such that $\forall (a_i, b_i) \in S$, $H(a_i, b_i) = K$. However, there is no way for Alice to verify which pair (a_i, b_i) condition C4.1.1 holds unless she knows the confirmer's private key z . Thus she would have to randomly choose one of the (a_i, b_i) pairs and hope that C4.1.1 holds; however, it is highly unlikely that C4.1.1 would hold for that pair (a_i, b_i) . Formally, the probability for Alice to forge the signature, if she first selects α and message m , is given by $\Pr[b = a^z \pmod{p} | H(a, b) = K]$. It is easy to see that this probability depends on the selection of the invertible function H .

Consider an example $H(a, b) = ab \pmod{p}$. Since a and b are elements in G , let $a = g^u \pmod{p}$ and $b = g^v \pmod{p}$ for some $u, v \in \mathbb{Z}_q$. Thus C4.1.1 holds with probability

$$\Pr[b=a^z \mid ab=K \pmod{p}] \quad , \text{where } K=\alpha^{k_v} \oplus F(m).$$

First, it is easy to see that $\Pr[b=a^z \mid ab=K \pmod{p}] = 0$, if $K \notin G$. And secondly, if $K \in G$ then it induces $K=g^t \pmod{p}$ for some $t \in \mathbb{Z}_q$. Therefore, the probability that Alice can successfully forge the signature is computed as follows:

$$\begin{aligned} & \frac{q}{n} \Pr[b = a^z \pmod{p} \mid ab = K \pmod{p}] \\ &= \frac{q}{n} \Pr[v = uz \pmod{q} \mid u + v = t \pmod{q}] \\ &= \frac{q}{n} \cdot \frac{1/q^2}{1/q} \\ &= 1/n \end{aligned}$$

since $\Pr[v=uz \pmod{q} \text{ and } u+v=t \pmod{q}] = 1/q^2$, and $P[u+v=t \pmod{q}] = 1/q$.

This implies that the probability of Alice forging a signature is $1/q$, provided that the function H is defined as $H(a,b)=a \cdot b \pmod{p}$.

In this section's last portion, we emphasized the necessity that the function H has to be easily invertible. Assume that H is a non-invertible function, and $(a,b,\alpha) \in G \times G \times \mathbb{Z}_n$ is a valid signature of the message m . If Alice had been convinced by the verifier that the signature is valid, then Alice could convince another party, Bob, by simply giving the

transcript to Bob. Bob will trust the transcript since it was computationally infeasible for Alice to find $(a, b) = H^{-1}(a^b \oplus F(m))$. Therefore, H must be an invertible function

4.1.4 Conversion Protocol

Surprisingly, the designated confirmer signature can be converted into a conventional digital signature by the confirmer through publishing r and y as follows

$$r = a^x \pmod{p}$$

$$y = w + zF(a, r), \quad \text{where } w \in_R Z_q.$$

Now a verifier can verify the correctness of the signature without the help of the confirmer by using the validation check

$$a^y \stackrel{?}{=} r b^{F(a, r)} \pmod{p}, \text{ and}$$

$$a^b \stackrel{?}{=} (H(a, b) \oplus F(m)) \pmod{n}.$$

4.2 The (1,k) Threshold Designated Confirmer Signature

Although (1,1) threshold designated confirmer signature separated the responsibility of the signing and confirmation into two distinct parties, it is subject to the same problems as the undeniable signature. If the confirmer is not available or refuses to cooperate with the confirmation, then there is no way for the indirect receiver to confirm the signature. This situation is exactly the same problem as we stated before, with the confirmer as the uncooperative party instead of the original signer. In this section, we will

propose a new scheme for the $(1,k)$ threshold designated confirmer, which allows the verification to be performed by any one of the k designated confirmers.

We will denote the $(1,k)$ threshold designated confirmer signature scheme as the $(1,k)$ -scheme for ease of use. Before the discussion of the $(1,k)$ -scheme, we first express the idea of how to modify the existing $(1,1)$ -scheme into the $(1,k)$ -scheme.

Main Idea

In the $(1,1)$ -scheme discussed earlier, the signature is given by (a,b,α) with condition that $b=a^z \pmod{p}$, where z is the private key of the confirmer. This technique is not applicable to the $(1,k)$ -scheme since the signer will not know which confirmer will be involved in the confirmation protocol. Instead of the private key z , our $(1,k)$ -scheme requires that $b = a^{a_0} \pmod{p}$ for some random $a_0 \in \mathbb{Z}_q$. Any confirmer who wishes to confirm the signature has to secretly recover the value a_0 , in the confirmation protocol. We first discuss how the confirmer can recover the secret value a_0 based on the Shamir's secret sharing scheme[Sha79] (see appendix C for more information about Shamir's scheme and secure message broadcast).

By broadcasting $Q = \{(x_1, f(x_1)), \dots, (x_k, f(x_k))\}$, any confirmer C_i can securely recover a_0 since $F(y_i, g') \notin X$, implying that $(F(y_i, g'), f(F(y_i, g'))) \notin \{(x_1, f(x_1)), \dots, (x_k, f(x_k))\}$. Therefore, C_i can recover the function $f(\cdot)$ by Lagrange interpolation, since Q together with $(F(y_i, g'), f(F(y_i, g')))$ essentially gives C_i " $t+1$ distinct points" of the polynomial $f(\cdot)$. Hence C_i is able to recover the function $f(\cdot)$ as well as the secret $a_0 = f(0)$.

Now we are ready to discuss our $(1, k)$ -scheme. Our model is exactly the same as the $(1, 1)$ -scheme except that each of the confirmer C_i shares a key $y_i \in Z_q$ with the signer. The private and public key of confirmer C_i is given by $z_i \in Z_q$ and $h_i = g^{z_i} \pmod{p} \in G$ respectively. In addition, the invertible function H is now $H: G^3 \rightarrow Z_n$.

4.2.1 Signing Protocol for the $(1, k)$ Designated Confirmer Signature

Initialization Phase

Assume the signer is willing to sign the message $m \in G$. The signer performs the following procedure.

- (i) Select $a_0 \in_R Z_q$;
- (ii) Select $r \in_R Z_q$ and generates the function $f(x) = \sum_{i=0}^t a_i x^i \pmod{p}$;
- (iii) Selects $T = \{x_1, \dots, x_k\}$ as described before;

(iv) Let $h = g^{a_0} \pmod{p}$;

(iv) Let $a = g^r \pmod{p}$ and $b = g^{r_0} \pmod{p}$;

(v) Let $\alpha = (H(a, b, h) \oplus F(m))^{k_1} \pmod{n}$.

Signing Protocol

Signer

Original Receiver

Figure 4.4. Signing Protocol for the $(1, k)$ Designated Confirmer Signature

Notice that $h = g^{a_0} \pmod{p} = g^{f(0)} \pmod{p}$, is part of the signature, but not the public key of any confirmer.

Definition 4.11. Function $Validate_{(l,k)}(m,a,b,h,\alpha,x_1,\dots,x_k,f(x_1),\dots,f(x_k))$ will return 1 if and only if all of the following conditions hold, otherwise $Validate_{(l,k)}(m,a,b,h,\alpha,x_1,\dots,x_k,f(x_1),\dots,f(x_k))=0$.

$$C4.2.1 \quad b = a^{f(0)} \pmod{p}$$

$$C4.2.2 \quad h = g^{f(0)} \pmod{p}$$

$$C4.2.3 \quad \alpha^{k_v} = H(a,b,h) \oplus F(m) \pmod{n}.$$

Definition 4.12. The signature $(a,b,h,\alpha,x_1,\dots,x_k,f(x_1),\dots,f(x_k)) \in G^4 \times Z_p^{2k}$ (corresponding to the message $m \in G$) is *valid* if and only if $Validate_{(l,k)}(m,a,b,h,\alpha,x_1,\dots,x_k,f(x_1),\dots,f(x_k)) = 1$.

Correctness of the Signing Protocol

Assuming that all the parties are honestly following the signing protocol, then it is easy to see that $a = g^r \pmod{p}$ and $b = g^{r_0} \pmod{p}$. This implies that condition C4.2.1 holds, since $b = g^{r_0} = a^{r_0} = a^{f(0)} \pmod{p}$. In addition, condition C4.2.2 holds since it is the way that we compute h .

Furthermore, condition C4.2.3 holds because $\alpha = (H(a, b, h) \oplus F(m))^{k_s} \pmod{n}$. Therefore $\alpha^{k_v} = (H(a, b, h) \oplus F(m))^{k_s k_v} \pmod{n}$. Recall that (k_s, k_v) is an instance of the RSA signature key. Therefore, by theorem 2.5 we have $\alpha^{k_v} = H(a, b, h) \oplus F(m) \pmod{n}$. This concludes that C4.2.3 holds.

Since all of the conditions C4.2.1, C4.2.2 and C4.2.3 hold in the confirmation protocol, by definition 4.11, this implies that $\text{Validate}_{(1,k)}(m, a, b, h, \alpha, x_1, \dots, x_k, f(x_1), \dots, f(x_k)) = 1$. Therefore we conclude that the original receiver will accept the signature $(a, b, h, \alpha, x_1, \dots, x_k, f(x_1), \dots, f(x_k))$ as genuine (definition 4.12).

Lastly, we have to prove that the conditions (i) $d = g^v \pmod{p}$, and (ii) $e/a^v = a^u b^1 \pmod{p}$ hold. It is easy to see that $d = g^v \pmod{p}$ holds if both the signer and original receiver are honestly following the protocol. In addition, condition $e/a^v = a^u b^1 \pmod{p}$ also holds, since

$$e/a^v \pmod{p} = (cd)^r / a^v \pmod{p}$$

$$= (g^t h^1 g^v)^t / a^v \pmod{p}$$

$$= g^{t(s+v)} h^t / a^v \pmod{p}$$

$$= a^{s+v} b^1 / a^v \pmod{p}$$

$$= a^s b^1 \pmod{p}.$$

Security of the Signing Protocol

In this section, we discuss the security of the signing protocol in the (1,k) designated confirmer protocol. Before the discussion of our result, we have to define the following.

Definition 4.13. Function $Sign_{(1,k)}(a,b,d,e,h,v,s,t,m,\alpha)$ will return 1 if and only if all of the conditions C4.2.5, C4.2.6 and C4.2.7 hold, otherwise $Sign_{(1,k)}(a,b,d,e,h,v,s,t,m,\alpha)=0$.

$$\text{C4.2.5} \quad d = g^v \pmod{p}$$

$$\text{C4.2.6} \quad e/a^s = a^s b^1 \pmod{p}$$

$$\text{C4.2.7} \quad \alpha^k = (H(a, b, h) \oplus F(m)) \pmod{n}$$

Definition 4.14. The original receiver will accept the signature $(a, b, h, \alpha, x_1, \dots, x_k, f(x_1), \dots, f(x_k)) \in G^4 \times Z_p^{2k}$ (corresponding to message $m \in G$) as *genuine* if and only if $\text{Sign}_{(1,k)}(a, b, d, e, h, v, s, t, m, \alpha) = 1$, where $(d, e, v) \in G \times G \times Z_q$ are responses from the signer and $(s, t) \in Z_q \times Z_q$ are random values chosen by the original receiver.

Theorem 4.3. Given an invalid signature $(a, b, h, \alpha, x_1, \dots, x_k, f(x_1), \dots, f(x_k)) \in G^4 \times Z_p^{2k}$ corresponding to message $m \in G$ (definition 4.11), then the original receiver will accept $(a, b, h, \alpha, x_1, \dots, x_k, f(x_1), \dots, f(x_k))$ as genuine with probability $1/q$ (definition 4.14).

Proof. By definition 4.14, the signer must respond to the original receiver with $(d, e, v) \in G \times G \times Z_q$ such that $\text{Sign}_{(1,k)}(a, b, d, e, h, v, s, t, m, \alpha) = 1$. Recall that condition C4.2.6 must be fulfilled if $\text{Sign}_{(1,k)}(a, b, d, e, h, v, s, t, m, \alpha) = 1$. This means the original receiver must receive $e \in G$ from the signer such that condition C.4.2.6 holds, where

$$\begin{aligned} e &= a^s b^t a^v \pmod{p} \\ &= a^{s+u_0 t+v} \pmod{p} \text{ with unknown } s \text{ and } t. \end{aligned}$$

This implies that the signer has to guess the correct value of $s + a_0 t + v \pmod{q}$ with a probability of $1/q$. This implies the signer can convince the original receiver to accept the signature $(a, b, h, \alpha, x_1, \dots, x_k, f(x_1), \dots, f(x_k))$ as genuine with probability no better than $1/q$. $\square$

4.2.2 Confirmation Protocol

This protocol allows any verifier to confirm the signature with the confirmer C_i , who has the private key $z_i \in Z_p$ and public key $h_i = g^{z_i} \pmod{p}$.

Figure 4.5. Confirmation Protocol for the $(1, k)$ Designated Confirmer Signature

Correctness of the Confirmation Protocol

If both the confirmer and verifier honestly follow the confirmation protocol, it is easy to see that $g^s = w \pmod{p}$. In addition, consider the following

$$\begin{aligned}
 c/h_i^s &= k^{a_0} w^{z_1} / h_i^s \pmod{p} \\
 &= (g^u a^d)^{a_0} g^{sz_1} / g^{sz_1} \pmod{p} \\
 &= g^{a_0 u} a^{a_0 d} \pmod{p} \\
 &= h^u b^d \pmod{p} \qquad \text{since } b = a^{a_0} \pmod{p}
 \end{aligned}$$

This implies that with high probability $b = a^{a_0} = a^{f(u)} \pmod{p}$ and that condition C4.2.1 holds. In addition, we have already proven that $\alpha^{x_1} \not\equiv (H(a,b) \oplus F(m)) \pmod{n}$ in chapter 4.2.1, which implies that condition C4.2.3 holds. Lastly, condition C4.2.2 must hold, since the confirmer has already ensured the condition before he sends (c,w) to the verifier. Consequently, we conclude that $\text{Validate}_{(1,k)}(m,a,b,h,\alpha,x_1,\dots,x_k,f(x_1),\dots,f(x_k)) = 1$ and therefore, the verifier will accept the signature $(a,b,h,\alpha,x_1,\dots,x_k,f(x_1),\dots,f(x_k))$ as genuine (definition 4.12).

Security of the Confirmation Protocol

In this chapter, we discuss the security of the confirmation protocol in the (1,k) designated confirmer protocol.

Definition 4.15. Function $Confirm_{(1,k)}(a,b,c,w,h,u,d,s,m,\alpha)$ will return 1 if and only if all of the conditions C4.2.7, C4.2.8 and C4.2.9 hold, otherwise $Confirm_{(1,k)}(a,b,c,w,h,u,d,s,m,\alpha)=0$.

$$\text{C4.2.7} \quad g^s = w \pmod{p}$$

$$\text{C4.2.8} \quad c/h_i^s = h^u b^d \pmod{p}$$

$$\text{C4.2.9} \quad \alpha^{k_v} = H(a,b,h) \oplus F(m) \pmod{n}$$

Definition 4.16. The verifier will accept the signature $(a,b,h,\alpha,x_1,\dots,x_k,f(x_1),\dots,f(x_k)) \in G^4 \times Z_p^k$ (corresponding to message $m \in G$), as genuine if and only if $Confirm_{(1,k)}(a,b,c,w,h,u,d,s,m,\alpha) = 1$, where $(w,c,s) \in G \times G \times Z_q$ are responses from the signer and $(u,d) \in Z_q \times Z_q$ are random values chosen by the verifier.

Theorem 4.2. Given an invalid signature $(a, b, h, \alpha, x_1, \dots, x_k, f(x_1), \dots, f(x_k)) \in G^4 \times Z_p^k$ corresponding to message $m \in G$ (definition 4.12), then the verifier will accept $(a, b, h, \alpha, x_1, \dots, x_k, f(x_1), \dots, f(x_k))$ as genuine with probability $1/q$. (definition 4.16)

Proof. By definition 4.15, the signer must respond to the original receiver with $(w, c, s) \in G \times G \times Z_q$ such that $\text{Confirm}_{(1,k)}(a, b, c, w, h, u, d, s, m, \alpha) = 1$. Recall that condition C4.2.8 must be fulfilled if $\text{Confirm}_{(1,k)}(a, b, c, w, h, u, d, s, m, \alpha) = 1$. This means the verifier must receive $c \in G$ from the signer such that condition C.4.2.8 holds, where

$$c = h^u b^d h_i^s \pmod{p}$$

$$= g^j \pmod{p} \quad \text{for some } j \in Z_q \text{ and unknown } (u, d).$$

This implies that the signer has to guess the correct value of $j \pmod{q}$ with a probability of $1/q$ and therefore the signer can convince the receiver to accept the signature $(a, b, h, \alpha, x_1, \dots, x_k, f(x_1), \dots, f(x_k))$ as genuine with probability $1/q$. $\square$

4.2.3 Unforgability of the (1,k) Threshold Designated Confirmer Signature

In this chapter, we will examine the possibility of forging a message by an adversary Alice. We assume Alice does not know the signer's private key, the confirmer's private key nor the shared key y_i .

We further assume that Alice has obtained a set of signatures $S = \{(a_1, b_1, h_1, \alpha_1, x_{11}, \dots, x_{1k}, f_1(x_{11}), \dots, f_1(x_{1k})) \dots, (a_s, b_s, h_s, \alpha_s, x_{s1}, \dots, x_{sk}, f_s(x_{s1}), \dots, f_s(x_{sk}))\}$ corresponding to the messages $M = \{m_1, \dots, m_s\}$. Additionally, we denote $\eta = \{f_1, \dots, f_s\}$.

If Alice wants to create a forged signature $(a_{s+1}, b_{s+1}, h_{s+1}, \alpha_{s+1}, x_{(s+1)1}, \dots, x_{(s+1)k}, f_{s+1}(x_{(s+1)1}), \dots, f_{s+1}(x_{(s+1)k})) \notin S$ corresponding to some message $m_{s+1} \notin M$, we must first consider the probability that Alice will be able generate such a function $f_{s+1} \notin \eta$. Under the assumption that the shared key y_i is not revealed to the public, Alice has virtually no hope of generating such a function, unless she can correctly deduce all of the y_i 's - with a probability of $1/q^k$. This implies that the probability of forging the signature is no better than $1/q^k$, if $f_{s+1} \notin \eta$.

Now we consider the case that Alice wants $f_{s+1} \in \eta$. Let $f_{s+1} = f_j$, for some $j \in \{1, \dots, s\}$. Recall that $h_j = g^{f_j(0)} \pmod{p}$ implies that $h_{s+1} = h_j$ in order that the condition C4.2.2 be fulfilled. To ensure condition C4.2.1 holds, Alice can compute a_{s+1}, b_{s+1} as a_j^t and b_j^t respectively for some $t \in \mathbb{Z}_q$, $t \neq 1$. Alice now has $(a_j^t, b_j^t, h_j, \alpha_{s+1}, x_{j1}, \dots, x_{jk}, f_j(x_{j1}), \dots, f_j(x_{jk}))$ with α_{s+1} unknown. However, computing α_{s+1} requires the private key of the signer; therefore Alice must break the RSA signature scheme [RSA79].

This demonstrates that in order for Alice to forge the signatures, she must correctly discover all of the y_i 's with a probability of $1/q^k$, or break the RSA signature scheme [RSA79].

Creating the Fake Transcript

Recall that in an undeniable signature scheme, any verifier must participate in the confirmation protocol in order to confirm the signature. To ensure this property in the $(1,k)$ -signature scheme, it must be easy to create a fake transcript for the confirmation protocol. Otherwise the verifier could confirm the signature based on the confirmation transcript from other parties. We emphasize that is not necessary for the signature to be genuine in the fake transcript. What we need to demonstrate is that it is easy to create a transcript (for the confirmation protocol) with all of the following conditions holding:

$$\mathbf{C4.2.10} \quad w = g^s \pmod{p}$$

$$\mathbf{C4.2.11} \quad c/h_i^s = h^u b^v \pmod{p}$$

$$\mathbf{C4.2.12} \quad \alpha^{k_i} = H(a,b,h) \oplus F(m) \pmod{n}$$

In addition, a forged transcript should take the following form:

Confirmer i (C_i)Recipient

$$u, d \in_R Z_q$$

$$\xleftarrow{k} \quad k = g^u a^d \pmod{p} \quad (1)$$

$$\xrightarrow{w, c} \quad (2)$$

$$\xleftarrow{u, d} \quad (3)$$

$$\xrightarrow{s} \quad w = g^s \pmod{p} \quad (4)$$

$$c/h_i^s = h^u b^d \pmod{p}$$

$$\alpha^{k_v} = H(a, b, h) \oplus F(m) \pmod{n}$$

Figure 4.6. The Forged Transcript for the (1,k) Designated Confirmer Signature.

The fake transcript can be created using the following steps:

- (i) We first select a random $\alpha \in Z_n$ and compute any tuple $(a, b, h) \in G^3$ such that $H(a, b, h) = \alpha^{k_v} \oplus F(m)$. (Recall that H is an invertible function.) Therefore a, b, h, α, m will fulfill **C4.2.12**.

- (ii) Compute $c = h^u b^d h_t^s \pmod{p}$ for some random $u, v, s \in \mathbb{Z}_q$. So that C4.2.11 holds.
- (iii) Compute $w = g^s \pmod{p}$ so that C4.2.10 holds.
- (iv) Compute $k = g^u a^d \pmod{p}$.

Notice that the fake transcript for the signing protocol can be created in a similar manner and will therefore not be discussed herein.

4.2.4 Conversion Protocol

Similar to the (1,1) signature scheme, the (1,k) designate confirmer signature can be converted into a conventional digital signature by any confirmer or the signer

$$\text{Let } r = a^w \pmod{p}$$

$$y = w + y_0 F(a, r), \quad \text{where } w \in {}_R \mathbb{Z}_q$$

Now the verifier can verify the correctness of the signature without the help of a confirmer using the validation check

$$a^y \stackrel{?}{=} r b^{F(a,r)} \pmod{p}, \text{ and } \alpha^{h_v} \stackrel{?}{=} H(a, b, h) \oplus F(m) \pmod{n}.$$

4.3 The $(1,1)^*$ Designated Confirmer Signature

The aforementioned designated confirmer signature allows the signer to confirm the signature with the original receiver only if the signer can recall the random value x . We proposed a variant of the confirmer signature which allows the signer to confirm the signature at any time, where it is not necessary that the signer keep track of random values being used for the message.

One simple solution for obtaining the $(t,k)^*$ designated confirmer signature is for the signer to keep track of the random value being used for each of the signed messages. However, this approach becomes infeasible if the signer has to sign a large number of messages. Our idea for the $(t,k)^*$ designated confirmer signature is simple - we encrypt the random value x by the signer's public key; the encrypted value now becomes part of the signature. Notice that x is encrypted by the signer's public key and would thus make it infeasible for any attacker to recover x from the message. Without loss of generality, we will only focus on the $(1,1)^*$ designated confirmer signature, but our result is applicable to the $(1,k)^*$ designated confirmer signature that we proposed previously.

System Model of $(1,1)^*$ Designated Confirmer Signature

Our model of the $(1,1)^*$ designated confirmer signature is similar to the model discussed in section 4.1. We assume there is a confirmer, Colin, and a signer, Simon. Simon also has a private-public key $(k_s, k_{s'})$ as mentioned in chapter 4.1. But we now

separate the signing protocol into two protocols: the signature protocol and the signer's confirmation protocol. Note that the confirmation referred to in chapter 4.1 is distinct from the signer's confirmation protocol.

The signature protocol allows the signer to create a signature which is verifiable by the signer or the confirmer at any time. Details of the protocol follow.

Signing Protocol:

To sign for the message $m \in G$, the signer performs the following protocol:

Signer

$$x \in_R Z_q$$

$$a = g^x \pmod{p}, \quad b = h^x \pmod{p}, \quad \beta = x^{k_v} \pmod{p}$$

$$\alpha = (H(a, b) \oplus F(m, \beta))^{k_s} \pmod{p}$$

the signature is given by the tuple (a, b, β, α) .

Signer's Confirmation Protocol**Signer****Recipient** **Figure 4.7.** Signer's Confirmation Protocol for the (1,1)* Designated Confirmer Signature

The receiver accepts the signature (a,b,α) as genuine if and only if

C4.3.1 $d = g^r \pmod p$, and

C4.3.2 $e/a^r = a^s b^t \pmod p$, and

$$\mathbf{C4.3.3} \quad \alpha^{k_1} = H(a,b) \oplus r(m,\beta) \pmod{n}.$$

The security of the above protocol is exactly the same as the (1,1) designated confirmer signature that we discussed earlier. The only difference is that we included the encrypted value of x as part of the signature, thus allowing the opponent to assume the role of signer if and only if the opponent can decrypt the random value x . The probability that the signer can convince the verifier to accept a fake signature still remains $1/q$ and details of the proof are found in section 4.1

2 of/de 2

PM-1 3½"x4" PHOTOGRAPHIC MICROCOPY TARGET
NBS 1010a ANSI/ISO #2 EQUIVALENT

PRECISIONSM RESOLUTION TARGETS

Confirmation Protocol:

The confirmer's confirmation allows the receiver to validate the signature with the designated confirmer, though the signature is not created by the confirmer

Confirmer**Recipient**

$$u, v \in \mathbb{Z}_q$$

$$t \in \mathbb{Z}_q$$

$$\xleftarrow{k}$$

$$k = g^u a^v \pmod{p}$$

$$y = g^t \pmod{p}$$

$$s = (ky)^z \pmod{p}$$

$$\xrightarrow{l, s}$$

$$g^u a^v \stackrel{?}{=} k \pmod{p}$$

$$\xleftarrow{u, v}$$

$$\xrightarrow{t}$$

$$g^t \stackrel{?}{=} y \pmod{p}$$

$$s/h^l \stackrel{?}{=} h^u b^v \pmod{p}$$

$$\alpha^{k_v} \stackrel{?}{=} H(a, b) \oplus F(m, \beta) \pmod{n}$$

Figure 4.8. Confirmer's Confirmation Protocol for the (1,1)* Designated Confirmer Signature

Again, this protocol retains the probability that the confirmer can convince the verifier to accept a fake signature found in section 4.1.

4.3.1 Security of the (1,1)* Designated Confirmer Signature

This section will discuss the security of the (1,1)* designated confirmer signature. For the signing protocol, the availability of β does not affect the security, as $x = \beta^{k_s} \pmod{n}$ does not give any additional information to the signer, since x is a random value chosen by the signer himself.

For the confirmation protocol, the availability of β again does not give any extra information to the confirmer, since recovering x from β requires that the confirmer to break the RSA scheme [RSA79]. In the view of the confirmer, β is nothing more than a useless random number.

Unforgability

With the availability of the $\beta = x^{k_s} \pmod{n}$ in (1,1)*-signature scheme, would it be possible for the adversary, Alice, to forge a signature? Assume Alice selects $x \in_R Z_q$ which will allow her to find $\beta = x^{k_s} \pmod{n}$, $a = g^x \pmod{p}$ and $b = h^x \pmod{p}$. However, to determine $\alpha = (H(a, b) \oplus F(m, \beta))^{k_s} \pmod{n}$ requires her to find k_s . This would be difficult under the assumption that the RSA signature [RSA79] is intractable. On the other hand, if Alice starts from some random α and x that will give $\beta = x^{k_s} \pmod{n}$ she will have $H(a, b) = \alpha^{k_s} \oplus F(m, \beta) \pmod{n}$. The probability of forging a signature is exactly the same as the (1,1)-signature scheme, which is $1/n$, provided that the function $H(a, b)$ is defined as $H(a, b) = a \cdot b \pmod{p}$. Details of the proof can be found in section 4.1.1.

5. Conclusion

In this report, we have discussed the digital signature and the undeniable signature as well as the designated confirmer signature. The digital signature preserves most of the important properties of handwritten signatures such as the unforgability, public verification and self-authenticity properties. However, as we demonstrated in chapter 3.1, the self-authenticity property is too strong in many applications. A typical example could involve a software vendor who provides a digital signature for the purpose of software authenticity. He may wish to allow only legitimate users to verify the signature. The undeniable signature is particularly suitable for such applications, in that it requires the cooperation of the signer for signature confirmation.

One of the advantages that the undeniable signature has over the conventional digital signature is that it requires the cooperation of the signer in order to confirm the signature. However, if the signer refuses to cooperate or is unavailable for confirmation, then the verifier can not make use of the signature. This problem can be solved by the designated confirmer signature scheme, which separates the responsibility of signing and confirmation into two distinct parties: the signer and the verifier. Although the signer has no way to prove the validity of the signature to any verifier, the verifiers can confirm the signature with the help of the confirmer.

We have also defined the (t,k) threshold designated confirmer signature scheme in chapter 4. The system model consists of a signer and k confirmers, such that the signer is

responsible for the signing of a message, but the signature can only be confirmed with any k out of the t confirmers. The designated confirmer signature proposed by [Cha94] is a special case of our $(1,1)$ threshold designated confirmer signature. As stated in chapter 4, the $(1,1)$ designated confirmer signature does suffer the same problem as the undeniable signature; if, in this case, the confirmer is unavailable or refuses to cooperate with the confirmation, then none of the verifiers can make use of the signature. This is in contrast to the undeniable signature, in which the signer may be unavailable or may refuse to cooperate.

In chapter 4.1, we proposed a $(1,k)$ designated confirmation signature, which is a generalization of the designated confirmer signature proposed by [Cha94]. In our model the signer is responsible for the signing while the confirmation can be done with the cooperation of any 1 out of the k confirmers. Compared to the $(1,1)$ designated confirmer signature, the confirmation protocol in our model does not depend on any individual confirmer. In case a particular confirmer is busy or refuses to cooperate, the verifier can confirm the signature with any one of the remaining confirmers. This property greatly increases the reliability of the $(1,1)$ designated confirmer signature.

In addition to the $(1,k)$ designated confirmation signature, we also proposed a $(1,1)^*$ designated confirmer signature which, unlike the $(1,1)$ designated confirmation signature, allows the signer to prove the signature validity to any verifier. Advantages of the $(1,1)^*$ designated confirmer signature are obvious: Consider that a computer network

consists of many signers and only one confirmer. In this case, the confirmer will be extremely busy for all of the confirmation requests; hence the availability of the confirmer will be dramatically decreased and he may not be able to fulfill all of the confirmation requests resulting in some of the signatures not being confirmed. With the $(1,1)^*$ designated confirmer signature, the signer may responsible for confirmation and the confirmer is only involved whenever the verifier cannot get help from the signer. In this case the system will be much more reliable when compared to the $(1,1)$ designated confirmer signature scheme. Although we have only discussed the $(1,1)^*$ designated confirmer signature scheme, our model can easily be generalized as the $(1,k)^*$ designated confirmer signature.

Future research in this area could involve further generalization of the $(1,k)$ threshold designated confirmer signature scheme, resulting in a (t,k) threshold designated confirmer signature. In addition, as shown in [BL90], the (t,k) threshold access structure has only limited applications in real life. A further generalization case of the designated confirmer signature is as follows: let C be the set of all of the confirmers, and let $\Gamma \subseteq 2^C$. We may want the signature confirmable by any subset $B \subseteq C$, provided that $A \subseteq B \subseteq C$, for some $A \in \Gamma$. For example, consider a system that consists of 5 confirmers $C = \{C_1, C_2, C_3, C_4, C_5\}$ such that C_1 has much more computation power when comparing with $C_2, \dots, C_5$. For this reason, we would like the confirmer, C_1 , to participate in more confirmation requests than $C_2, \dots, C_5$ does. Assume we have the following policy: the signature is confirmable with either (i) C_1

and any one of the remaining $\{C_2, \dots, C_5\}$, or (ii) any three confirmers from $\{C_2, \dots, C_5\}$. We can define Γ as $\{\{C_1, C_2\}, \{C_1, C_3\}, \{C_1, C_4\}, \{C_1, C_5\}, \{C_2, C_3, C_4\}, \{C_2, C_3, C_5\}, \{C_3, C_4, C_5\}\}$. In this example, the (t, k) threshold designated confirmer will not fulfill our requirement. Therefore, the designated confirmer signature for general access structure $\Gamma \subseteq 2^C$ will be much more flexible, compared to the (t, k) designated confirmer signature scheme.

In conclusion, the existence of the designated confirmer signature for the (t, k) threshold and general access structure is an open question in cryptography. Further research on those signature schemes may result in wider usage of this system in commerce and industry.

Appendix A . Zero-Knowledge Proof

Assume there exists a prover, Peggy, who knows the combination of a safety deposit box. She wishes to convince another party, Vicky, that she knows the combination. One simple solution is that Peggy tells Vicky the combination. However, this approach is not desirable, since Peggy has to give Vicky much more information than the fact that "Peggy knows the combination". Very informally, a zero-knowledge proof [GMR89] is an interactive proof system which allows the prover to convince the verifier about some fact without revealing any extra information to the verifier

We first consider the computational model of the zero-knowledge proof system. We will restrict our attention to all NP problems, since they are all polynomial-time verifiable, while no known polynomial-time algorithm can solve the problem. In this chapter, we shall demonstrate the zero-knowledge proof system using a simple example which allows the prover to convince the verifier that two graphs G_1 and G_2 are isomorphic.

Definition A1. A graph G is defined as the ordered tuple (V,E) where V is the set of the vertices, $E=\{(u_1,v_1), \dots, (u_t,v_t)\}$ and E is the set of edges in G where $(u_i,v_i) \in V^2$ and $u_i \neq v_i$, for $1 \leq i \leq t$.

Definition A2. A graph $G_1=(V_1,E_1)$ is said to be isomorphic to another graph $G_2=(V_2,E_2)$ if $V_1=V_2$ and there exists a permutation π such that $E_2=\{(\pi(u), \pi(v)) \mid (u,v) \in E_1\}$. In figure A1, G_1 is isomorphic to G_2 under the permutation $\pi=(1342)$

Figure A1 Graph Isomorphism

Consider the problem that the prover, Peggy, knows the permutation π , such that G_1 is isomorphic to G_2 under permutation π and Peggy wants to convince Vicky that she knows the permutation. Additionally, we assume that Vicky, the verifier, and Peggy know G_1 and G_2 but Vicky does not know whether G_1 and G_2 are isomorphic (i.e. Vicky does not know the permutation π). We are looking for an interactive proof system which allows Peggy to convince Vicky that “Peggy knows the permutation”, without revealing any information other than “Peggy knows the permutation”. Informally, such a proof system is known as the zero-knowledge interactive proof system (or zero-knowledge proof). An important feature of the zero-knowledge proof system is that a verifier, already convinced of a fact, has no way to likewise convince others, as he does not know how to prove the information.

Definition A3 An Interactive Proof system (IPS) consists of two distinct parties: the prover and the verifier. The prover wishes to convince the verifier to accept some fact. The prover convinces the verifier by executing a protocol for n rounds.

In each of the rounds, the verifier sends a challenge to the prover, who responds to the verifier based on some private computations. The verifier will either accept or reject the reply depending on whether the prover correctly responds to the challenge. We define the protocol as an interactive proof system for the decision problem Π , if properties CA1 and CA2 are satisfied, provided of course, that both Peggy and Vicky follow the protocol.

CA1 Completeness. If x is a yes-instance of the decision problem Π , then Vicky will always accept Peggy's proof. In our graph isomorphism example, the completeness property means Vicky will always accept Peggy's proof, provided that G_1 is isomorphic to G_2 .

CA2 Soundness. If x is a no-instance of the decision problem Π , then Vicky will accept Peggy's proof with a negligible probability. Over n rounds of the graph isomorphism example, if G_1 is not isomorphic to G_2 , then Vicky will accept Peggy's proof with probability $1/2^n$.

Now we are ready to discuss the zero-knowledge interactive proof system for the graph isomorphism problem. A zero-knowledge proof for the graph isomorphism problem is shown in figure A2. Peggy would like to convince Vicky to accept the fact that G_1 and

G_2 are isomorphic without revealing any information about the permutation π . Notice that both of the computations can be done in polynomial time - $O(|V|)$.

Input: $G_1=(V_1,E_1)$ and $G_2=(V_2,E_2)$ such that $V_1=V_2=\{1, 2, \dots, n\}$.

(1) Repeat n times

(2) Peggy chooses a random permutation σ of $\{1, \dots, n\}$ and sends H to Vicky such that H is the image of G_2 under the permutation σ .

(3) Vicky chooses an integer i from $\{1, 2\}$ and sends i to Peggy.

(4) Define $\rho=\sigma$ if $i=2$, otherwise define $\rho=\sigma\pi$. Peggy now sends ρ to Vicky.

(5) Vicky checks to see if H is the image of G_i under ρ .

(6) Vicky accepts Peggy's proof if H is the image of G_i in each of the n rounds

Figure A2 Zero-Knowledge Proof System for the Graph Isomorphism Problem

We need the following definitions in order to show that the above IPS satisfies the completeness and soundness properties.

Definition A4. Define the function $M(G, \pi) = G'$ such that G' is the isomorphic copy of the graph G under the permutation π . More precisely, if $G=(V,E)$, then $G'=(V,E')$ such that $E'=\{(\pi(u),\pi(v)) \mid \forall (u,v) \in E\}$.

Definition A5. G_1 is isomorphic to G_2 under the permutation π if and only if $G_2=M(G_1,\pi)$.

Lemma A1 For any graph $G=(V,E)$ and any permutation $\pi_1\pi_2$. $M(G,\pi_1\pi_2) = M(M(G, \pi_2), \pi_1)$.

Proof. Let $G_1=(V,E_1)=M(G,\pi_1\pi_2)$, $G_2=(V,E_2)=M(G,\pi_2)$. Thus we have $E_1=\{(\pi_1\pi_2(u),\pi_1\pi_2(v)) \mid \forall (u,v) \in E\}$ and $E_2=\{(\pi_2(u),\pi_2(v)) \mid \forall (u,v) \in E\}$. Now consider $M(M(G,\pi_2),\pi_1)=M(G_2,\pi_1)=G_3=(V,E_3)$, and we then have $E_3=\{(\pi_1(u),\pi_1(v)) \mid \forall (u,v) \in E_2\}=\{(\pi_1\pi_2(u),\pi_1\pi_2(v)) \mid \forall (u,v) \in E\}=E_1$. This implies that $M(G,\pi_1\pi_2)=M(M(G,\pi_2),\pi_1) \square$

Definition A6. In the interactive proof system as stated in figure A2, Vicky will accept the round if and only if $H=M(G, \rho)$. (Step 5 of the IPS, as stated in figure A2.)

Theorem A1. The completeness properties of the IPS holds as stated in figure A2.

Proof. First observe that in step 2 of figure A2, we have

$$H=M(G_2,\sigma). \quad (1)$$

Consider $i=2$, meaning Vicky's reply to the verifier shall be $\rho=\sigma$, thus $M(G_2,\rho)=M(G_2,\sigma)=H$. Therefore, by definition A5, Vicky will accept the round.

Now consider $i=1$, then we have $\rho=\sigma\pi$. By lemma A1, we have

$$M(G_1,\rho)=M(G_1,\sigma\pi)=M(M(G_1,\pi),\sigma). \quad (2)$$

Recall that G_1 is isomorphic to G_2 under the permutation π . Thus, by definition A5 we have

$$G_2=M(G_1,\pi). \quad (3)$$

(1) (2) and (3) imply

$$M(G_1,\rho)=M(G_2,\sigma)=H.$$

Therefore Vicky will accept the round if $i=1$. Since Vicky will accept the round in both the cases of $i=1$ or $i=2$, this implies that the completeness property holds in the proving system. $\square$

Theorem A2. The soundness property of the IPS holds, as stated in Figure A2

Proof. By definition A6, for Vicky to accept the round Peggy must return ρ such that $H=M(G_i, \rho)$. Assume that G_1 and G_2 are non-isomorphic. Then there does not exist any π such that $G_2=M(G_1, \pi)$. Assume that if Peggy computes $H=M(G_2, \sigma)$, where σ is some random permutation, and Vicky selects $i=1$, then it is impossible for Peggy to find any permutation ρ such that $H=M(G_1, \rho)$. Therefore, if Peggy computes $H=M(G_2, \sigma)$ for each of the n rounds, then Vicky will accept the proof only if the verifier selects $i=1$ for all of the rounds. This gives the probability $1/2^n$ that Peggy will accept the proof.

On the other hand, if Peggy can predict the random value i selected by Vicky, then she can compute $H=M(G_i, \rho)$ for some randomly selected permutation ρ . In this case Peggy can convince Vicky to accept the proof only if Peggy can correctly guess all of the random values of i selected by Vicky. This concludes that Peggy can convince Vicky to accept the proof with probability $1/2^n$ $\square$

We are going to discuss why the proof system mentioned in figure A2 is “zero-knowledge”. At the end of each round, Vicky will accept a permutation ρ such that either

G_1 or G_2 is isomorphic to H under that permutation. Such a set of the permutation ρ 's does not seem to give the verifier any information about π .

In order to precisely define the notation of the zero-knowledge proof system it is necessary to examine the information that Vicky obtains by participating in the IPS. We denoted the transcript of the proof system as all of the information that Vicky will obtain during one instance of the interactive proof system. The transcript consists of all of the following information:

- (1) G_1 and G_2
- (2) All the messages being sent during the proof.
- (3) The random number $i \in \{1, 2\}$ for each round.

Therefore the transcript T for the IPS mentioned before takes the form of:
 $T = ((G_1, G_2); (H_1, i_1, \rho_1); \dots; (H_n, i_n, \rho_n))$. One interesting problem is this: does the information contained in T give Vicky any advantage for determining the permutation π ? We will now prove that Vicky gains absolutely no "new information" from the transcript.

One major reason Vicky cannot gain any extra information from T is that she could easily create a fake transcript, F, which is indistinguishable from T. Roughly speaking, the transcript T is just a reconstruction of some information that Vicky already knew. Thus Vicky is not able to obtain any new information from the transcript other than that which she already knew. As shown in figure A3, there is a polynomial-time algorithm which allows Vicky (or anybody else) to create a forged transcript which is “indistinguishable” from the real transcript. We refer to such a forging algorithm as the *simulator*. We are now able to explain why the above protocol is zero-knowledge. Assume that Vicky could gain some information from the real transcript T. Since F and T are “indistinguishable”, this implies that the same information can also be found from the forged transcript F. However, this contradicts with the assumption that F is created by the forger himself, since it is not possible for him to create some new information by all the information that Vicky already knew.

Input: $G_1=(V_1,E_1)$ and $G_2=(V_2,E_2)$ such that $V_1=V_2=\{1, 2, \dots, n\}$ and G_1 and G_2 are isomorphic.

- (1) $T=(G_1,G_2)$
- (2) Repeat n times
- (3) Set $i_j \in_R \{1, 2\}$
- (4) Set ρ_j as a random permutation of $\{1, \dots, n\}$
- (5) Set H_j as the image of G_{i_j} under ρ_j
- (6) Append (H_j, i_j, ρ_j) to the end of T

Figure A3 The Polynomial-time Forging Algorithm for the Transcript for the Graph Isomorphism Proof System

Now we will formally define the zero-knowledge proof system. ([Stin95] pp.393)

Definition A7 Zero-knowledge interactive proof system. Suppose that we have a polynomial-time interactive proof system for a decision problem Π , and a polynomial-time simulator S . Denote the set of all possible transcripts that could be produced as a result of Peggy and Vicky carrying out the interactive proof with a yes-instance x by $T(x)$, and denote the set of all possible forged transcripts that could be produced by S by $F(x)$. For any transcript $t \in T(x)$, let $\rho_T(t)$ denote the

probability that t is the transcript produced from the interactive proof. Similarly, for $t \in F(x)$, let $p_F(t)$ denote the probability that t is the (forged) transcript produced by S . Suppose that $T(x) = F(x)$, and for any $t \in T(x)$, suppose that $p_T(t) = p_F(t)$. (In other words, the set of real transcripts is identical to the set of forged transcripts, and the two probability distributions are identical). Then we define the interactive proof system to be perfect zero-knowledge for Vicky.

Theorem A3. The interactive proof system in figure A2 is perfect zero-knowledge.

Proof. First, any transcript will consist of (G_1, G_2) and n tuples of the form (H_j, i_j, ρ_j) . We will now discuss the probability distribution for the real transcript. Since i_j, ρ_j are independent random variables, it is easy to see that there are exactly $2 \times (n!)$ possible choices for (i_j, ρ_j) . Additionally, each (i_j, ρ_j) will uniquely determine the graph H_j . Assuming that i_j and ρ_j are chosen at random by Vicky, we conclude that every tuple (H_j, i_j, ρ_j) is equally probable with probability $(2 \times (n!))^{-1}$. Now we consider the probability distribution of the forged transcript. Similarly, there are exactly $2 \times (n!)$ possible choices for (i_j, ρ_j) and each of them uniquely determines an isomorphic graph H_j . Therefore every tuple (H_j, i_j, ρ_j) is equally probable with probability $(2 \times (n!))^{-1}$. Finally, we conclude that the interactive proof system is

perfect zero knowledge, since the probability distribution of the real transcript is identical to that of the forged transcript (definition A7). $\square$

Appendix A. Bit Commitment Scheme

The idea of the bit commitment scheme is best illustrated by a simple example. Assume that Alice writes a message and locks it in a safety deposit box. The key to the safety deposit box is then given to Bob, since Bob has the key he is certain that Alice cannot alter the message.

One way to achieve the bit commitment scheme is by using the probabilistic cryptosystem proposed by S. Goldwasser and A. Micali [GM84]. We will first describe the probabilistic cryptosystem followed by the bit commitment scheme.

Probabilistic Cryptosystem

Consider the case that Alice wants to privately send a bit $b \in \{0, 1\}$ to Bob that is encrypted using a public key cryptosystem. We denote E as the encryption function under the public key and D as the decryption function under the private key. Alice will send Bob the encrypted value of b , i.e. $E(b)$, over an unprotected channel. Although it is computationally infeasible for the opponent, Oscar, to compute the private key or decrypt the ciphertext, it is easy for Oscar to compute $y_1 = E(0)$ and $y_2 = E(1)$. In this case Oscar is able to determine the message b by comparing $E(b)$ with y_1 and y_2 .

Informally, in a probabilistic cryptosystem, there exist many possible encrypted values of each of the plaintext x . Additionally, it is infeasible to test whether a ciphertext is an encryption of a particular plaintext. We need to define the quadratic residue and quadratic non-residue before we discuss the probabilistic cryptosystem proposed by [GM84]

Definition B1. Quadratic Residue, $QR(n) = \{x | x = y^2 \pmod{n}, \exists y \in \mathbb{Z}_n^*\}$. That is, $QR(n)$ consists of all the elements in $\mathbb{Z}_n^*$ such that x is a square of some $y \in \mathbb{Z}_n^*$.

Definition B2. Quadratic Non-residue, $QNR(n) = \mathbb{Z}_n^* / QR(n)$

The public and private key of the probabilistic cryptosystem proposed by [GM84] is as follows.

Private key: p, q , where p, q are large prime,

Public key: $n = pq, m \in QNR(n)$.

The Encryption Function

To encrypt a message $x \in \{0, 1\}$, we compute

$$y = m^x r^2 \pmod{n} \quad \text{where } r \in_R \mathbb{Z}_n^*.$$

Theorem B3. If $x=0$, $m \in \text{QNR}(n)$ and $r \in \mathbb{Z}_n^*$, then $y = m \cdot r^2 \in \text{QR}(n)$.

Proof. If $x=0$, then we have $y = r^2 \pmod{n}$, thus $y \in \text{QR}(n)$ (definition B1). $\square$

Theorem B4. If $x=1$, $m \in \text{QNR}(n)$ and $r \in \mathbb{Z}_n^*$, then $y = m \cdot r^2 \in \text{QNR}(n)$.

Proof. If $x=1$, then we have $y = m \cdot r^2 \pmod{n}$. Assume that $y \in \text{QR}(n)$, which implies $m \cdot r^2 = k^2 \pmod{p}$ for some $k \in \mathbb{Z}_p^*$ such that $m = k^2 \pmod{p}$. However, $m = k^2 \pmod{p}$ contradicts the assumption that $m \in \text{QNR}(n)$. Therefore $m \cdot r^2 \notin \text{QR}(n)$, or equivalently, $m \cdot r^2 \in \text{QNR}(n)$. $\square$

Theorem B3 and B4 imply that the encryption of 0 is an element of $\text{QR}(n)$, while the encryption of 1 is an element of $\text{QNR}(n)$. Therefore, we need to find an efficient algorithm $A(y)$ to determine whether $y \in \text{QR}(n)$ in order to decrypt the ciphertext.

The Decryption Function

Definition B3. Assume p is a prime. For any integer $a \in \mathbb{Z}_p^*$, define the *Legendre*

symbol $\left(\frac{a}{p}\right)$ as follows:

$$\left(\frac{a}{p}\right) = \begin{cases} 0 & \text{if } a \equiv 0 \pmod{p} \\ 1 & \text{if } a \in \text{QR}(p) \\ -1 & \text{if } a \in \text{QNR}(p). \end{cases}$$

Definition B4. Let $n = \prod_{i=1}^r p_i$, where $p_1, \dots, p_r$ are prime. The *Legendre-Jacobi*

symbol is given by

$$\left(\frac{a}{n}\right) = \begin{cases} 0 & \text{if } a \equiv 0 \pmod{n} \\ 1 & \text{if } a \in \text{QR}(n) \\ -1 & \text{if } a \in \text{QNR}(n). \end{cases}$$

Additionally, we state the following theorems with the proofs omitted.

Theorem B5. If p is prime, then

$$\left(\frac{a}{p}\right) = a^{(p-1)/2} \pmod{p}.$$

Proof. See [Kra86] pp.16-17. $\square$

Theorem B6. If $n = \prod_{i=1}^r p_i$, then $\left(\frac{a}{n}\right) = \prod_{i=1}^r \left(\frac{a}{p_i}\right)$, where $\left(\frac{a}{n}\right)$ and $\left(\frac{a}{p_i}\right)$ are the

Legendre-Jacobi symbol and Legend symbol respectively.

Proof. See [Kra86] pp17-18.

Theorem B7. If $n=pq$ and $x \in \mathbb{Z}_n^*$, then there exists an algorithm which will determine whether $x \in \text{QR}(n)$ in $O(\log(p) + \log(q))$ time.

Proof. By definition B4, we have $x \in \text{QR}(n)$ if and only if $\left(\frac{a}{n}\right) = 1$. Recall that $n=pq$

and by theorem B6, we have $\left(\frac{a}{n}\right) = \left(\frac{a}{p}\right) \left(\frac{a}{q}\right)$. Consider the problem of computing

$\left(\frac{a}{p}\right)$. By theorem B5, it can be computed as $\left(\frac{a}{p}\right) = a^{(p-1)/2} \pmod{p}$. This implies

that $\left(\frac{a}{p}\right)$ could be computed in $O(\log p)$ time. Similarly, $\left(\frac{a}{q}\right)$ can also be computed in $O(\log q)$ time. This implies that $\left(\frac{a}{n}\right)$ is computable in $O(\log(p) + \log(q))$ time. $\square$

Theorem B8. In the probabilistic cryptosystem proposed by [GM84], assume $y = m^x r^2 \pmod{m}$, is a ciphertext and (p, q) constitute the private key. There then exists a efficient algorithm for decrypting the ciphertext (i.e. determine $x \in \{0, 1\}$), provided that the private key (p, q) is known.

Proof. It is a direct result of Theorem B7. $\square$

The Bit Commitment Scheme

Assume that if Alice wants to commit a bit $x \in \{0, 1\}$ to Bob, then she will have to randomly select $r \in \mathbb{Z}_n^*$ and compute $y = m^x r^2 \pmod{n}$ (the value y is also referred to as the *blob*). Alice should not reveal x until she wants to open the commitment.

If Alice later wants to open the commitment, she can do it by revealing r . In this case Bob can uniquely determine x , since $x=1$ if and only if $y = m r^2 \pmod{n}$.

Appendix C : Shamir Secret Sharing Scheme and Secure Message Broadcasting

A threshold secret sharing scheme is a cryptographic protocol which allows a secret s to split into n shares in such a way that any t of the n shares can recover the secret. Assume that there is a dealer D and n share holders $C=\{c_1, \dots, c_n\}$. The dealer is able to split the secret s into n pieces $S=\{s_1, \dots, s_n\}$, referred to as shares, in such a way that the secret can only be recovered by any subset $S' \in S$, provided that $|S'| \geq t$. We assume that share holder c_i will privately receive the share s_i from the dealer, therefore allowing any t (or more) share holders to recover the secret. Such secret sharing is known as the (t,n) threshold secret sharing scheme. Note that any subset of $(t-1)$ shares or less cannot recover the secret.

We will discuss a method of constructing a (t,n) threshold scheme known as the Shamir threshold scheme, which was proposed by A. Shamir in 1979 [Sha79]. The Shamir threshold scheme is based on the polynomial interpolation over a finite field Z_p where p is a large prime. The share and secret spaces are given by Z_p . The Shamir threshold scheme can be divided into two sub-protocols: the share distribution and the secret reconstruction protocol.

The Share Distribution Protocol

Assume the secret is $a_0 \in Z_p$, and $a_1, \dots, a_{t-1} \in Z_p$ are random values selected by the dealer. Define $f(x)$ as the following polynomial:

$$f(x) = a_0 + a_1x + \dots + a_{t-1}x^{t-1} \pmod{p}$$

Notice that $f(0) = a_0$ is the secret that the dealer wishes to share with the share holders. The dealer now privately send the share $s_i = f(i)$ to the share holder C_i , $1 \leq i \leq n$.

The Secret Recovery Protocol

Assume share holders $C_{x_1}, \dots, C_{x_t}$ are going to recover the secret a_0 . Each of the share holders C_{x_i} , $1 \leq i \leq t$, will publish their shares s_i . We can therefore obtain t linear equations with t unknowns $a_0, \dots, a_{t-1}$ as follows:

$$\begin{array}{ccccccccc} a_0 & + & a_1x_1 & + & a_2x_1^2 & + & \dots & + & a_{t-1}x_1^{t-1} & = & f(x_1) \\ a_0 & + & a_1x_2 & + & a_2x_2^2 & + & \dots & + & a_{t-1}x_2^{t-1} & = & f(x_2) \\ \vdots & & \vdots & & \vdots & & & & \vdots & & \vdots \\ a_0 & + & a_1x_t & + & a_2x_t^2 & + & \dots & + & a_{t-1}x_t^{t-1} & = & f(x_t). \end{array}$$

We can, therefore, recover the secret a_0 by solving the equations system above.

Theorem B1. If $S = \{(s_1, x_1), \dots, (s_t, x_t)\}$, where $x_1, \dots, x_t$ are distinct values and $(s_i, x_i) \in \mathbb{Z}_p^2$, then we can uniquely determine a polynomial $f(x) = a_0 + a_1x + \dots + a_{t-1}x^{t-1} \pmod{p}$ such that $s_i = f(x_i)$ for all $i \in \{1, \dots, t\}$.

Proof. First, the above information gives the following equations system:

$$\begin{array}{ccccccccc} a_0 & + & a_1x_1 & + & a_2x_1^2 & + & \dots & + & a_{t-1}x_1^{t-1} & = & f(x_1) \\ a_0 & + & a_1x_2 & + & a_2x_2^2 & + & \dots & + & a_{t-1}x_2^{t-1} & = & f(x_2) \\ \vdots & & \vdots & & \vdots & & & & \vdots & & \vdots \\ a_0 & + & a_1x_t & + & a_2x_t^2 & + & \dots & + & a_{t-1}x_t^{t-1} & = & f(x_t) \end{array}$$

where $a_0, \dots, a_{t-1}$ are unknowns. It is easy to see that the above system has a unique solution, since we can rewrite the above system as a Vandermonde matrix as follows

$$\begin{bmatrix} 1 & x_1 & x_1^2 & \dots & x_1^{t-1} \\ 1 & x_2 & x_2^2 & \dots & x_2^{t-1} \\ \vdots & \vdots & \vdots & & \vdots \\ 1 & x_t & x_t^2 & \dots & x_t^{t-1} \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ \vdots \\ a_{t-1} \end{bmatrix} = \begin{bmatrix} s_{x_1} \\ s_{x_2} \\ \vdots \\ s_{x_t} \end{bmatrix}$$

The determinant of the Vandermonde matrix is computable by the formula

$$\det[A] = \prod_{1 \leq j < k \leq t} (x_k - x_j) \pmod{p}$$

x_i 's are all distinct values, implying that $\det[A] \neq 0$ and therefore $(a_0, \dots, a_{t-1})$ has a unique solution over the field $\mathbb{Z}_p$. $\square$

Theorem B2. In the Shamir secret sharing scheme, the secret a_0 can be uniquely determined by t distinct shares $\{s_{x_1}, \dots, s_{x_t}\}$.

Proof. Let $f(x) = a_0 + a_1x + \dots + a_{t-1}x^{t-1} \pmod{p}$ be the function that the dealer selected in the share distribution phase. Since $\{s_{x_1}, \dots, s_{x_t}\}$ are distinct shares, therefore, we have $\{(x_1, s_{x_1}), \dots, (x_t, s_{x_t})\}$. Therefore by theorem B1, we can uniquely determine a polynomial $f(x)$ such that $f(x_i) = s_{x_i} \pmod{p}$. Consequently the secret a_0 can be reconstructed by evaluating $f(0)$. $\square$

Secrecy of the Shamir Threshold Scheme

Assume that there are only $t-1$ shares available for the secret recovery protocol. In this case we are interested in how much information about the secret can be recovered from those $t-1$ shares. Obviously, the $t-1$ shares will give a system of $t-1$ equations with t unknowns. Assuming $f(0) = a_0$, the $t-1$ shares together with a_0 efficiently yield a system of t equations, we can uniquely determine the secret. However, there is a unique polynomial $f_{a_0}(x)$ for every possible value of a_0 . Hence all of the a_0 's are equally likely to be the secret. Thus, we conclude that any group of $t-1$ shares will provide absolutely no information about the secret.

Appendix D. Secure Message Broadcasting

Consider that there exists a dealer, D , and n receivers, $C=\{C_1, \dots, C_n\}$, in a communication network, such that each of the parties, C_i , has an unprotected communication channel with the dealer. Suppose that each of the users C_i privately shares a key, k_i , with the dealer. Further consider that the dealer wishes to privately broadcast a message, m , to some subset $B=\{C_{x_1}, \dots, C_{x_t}\} \subseteq C$. A simple solution would be for the dealer to individually send the encrypted message $E_{k_{x_i}}(m)$ to receiver x_i where $E_{k_{x_i}}(\cdot)$ is the encryption function under the shared key k_i . This solution is undesirable since the dealer has to repeatedly encrypt the same message. Surprisingly, this problem can be solved by the following Secure Message Broadcast Protocol (SMBPP) which is based on the idea of Shamir's secret sharing scheme.

SMBP (1)

Assuming y_i is the key shared by the dealer and C_i , $1 \leq i \leq n$. Let a_0 be the message that the dealer wishes to broadcast to the users $B=\{C_{x_1}, \dots, C_{x_t}\} \subseteq C$, $t \leq n$. The dealer can form the following system consisting of t equations and t unknowns $\{a_0, \dots, a_t\}$:

$$\begin{array}{ccccccccc} a_0 & + & a_1 x_1 & + & a_2 x_1^2 & + & \dots & + & a_t x_1^t & = & y_{x_1} \\ a_0 & + & a_1 x_2 & + & a_2 x_2^2 & + & \dots & + & a_t x_2^t & = & y_{x_2} \\ \vdots & & \vdots & & \vdots & & & & \vdots & & \vdots \\ a_0 & + & a_1 x_t & + & a_2 x_t^2 & + & \dots & + & a_t x_t^t & = & y_{x_t} \end{array}$$

Thus we can uniquely determine $\{a_1, \dots, a_t\}$ and let

$$f(x) = a_0 + a_1x + \dots + a_tx^t \pmod{p}.$$

The dealer can now broadcast the shares $\{f(t+1), \dots, f(2t)\}$. In this case any $C_{x_i} \in B$ can uniquely determine the broadcast message, since $\{f(t+1), \dots, f(2t)\}$ together with the shared key k_{x_i} , essentially gives $t+1$ shares to C_{x_i} . Additionally, any $y \in B$ can only get t shares and therefore $\{f(t+1), \dots, f(2t)\}$ will give them absolutely no information about the broadcast message a_0 .

In order to discuss the weaknesses of the SMBP(1) without loss of generality, we will use a small example of the SMBP(1) with $t=n=2$. Assume that dealer D wants to broadcast a secret a_0 to the parties C_1 and C_2 who share the keys k_1, k_2 with D respectively. Therefore, D has to solve the following system for a_1 and a_2 :

$$\begin{cases} a_0 + a_1 \times 1 + a_2 \times (1)^2 = k_1 \pmod{p} \\ a_0 + a_1 \times 2 + a_2 \times (2)^2 = k_2 \pmod{p} \end{cases} \quad (1)$$

Let $f(x) = a_0 + a_1x + a_2x^2 \pmod{p}$, then the dealer will publish $f(3)$ and $f(4)$ in order to securely broadcast the message a_0 . The system is insecure since both C_1 and C_2 can compute $k_2 = f(2)$ and $k_1 = f(1)$ respectively, once the dealer reveals the shares $f(3)$ and $f(4)$. Therefore, once C_1 and C_2 receive the message from the dealer, both C_1 and C_2 will know

k_1 and k_2 . This concludes that the SMBP(1) is limited to one-session message broadcasts, since their secret has been revealed.

SMBP(2)

The problem stated in SMBP(1) can be solved with the help of the 1-way hash function (definition 4.4). Let h be a public 1-way hash function which maps any arbitrary input to Z_p . The dealer now selects a random number r and solves for the following system:

$$\begin{cases} a_0 + a_1 \times 1 + a_2 \times (1)^2 = h(k_1, r) \pmod{p} \\ a_0 + a_1 \times 2 + a_2 \times (2)^2 = h(k_2, r) \pmod{p} \end{cases}$$

The dealer then sends $(r, f(3), f(4))$ to C_1 and C_2 . It is easy to see that the above system is immune from the insider attacks. First, the insider's attack allows C_1 to find $h(k_2, r)$ but it is infeasible for him to recover k_2 , assuming that inverting the function h is infeasible.

Assume that the dealer broadcast a message $a_{1,0}$ to C_1 and C_2 then broadcasts another message $a_{1,2}$ to C_2 and C_3 . Let $f_{m_1}(\cdot) = a_{1,0} + a_{1,1}x + a_{1,2}x^2 \pmod{p}$ and $f_{m_2}(\cdot) = a_{2,0} + a_{2,1}x + a_{2,2}x^2 \pmod{p}$ be the functions generated by the dealer in the first and second session respectively. If the same random value r is used in both broadcast

sessions, then C_1 can compute $T = h(k_2, r) = f_{m_1}(2)$. Now considering the session with respect to the function f_{m_2} , we have

$$a_{2,0} + a_{2,1}(2) + a_{2,2}(2)^2 = h(k_2, r) = T \pmod{p}.$$

In addition, recall that in order to broadcast the message $a_{2,0}$, the dealer must send $f_{m_2}(x_1)$ and $f_{m_2}(x_2)$ to C_1 and C_2 such that $\{x_1, x_2\} \neq \{2, 3\}$. Therefore, C_1 will obtain the following equations system:

$$\begin{cases} a_{2,0} + a_{2,1}(2) + a_{2,2}(2)^2 = T \pmod{p} \\ a_{2,0} + a_{2,1}x_1 + a_{2,2}x_1^2 = f_{m_2}(x_1) \pmod{p} \\ a_{2,0} + a_{2,1}x_2 + a_{2,2}x_2^2 = f_{m_2}(x_2) \pmod{p} \end{cases}$$

where T , $f_{m_2}(x_1)$ and $f_{m_2}(x_2)$ are already known by C_1 . Therefore C_1 is able to determine the broadcast message $a_{2,0}$, which the dealer wishes to securely broadcast to C_2 and C_3 . Therefore, we conclude that r can not be reused.

SMBP(3)

Assume the dealer wishes to broadcast a message a_0 to both C_1 and C_2 . Consider the case the if C_1 can intercept the dealer's message $(f(3), f(4))$, to C_2 by replacing it with some other message $(f'(3), f'(4))$. Therefore, C_2 will recover a message m' , rather than the

original message m sent from the dealer. The SMBP(2) allows C_1 to perform the above attack on any m' chosen by him since C_1 can find $h(k_2, r)$ by evaluating $f(2)$. Therefore he can generate a function f' (of degree 2) such that $f'(0)=m'$, $f(2)=h(k_2, r)$.

If $f(x)$ is obtained by the following equations system, then C_1 can only cause C_2 to receive a random message, rather than a chosen message.

$$\begin{cases} a_0 + a_1 \times h(k_1, r) + a_2 \times (h(k_1, r))^2 = k_1 \pmod{p} \\ a_0 + a_1 \times h(k_2, r) + a_2 \times (h(k_2, r))^2 = k_2 \pmod{p} \end{cases}$$

Now C_1 , by modifying $f(3)$ and $f(4)$, cannot modify a_0 in any predictable fashion. In addition, the above system may not have a unique solution since the hash function h may cause the system to be linear dependent. Therefore, in case the above system is linear dependent, then we must choose another random value r in order to generate the function f .

SMBP(4)

By combining SMBP(2) and SMBP(3), we can form the following system:

$$\begin{cases} a_0 + a_1 \times h(k_1, r) + a_2 \times (h(k_1, r))^2 = h(k_1, r) \pmod{p} \\ a_0 + a_1 \times h(k_2, r) + a_2 \times (h(k_2, r))^2 = h(k_2, r) \pmod{p} \end{cases}$$

$f(x)=a_0+a_1x+a_2x^2 \pmod{p}$, where a_0 is the message about to broadcast.

Bibliography

- [BCD90] J. Boyar, D. Chaum, I. Damgård, Convertible Undeniable Signatures, *Advances in Cryptology: CRYPTO '90*, Springer Verlag, 1990, pp 189-205.
- [BL90] J. Benaloh and J. Leichter. Generalized Secret Sharing and Monotone Functions. *Advances in Cryptology: CRYPTO '88*, Springer Verlag, 1988, pp 27-35.
- [CA89] David Chaum & Hans van Antwerpen, Undeniable Signatures, *Advances in Cryptology: CRYPTO '89*, Springer Verlag, 1990, pp 212-216.
- [Cha90] David Chaum, Zero-Knowledge Undeniable Signatures, *Advances in Cryptology: Eurocrypt '90*, Springer Verlag, 1990, pp 458-464.
- [Cha91] David Chaum, Some Weaknesses of "Weaknesses of Undeniable Signatures", *Advances in Cryptology: Eurocrypt '91*, Springer Verlag, 1991, pp 554-556.
- [Cha94] David Chaum, Designated Confirmer Signatures, *Advances in Cryptology: Eurocrypt '94*, 1994, Springer Verlag, 1994, pp 86-91.
- [CLR] T.H. Cormen, C.E. Leiserson and R.L. Rivest, *Introduction to Algorithms*, MIT Press, McGraw-Hill Book Co., 1989.
- [DH76] W. Diffie and M.E. Hellman. New directions in Cryptography. *IEEE Transactions on Information Theory*, 22 (1976), pp 644-654.

- [DP94] D.W.Davies and W.L.Price, Security for Computer Network, John Wiley & Sons, Inc., 1994.
- [DY91] Y.Desmedt, M.Yung, Weaknesses of Undeniable Signature Schemes, Advance in Cryptography: Eurocrypt '91, Springer Verlag, 1991, pp 205-220.
- [El85] T.ElGamal. A Public Key Cryptosystem and a Signature Scheme Based on Discrete Logarithms. IEEE Transactions on Information Theory, V31, 1985, 469-472.
- [GM84] S.Goldwasser and A.Micali. Probabilistic Encryption. Journal of Computer and Systems Science, V28, 1984, 270-299.
- [GMR88] S.Goldwasswr, S.Micali, and R.Rivest, A Digital Signature Scheme Secure Against Adaptive Chosen-Message Attacks, SIAM J. Computing 17(2), 1988, pp 281-301.
- [GMR89] S.Goldwasser, S.Micali and C.Rackoff. The Knowledge Complexity of Interactive Proof Systems. SIAM Journal of Computing, V18 ,1989, pp 186-208.
- [Gold88] Oded Goldreich, Randomness, Interactive Proofs, and Zero-Knowledge - A Survey, The Universal Turing Machine - A Half-Century Survey, Oxford University Press, 1988, pp 377-405.

- [Jak95]** M.Jakobsson, Blackmailing using undeniable signatures, Advances in Cryptology: Eurocrypt '94, Springer Verlag, 1994, pp 425-427.
- [JKK094]** Mike Just, Evangelos Kranakis, Danny Krizanc and Paul van Oorshot, On Key Distribution via True Broadcasting, 2nd ACM Conference on Computer and Communication Security, 1994, pp81-88.
- [Kra86]** E.Kranakis, Primality and Cryptography, John Wiley & Sons, 1986.
- [Li94]** Li Gong, New Protocols for Third-Party-Based Authentication and Secure Broadcast, 2nd ACM Conference on Computer and Communication Security, 1994, pp 176-183.
- [RSA78]** R.L.Rivest, A.Shamir, and L.Adleman. A Method for Obtaining Digital Signatures and Public Key Cryptosystems. Communications of the ACM, V21, 1978, pp 120-126.
- [Sch93]** Bruce Schneier, Applied Cryptography, John Wiley & Sons, Inc., 1993
- [Sha79]** A.Shamir, How to Share a Secret, Communication of ACM, 22(11), November 1979, pp 612-613.
- [Stin95]** D.R. Stinson, Cryptography:Theory and Practice, CRC Press, 1995.

END

2 1 - 1 1 - 9 6

FIN