

國立交通大學

資訊工程學系

博士論文

代理簽章具有前向安全與單次特性
及應用於公開金鑰建設

**On Proxy Signatures with Forward-Secure and One-time
Properties and their Applications in PKI**



研究生：張明信

指導教授：葉義雄 博士

中華民國 九十四 年 三 月

代理簽章具有前向簽章與單次特性及
應用於公開金鑰建設

**On Proxy Signatures with Forward-Secure and One-time
Properties and their Applications in PKI**

研究生：張明信 Student：Ming-Hsin Chang

指導教授：葉義雄 博士 Advisor：Dr. Yi-Shiung Yeh

國立交通大學



A Dissertation Submitted to the
Department of Computer Science and Information Engineering
College of Electrical Engineering and Computer Science
National Chiao Tung University
in partial Fulfillment of the Requirements
for the Degree of
Doctor of Philosophy
in

Computer Science and Information Engineering

Mar 2005

Hsinchu, Taiwan, Republic of China

中華民國 九十四 年 三 月

代理簽章具有前向安全與單次特性及應用於公開金鑰建設

學生：張明信

指導教授：葉義雄博士

國立交通大學資訊工程學系研究所博士班

摘 要

網際網路越來越多使用於商業上，安全之機制的實際應用，必須要從紙式的文件世界，改為電子文件世界，而電子文件之數位簽章提供了個人簽章的功能，但此簽章很少考慮到使用於代表機關之簽章或是代理簽章。而代理簽章產生主要目標是解決這一個問題，而且不洩漏代理人之私密訊息，且具有原簽章者之簽章權利。

實際上，已經有很多代理簽章方法之建議，但很多都無法運用於實際系統，因為提出建議方法時，作者證明其方法的安全性後，又常有其他的缺失被發現。此外，其建議方法無法使用於現行之簽章方法。而 DSA 和 ECDSA 為大家所熟知之簽章方法，具有安全性，所以我們建議代理簽章使用 DSA 的方法，使代理簽章成為可行的簽章機制。補足簽章者之驗證問題，在實用性上，更進一步，公開金鑰建設 (Public-Key Infrastructure, PKI) 是整合密碼學與憑證機構 (Certificate Authority, CA) 之整體全球網路安全，傳統之代理簽章機制，幾乎無法使用於 PKI 的架構中，我們依照 PKI 的特性設計新的流程，使代理簽章能使用於 PKI 的架構中，實際使用於應用系統，使代理簽章更能符合實用。

另一方面，我們發展了前向安全性代理簽章，它可以保證，至目前之簽章私密金鑰與資料，沒有被洩漏，保證之前資料之安全性。而且使用之方法必須簡

單，特別不需要散佈資料，或者是保護儲存機制，所以不增加金鑰管理的經費。其另外應用可以使用於代理簽章之簽章的時戳和代理簽章時限，簽章者需要使用當時之合法金鑰，即有隱性時戳之功能，簽章金鑰更新超出使用時限，限制了簽章者之代理功能。

單次簽章 (one-time signature) 方法的簽章和驗證優點是非常有效率，他們適用於低運算功能之晶片卡，而 Lamport 是最先發明的數位簽章是基於單向雜湊函數為基礎。如果簽章資料長度很大，然而 Lamport 的單次簽章方法需要很大之驗證資料與儲存空間，我們改進這大量公開金鑰，與簽章訊息之儲存空間問題。我們提出新的有效方法去簽發資料長度很大的資訊，我們也發展出單一代理簽章，使得 Lamport 的單次簽章變的可用於實際系統。



On Proxy Signatures with Forward-Secure and One-time Properties and their Applications in PKI

Student : Ming-Hsin Chang

Advisor : Yi-Shiung Yeh

Department of Computer Science and Information Engineering
College of Electrical Engineering and Computer Science
National Chiao Tung University

ABSTRACT

As the Internet is used more and more for business, security mechanisms in the electronic world are needed to replace established practice in the paper-based world. While basic digital signature schemes are able to provide most of the functionalities of personal signature, they are less than ideal for institutional purpose or a proxy purpose. A proxy signature scheme was introduced in order to solve this problem without revealing the secret information of a person who wants to delegate his digital signature signing power to someone else.

Actually, most of the proposed schemes are theoretical research, because the proxy schemes are not in practice on the field of cryptography. Digital signature Algorithm (DSA) and Elliptic curve DSA (ECDSA) which are pretty well known by their security properties. We develop a proxy signature based on the DSA in which leads the proxy signature scheme on applications in practice. Moreover, PKIs (Public-Key Infrastructures) integrate digital certificates, public cryptography, and certificate authorities into a total worldwide network security architecture. A typical

PKI is less effort for utility of proxy signature. We design a new procedure to make proxy signatures adopt for PKIs leading to the proxy signature more applicable in practical application.

On the other way, we develop a forward-secure proxy signature scheme. It guarantees that the secret key material at present (or up to date) does not compromise the secrecy of the earlier signature or encrypted material; and it also must be achieved in a simple way, in particular without requiring distribution or protected storage devices, and without increasing key management costs. The forward-secure proxy signature scheme also can be applied on proxy time limitation.

The advantages of the one-time signature generation and verification are very efficient and useful for chip cards where low computation complexity is required. Lamport first invented a one-time digital signature scheme based on one-way functions. However, the Lamport one-time scheme requires a large amount of space for storage of authentic information if a large number of messages are signed. We improve the Lamport one-time signature on the amount of storage space for public keys and signed message saving storage space and propose an efficient scheme to sign a long message. We also develop a one-time proxy signature scheme in which we make the Lamport one-time signature useful in practice.

誌 謝

這個學位是我畢生的榮耀，我將分享這一份榮耀給協助我的長官與親友，也感謝他們的付出與容忍。最感謝我的指導教授葉義雄老師，他親和的展現長者的風範，亦師亦友，在我修習學位期間，其學術與為人，均為我表率；也特別感謝以德同學，因我在職進修，無法事事照應，均由於他無私的協助，得以順利完成學位進修，是我畢生難得的益友。

感謝工作單位中華電信研究所，提供我進修的機會，尤其是鄭伯順副所長、張光耀主任、張耿豪經理與謝東明博士的提攜與關心，也是我進修動力的推手；工作期間，接受樊國楨博士、盧登臨博士、楊中皇博士等安全領域知識界之前輩引領，在此誠心感謝。

論文口試委員張真誠教授的深入指導與教誨是我爾後做學問與為人座右銘，其他賴溪松教授、謝續平教授、蔡錫鈞教授、雷欽隆教授與詹進科教授，對於我論文之指導，懇切的建議，也是我需隨時警惕自我的良言。

我也將我最多的榮耀分享給我妻子芬芳，與兩位兒子伯仲與伯瑞，我們沒有榮華富貴，但僅有這一點名位，聊表心意；也要誠心感謝我岳父與岳母，多年細微的照料我家，得以安心進修，也感謝我兄弟西韻、阿圖、銘章、有政、阿通的鼓勵，其餘無法一一陳述，在此一併致謝。

On Proxy Signatures with Forward-Secure and One-time Properties and their Application in PKI

Contents

中文摘要	II
ABSTRACT	IV
誌 謝	VI
LIST OF TABLES	XI
LIST OF FIGURES	XII
CHAPTER 1 INTRODUCTION	1
1.1. MOTIVATION	1
1.2. RELATED WORK	5
1.3. RESEARCH CONTRIBUTIONS	9
1.4. ABOUT THIS DISSERTATION	10
CHAPTER 2 PRELIMINARIES	11
2.1. DIGITAL SIGNATURE	11
2.2. PROXY SIGNATURE	13
2.2.1. <i>Proxy signature background</i>	13
2.2.2. PROXY SIGNATURE REQUIREMENTS	15
2.2.3. <i>Proxy signature model</i>	16

2.2.4. Proxy signature protocol.....	18
2.2.5. Strong proxy signature	20
2.3. FORWARD-SECURE SIGNATURE	23
2.3.1. Background of forward-secure proxy signature.....	23
2.3.2. Protocol of forward-secure signature	24
2.4. LAMPORT ONE-TIME SIGNATURE	26
CHAPTER 3 PROXY SIGNATURE IN PUBLIC-KEY	
INFRASTRUCTURES (PKIS).....	30
3.1 PROXY-PROTECTED SCHEME.....	30
3.2. SECURITY ANALYSIS AND PERFORMANCE	35
3.3. APPLICATION ON THE ECDSA.....	38
3.3.1. Proxy-protected ECDSA.....	38
3.3.2. Example demonstration	41
3.4. APPLICATIONS ON PUBLIC KEY INFRASTRUCTURES (PKIS).....	47
3.4.1. PKI architecture.....	47
3.4.2. PKIX components.....	47
3.4.3. PKIX management functions.....	48
3.4.4. Proxy signature applied in PKIs	51
CHAPTER 4 FORWARD-SECURE PROXY SIGNATURE	54
4.1. FORWARD-SECURE PROXY SIGNATURE SCHEME.....	55
4.1.1. Protocol of the forward-secure proxy signature scheme.....	56

4.1.2. <i>Correctness and conformance of the forward-secure proxy signature scheme</i>	61
4.1.3. <i>Security analysis of the forward-secure proxy signature scheme</i>	63
4.1.4. <i>Comparison</i>	64
4.2. A VARIANT FORWARD-SECURE PROXY SIGNATURE	65
4.2.1. <i>Protocol variant of forward-secure proxy signature</i>	66
4.2.2. <i>Correctness</i>	68
4.2.3. <i>Conformance with properties of proxy signature</i>	69
4.2.4. <i>Security analysis</i>	70
CHAPTER 5 ONE-TIME SIGNATURE AND ITS APPLICATION TO	
PROXY SIGNATURE	78
5.1. IMPROVING LAMPORT ONE-TIME SIGNATURE	78
5.1.1. <i>Protocol of variant Lamport one-time signature</i>	79
5.1.2. <i>Efficiency</i>	82
5.2. MORE IMPROVING ON LAMPORT ONE-TIME SIGNATURE	84
5.2.1 THE DEFINITION OF LAMPORT-T SCHEME	84
5.2.2. SECURITY ANALYSIS AND EFFICIENCY	88
5.2.3. COMPARISON.....	92
5.3. ONE-TIME SIGNATURE SCHEME APPLIED ON PROXY SIGNATURE	94
5.4. DISCUSSION	96
CHAPTER 6 CONCLUSION	98

6.1. CONCLUSION.....98

6.2. FUTURE WORK.....99

REFERENCES.....101



List of Tables

Table 3.1. The comparison of time complexity between proposed scheme and DSA.....	38
Table 3.2. Points on the elliptic curve $x^3 + x + 6 \pmod{11}$	46
Table 3.3. The multiples of generator G	46
Table 4.1. The comparison of proxy signature schemes.....	65
Table 5.1. Comparison of the variant Lamport one-time signature scheme and Lamport one-time signature scheme.....	84
Table 5.2. Comparison of the $L(1)$ and $L(t)$ with message n bits.....	87
Table 5.3. The iterations of the private keys of $L(5)$	91
Table 5.4. The comparison of variant $L(t)$ s with message 320 bits.....	92
Table 5.5. The comparison of Bos-Chaum scheme and $L(t)$ with the message 160 bits.....	93
Table 5.6. The comparison of one-time proxy signatures.....	97

List of Figures

Figure 3.1. The PKIX Architecture Model	50
Figure 3.2 Proxy signer initialization in PKI.....	53
Figure 3.3 Verification of proxy signature in PKI.....	53
Figure 4.1.1 The protocol of the forward-secure proxy signature scheme (1)	59
Figure 4.1.2 The protocol of the forward-secure proxy signature scheme (2)	60
Figure 4.2.1 The protocol of a variant forward-secure proxy signature (1)	72
Figure 4.2.2 The protocol of a variant forward-secure proxy signature (2)	73
Figure 4.3. The algorithm of forward-secure proxy key generation.....	74
Figure 4.4. The algorithm of forward-secure proxy acceptance.....	75
Figure 4.5 Forward-secure signature algorithm.....	76
Figure 4.6 Forward-secure verification algorithm.....	77

Chapter 1 Introduction

1.1. Motivation

Since the Internet pervasion on business, the security mechanisms in the electronic world are needed to replace in the paper-based world established practice. Therefore, digital signatures have been invented as a counterpart of handwritten signatures. Then, a digital signature not only provides the proof of authenticity of document and its originator as handwritten signature does, but digital signature properties are widely used in security mechanisms, such as integrity, authentication and non-repudiation.

While conventional digital signature schemes are able to provide most of the functionalities of personal signature, a conventional digital signature however is not suitable for some practical applications. They are less than ideal for institutional purpose or a proxy purpose. For institution, an institutional seal presents the institution and is used for signing on behalf of institution. A major difference between signing by hand and by seal is that seal is transferable such that the seal could reduce costs.

Take the following scenario for example. Suppose that a manager of a company goes on holiday. He may hand over his company seal to his deputy to sign on behalf of the company. Nevertheless, the signatures presented to customers are the same as before and they can verify the signatures using the same process, i.e. the customers of

a company are not affected. It will be economically infeasible for a big company to notify all its customers each time, when there is a personnel change in the company.

So far, there are still no straightforward methods of transferable seal proposed. A proxy signature scheme was introduced in order to solve this problem without revealing the secret information of a person who wants to delegate his digital signature signing power to someone else. The main idea of the proxy signature scheme is that: an original signer allows a designated person, called a proxy signer, to sign a message on behalf of original signer. In recent years, there are many papers proposed on this matter. Strong proxy signature [LK99,LKK₁01] is one of these papers that make the requirements of the proxy signature more complete.

Actually, most of the proposed schemes trend to theoretic approach and less consider in practice. When a new scheme is proposed, the authors always believe that their scheme will be sufficiently strong, secure, and unbreakable. In fact, all that the authors can do is to demonstrate the scheme's power against some known attacks; however, it occurs often that there will be always a new attack invented exactly against this scheme. Hence, a newly proposed scheme almost always suffers from some inborn weaknesses. To conquer this disadvantage, our proposed proxy signature scheme does not invent new signature schemes, but rather than combines existing mechanisms – Digital Signature Algorithm (DSA) and Elliptic curve DSA (ECDSA) [NIST00] which are pretty well-known by their security properties. Therefore, the proxy signature schemes based on the existence algorithms [LC03] are more

practicable. Even more the scheme can apply on the PKIs [CF⁺03,AF99,BPH02,ANSI99] in which the scheme can be pervade over Internet.

On the other way, although the security in many cryptographic techniques today, whether only available in the literature or used in practice, are believed to be considerably secure, if a secret information is revealed, either accidentally or via an attack. Security is often compromised not only for subsequent uses of the secret, but also for prior signed documents. That is, the greatest problem against the security of a digital signature scheme or a cryptographic method is exposure of the secret key. The problem is even worse especially in the open environments such as the Internet, where every computer node is a potential victim of hackers, because we cannot trust all signatures signed by this key even the signature was signed before key compromise. Once a hacker gets the secret key, he can create a signature and claim it was signed prior to the time he caught this key.

Take the following scenario for example. Suppose Alice is a notary public who has public key PK ; and uses normal signature scheme without forward security. On January 1st, 2002, a client Bob brings to Alice a document m and she notarizes the document by signing a signature δ . Bob expects to be able to use the document m for a long time. Unfortunately, Alice's secret key is compromised later, says January 1st, 2003. She discovers the fact and revokes her public key. Now, the notarized document m will no longer be accepted. The fact the m is dated "in the past" makes no difference, because everyone believes that Bob can produce a signature on m by

himself if Alice's secret key is no longer secure. This reduces the quality of the service which a notary Alice can provide.

To deal with this problem, several different approaches have been proposed. Many people attempt to lower the chance of exposure of secrets by distributing them across several systems, usually through secret-sharing method. Nevertheless, the cost of this method is usually extremely high; and as a matter of fact, is too expensive to be implemented by a typical individual user. What is more, since each of the systems may be susceptible to the same attack, the actual risk may not decrease. Other ways of protecting against key exposure include use of protected hardware or smartcard, but these are also costly and not suitable for ordinary people. The use of a trusted time stamping service applied to the signature to validate its date of creation is also a solution, but it needs extra resource to provide time issuing service.

Forward secure [AMN01,AR00,BM99,Kra00,BC⁺01] is a better way to reduce the damage. It guarantees that disclosure of the secret key material at present does not compromise the secrecy of the earlier signature or encrypted material; and it must be achieved in a simple way, in particular without requiring distribution or protected storage devices, and without increasing key management costs. We extend the concept to proxy signatures as forward-secure proxy signature [AR00] to make the system with a forward-secure property.

In recent years, one-time signature schemes [Lam79,Rab79,WS96,Mer87,Sch00] have attracted more and more attention, as an attractive alternative to the traditional

signature schemes based on public key cryptography. One of the main advantages of one-time signature schemes is their reliance on one-way functions that can be implemented using fast hash functions such that SHA-serials [NIST02]. The resulting signatures are the order of magnitude faster than signatures based on public cryptography applying on the resource-constrained, small devices, such as cell phones, pagers, smart cards etc. The other of advantage of such a scheme is that it is generally quite fast. However, the scheme tends unwieldy when used to authenticate multiple messages because additional data needs to be generates to both sign and verify each new message. By contrast, with conventional signature scheme like RSA [RSA78], the same key pair can be used to authenticate multiple documents, which will face the threat of replay attacks.

We propose a new scheme to generalize the Lamport one-time signature. Thus, the proposed scheme is a generalized Lamport one-time signature scheme and save the storage of the public key and the size of the signature. Moreover, we propose an efficient solution for signing a long message to make the proposed scheme more operative in practical. Moreover, we apply the proposed scheme to proxy signature.

1.2. Related Work

To facility delegation of capability in the electronic world, proxy signature

schemes have been proposed. Mambo, Usuda and Okamoto firstly proposed a proxy signature scheme (MUO scheme) [MUO₁₉₆, MUO₂₉₆] based on discrete logarithms [EIG85] for partial delegation of signing capacity. However, MUO scheme does not provide non-repudiation of proxy signatures [Zha97,Sun99,]. Non-repudiation means signature signers, both the original signer and proxy signers, cannot falsely deny later that he generated a signature. In practice, it is important, and sometimes necessary, to have the capability to know who is the actual signer of the proxy signature for auditing purpose or when there is abusing of signing capability. Thus some papers propose non-repudiable proxy signature scheme [HWW01,LHW98,LKK₂₀₁,Sun99, Zha97] which means the signature signers, both original and proxy signers, cannot disavow later that he generated a signature. This property is necessary in later proxy signature scheme.



In the mobile communication [LKK₃₀₁,KB⁺01,ZW⁺04], a proxy signature can be used into a mobile agent who can be applied in the electronic commerce. In a mobile agent system applying proxy signature, a customer, representing an original signer, generates delegation key pair and loads this key pair and other constraint requirements to the mobile agent. Mobile agents are autonomous software entities which are able to migrate across different execution environments. Mobility and autonomy make permanent connections unnecessary. So, mobile agents are suitable for heterogeneous environment. Non-repudiation property are considered in the mobile communication [ZW⁺04], so the original signer or proxy signer can falsely deny latter the fact that he

generated the signature. Therefore a dispute between the original signer and the proxy signer may be happened.

In some group-oriented application, it is often desire to share the signing capability among signers in a proxy group signer than one proxy signer or a group delegated by him can sign documents under the company's security policy. Thus someone proposes the multi-proxy signature schemes [CCH03,Son01,YBX00] and threshold proxy signature schemes [SLH99,Sun99,Zha97,KPW97] to solve problems. Threshold proxy signature scheme comes from the threshold cryptography. The idea about threshold cryptography is to protect information by fault-tolerantly distributing it among a cluster of cooperating computer and to diminish the risk attacking by adaptive attackers, who can corrupt parties' run protocols during any time in some run; and have the ability to integrate information comprised from different parties.

Some threshold proxy signature scheme haves the property of non-repudiation with known signers [Sun99,ElG85]. Thorough this property, a verifier not only can prove that proxy signature is valid but also can identify the actual proxy signer in the group who signs this proxy signature. As a result, the signer who did sign the message on behalf of the proxy group cannot deny their participation in signing the message.

In order to declare the valid delegation period, most proxy signature schemes use a warrant appearing in the signature verification equation. But the declaration in the warrant may be useless because the proxy signer can still create a proxy signature and claim that this signature was done during the delegation period even if the delegation

period has expired. Schemes of a time-stamped proxy signature with traceable receivers [Sun00,HS97] can make sure whether a proxy signature is created during the delegation period, and can trace the receivers who did receive the proxy signatures from the proxy signer.

Proxy signature schemes should be designed carefully for the proxy key pair not to be used for other purposes, such that the strong proxy signature [LK99,LKK₁01,LKK₂01,LKK₃01] is need for undeniability of an original signer. The strong proxy signature represents both original signer's and proxy signer's signatures. Once a proxy signer creates a valid proxy signature, he cannot repudiate his signature creation against anyone.

The elliptic curve cryptosystem (ECC) [Men93,MVS96] is constructed by integer points over elliptic curves in finite fields, The ECC can reach the same level of security of security constituted by DSA or RSA but provides greater efficiency than either discrete logarithm [LTH03] or factorization systems. Therefore, the proxy signatures based on elliptic curves are more efficient than on others.

Furthermore, there are other papers proposed variant proxy signature: such as blind proxy signature scheme [SH04,LA03,Cha83,MEE00] in which a proxy is able to make proxy blind signature which is able to verify in a way similar to proxy signature scheme. Generalizations of proxy signature [HTT04,LTH03] are proposed that can be applied to every proxy situation. The novel scheme allows the original group of original signers to delegate their signing capacity to a designed proxy group.

1.3. Research Contributions

Digital signature Algorithm (DSA) and Elliptic curve DSA (ECDSA) [NIST00] are pretty well known by their security properties to reach the properties of proxy signature. Our proposed proxy signature scheme combines existing mechanisms. We believe that their scheme will be sufficiently strong, secure, and unbreakable. Moreover, we develop a registration procedure in PKIs leading to proxy signature in practice.

Using proxy signature could make delegation of signing ability possible and forward secure property makes digital signature much more robust than common ones. Because proxy signature schemes involve in more participants, an original signer and a group of proxy signers, than ordinary signature scheme, it is required to make it stronger. Thus, it would be good to combine proxy signature and forward secure property to implicating to time limitation.

The Lamport one-time signature scheme is quite elegant, but it is not practical use. One problem is the size of the signature it produces. We propose a general Lamport one-time signature scheme called Lamport-t scheme in which the size of the signature and the public key are greatly reduced such that the Lamport one-time signature scheme are in practice. Moreover, we apply it to the proxy signature scheme as a one-time proxy signature scheme.

1.4. About this Dissertation

This dissertation firstly explains the scope of our dissertation. Then, we give some fundamental information about digital signature, (strong) proxy signature and forward secure property in chapter 2. In chapter 3, we propose a new proxy signature and apply it in PKIs. In chapter 4, we will provide a forward-secure proxy signature scheme and its applications on non-repudiation property. In chapter 5, we will present an enhanced one-time signature and its application on proxy signature. Finally, we will have a conclusion in chapter 6.



Chapter 2 Preliminaries

In this chapter, we briefly describe the necessary cryptographic systems in detail which are used in this dissertation. They include digital signature, proxy signature, one-time signature and forward secure property. In addition, we will mention some extended concept according to proxy signature; one-time signature and forward secure property. Based on those basic concepts, we will propose novel schemes or improve original schemes.



2.1. Digital Signature

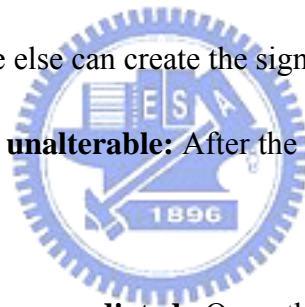
The concept of a digital signature was recognized several years before any practical approach was available. A digital signature is created to replace the real hand-written signature in the electronic world. A digital signature scheme is a method of signing a message stored in an electronic form. As such, a signed message can be transmitted over a computer network. The first method discovered was the RSA signature scheme [RSA78], which remains today one of the most practical and versatile techniques available. Subsequent research has resulted in many alternative digital signature techniques.

Specifically, a digital signature of a message is a number dependent on some

secret known only to the signer and the content of the message being signed. In practical implementations, we often combine one-way hash function with it to increase the efficiency.

A digital signature must have many characteristics. We list some of characteristics [Sch00] in the following:

- **The signature is authentic:** The signature convinces the document's recipient that the signer deliberately signed the document.
- **This signature is unforgeable:** The signature is proved that the signer did sign the document, and no one else can create the signature on behalf of this signer.
- **The signed document is unalterable:** After the document is signed, it cannot be altered.
- **The signature cannot be repudiated:** Once the signer signed a signature, he cannot later claim that he didn't sign it.
- **The signature must be verifiable:** If a dispute arises as to whether a party signed a document, an unbiased third party should be able to resolve the matter equitably, without requiring access to the signer's secret information (private key).



2.2. Proxy Signature

2.2.1. Proxy signature background

Proxy signatures are signature scheme that an original signer delegates his signing capability to a proxy signer, and then the proxy signer creates a digital proxy signature on behalf of the original signer.

According to proxy signature of Mambo *et al* [MUO₁96,MUO₂96], there are three types of delegation: full delegation, partial delegation and delegation by warrant. For the security consideration, full delegation is barely used.



(i) Full delegation: In full delegation, a proxy signer is given the same secret s that an original signer has. Because of full delegation, the proxy signature created by this proxy signer is indistinguishable from the signature created by the original signer.

(ii) Partial delegation: In partial delegation, a new secret σ is computed from the secret s of an original signer, and σ is given to a proxy signer in a secure way.

From security requirement s should not be computed from σ . Moreover, there are two types of signature scheme for partial delegation.

(1) Proxy-unprotected proxy signature: Besides this proxy signer, the original

signer also can create a valid proxy signature. But the third parties who are not designated as a proxy signer cannot create a valid proxy signature of the proxy signer.

- (2) Proxy-protected proxy signature: Only designated proxy signer can create a valid proxy signature for the original signer. The third parties and even the original signer cannot create a valid proxy signature of the proxy signer.

(iii). Delegation by warrant: This kind of delegation is implemented by using a warrant m_w [LK99,Neu93], which certifies that designated proxy signer is exactly the signer to be entrusted. Delegation by warrant is performed by the consecutive execution on signing of the public key signature scheme, which is time-consuming. But, it is appropriate for restricting documents to be signed, e.g. a warrant can state the valid time. In addition, there are two types of scheme for this approach.

- (1) Delegate proxy: In this type, an original signer signs a document, declaring some person, said Bob, is designated as proxy signer under the original signer's secret key by an ordinary signature scheme. The created warrant is given to Bob.
- (2) Bearer proxy: In this type, a warrant is composed of a message part and an original signer's signature for newly generated public key. The secret key for a newly generated public key is given to Bob in a secure way

In the partial delegation, we can classify proxy signature schemes into designed and non-designed proxy signature schemes according to whether the original signer designate a proxy signer in the proxy key generation phase.

- (1). Designated proxy signature: In this scheme original signer specifies the identity of a proxy signer as a form of warrant in proxy generation.
- (2). Non-designate proxy signature: In this scheme original signer does not satisfy a proxy signer in the proxy generation phase. Instead she can specify the set of allowed proxy signers of allowed message space.

2.2.2. Proxy signature requirements

The basic construction of [MUO₁96] and [MUO₂96] do not satisfy the strong undeniability property, i.e. the proxy signer can repudiate the fact of that he has created the proxy key pair does not contain any authentic information of the proxy signer. Although, they classify proxy signature schemes into strong and weak ones according to the undeniability property. Strong proxy signature represents both original signer's and proxy signer's signature, while weak proxy signature represent only original signer's signature.

There are some requirements with which strong proxy signatures must conform to verifiability, strong unforgeability, strong identifiability and strong undeniability [LK99,LKK₁01,MUO₁96,MUO₂96].

R1 - Verifiability: From a proxy signature a verifier can be convinced of the original signer's agreement on the signed message either by a self-authenticating form or by an interactive form.

R2 - Strong unforgeability: A designated proxy signer can create a valid proxy signature for the original signer. But the original signer and other third parties who are not designated as a proxy signer cannot create a valid proxy signature.

R3 - Strong identifiability: Anyone can determine the identity of the corresponding proxy signer from a proxy signature.

R4 - Strong undeniability: Once a proxy signer creates a valid proxy signature for an original signer, he cannot repudiate his signature creation against anyone.



(Note: The requirement R3 is an explicit authentication in which the authenticator can verify the identity of the proxy signer in this thesis.)

Besides, to avoid dispute, it is sometime necessary to identify the actual signer who generates the proxy signature. This property is called non-repudiation. Hence, a proxy signature scheme with non-repudiation property is a necessary property that we need.

2.2.3. Proxy signature model

A proxy signature scheme is a digital signature scheme. In addition, it must

conform to requirement **R1** to **R4** mentioned above.

In general, there are four phases in a proxy signature scheme: proxy generation and delivery phase, proxy verification and proxy key generation phase, proxy signature signing phase and verification of the proxy signature.

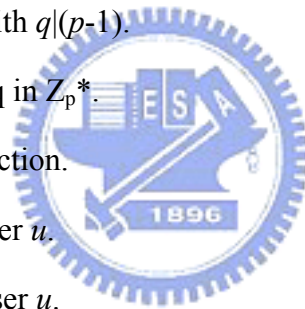
- (i). Proxy generation and delivery phase: An original signer generates the proxy secret and sends the proxy secret to a proxy signer in a secure way.
- (ii). Proxy verification and proxy key generation phase: The proxy signer checks whether the proxy secret really comes from the original signer. If the proxy-protected scheme is considered to be used, the proxy secret that original signer gives to proxy signer also needs to be alternated to proxy key otherwise the proxy secret is a proxy key.
- (iii). Proxy signature signing by the proxy signer phase: The proxy signer signs a proxy signature on document.
- (iv). Verification of the proxy signature phase: The receiver gets the proxy signature and verifies its accuracy.

A proxy signature scheme provided in this dissertation will also use these four phases. Besides, we will add another phase – “proxy key updating phase” introduced as our forward proxy signature scheme because of the property forward security required.

2.2.4. Proxy signature protocol

The basic of the MUO scheme [MUO₁96,MUO₂96] is a proxy-unprotected signature, which includes four phases - proxy generation and proxy delivery, proxy verification, proxy signature signing and verification of the proxy signature. The notations are as follows:

- O An original signer.
- P A proxy signer.
- V A verifier.
- p, q Two large primes with $q|(p-1)$.
- g A element of order q in Z_p^* .
- $h()$ A one-way hash function.
- x_u The secret key of user u .
- y_u The public key of user u .
- m A message to be signed.
- $A \rightarrow B$ A sends message to B



The original signer has a key pair, (x_A, y_A) . The scheme uses the following protocol.

(Proxy generation and delivery)

An original signer O selects random k_0 , computes K_A and sets the proxy key s_A . Then, he sends (s_A, K_A) to a designed proxy signer P in a secure way.

The scenario is showed as follows:

O : Select a random, k_0 ($1 \leq k_0 \leq p-1$).
 Compute $K_A = g^{k_0} \bmod p$.
 Set $s_A = x_A + k_0 K_A \bmod q$ as a proxy key.
 $O \rightarrow P (s_A, K_A)$ in a secure way.

(Proxy verification and proxy key generation)

The proxy signer checks the validation of (s_A, K_A) and set s_A as a proxy key.

P : Accept the delegation, if and only if $g^{s_A} = y_A K_A^{K_A} \bmod p$.

(Proxy signature signing)

The proxy signer, P , using the s_A as an alternative x_A , signs on the message m on behalf of the original signer, O . Then, P executes the ordinary signing operation $S(s_A, m)$, thus $(S(s_A, m), K_A)$ is a proxy signature.

P : Signs the message m , $S(s_A, m)$.

(Verification of the proxy signature)

The verification of the proxy signature is similar to the verification of ordinary signature by executing the verification $V(S(s_A, m), y_A K_A^{K_A})$.

V : Compute the original's public key $y_A K_A^{K_A}$, where $g^{s_A} = y_A K_A^{K_A}$.

Execute $V(S(s_A, m), y_A K_A^{K_A})$.

In the proxy-unprotected scheme, the original signer can sign a proxy signature as

the proxy signs such that the original signer can forge a proxy signature.

2.2.5. Strong proxy signature

Lee *et al* [LK99] first proposed the concept of the strong proxy signature. In their consideration, four basic requirements for **R1** to **R4**, i.e. verifiability, strong unforgeability, strong identifiability and strong identifiability undeniability, are not enough. This is because a proxy signer may maliciously sign documents or even give his proxy key pair to other people. The strong proxy signature needs to add a requirement as follows:



R5 - Prevention of misuse: it should be confident that proxy key should be used only for creating proxy signature conforming to delegation information (ex: some conditions specified in m_w). Proxy key pair cannot be used for other purposes. In case of any misuse of proxy key pair, the responsibility of proxy signer should be determined explicitly.

In the strong proxy, once a proxy signer creates a valid proxy signature, he cannot repudiate his signature creation against anyone. If a proxy signer creates a signature conforming to m_w , then the original signer is responsible for it, too. Namely, the original signer is responsible only for m_w and the proxy signature signer is responsible

for m .

Strong signature scheme in LK99 is discrete-logarithms based as well. Therefore, the mathematic model and symbols are similar to those in section 2.2.4. (x_A, y_A) and (x_B, y_B) are original signer's key pair and proxy signer's key pair respectively. In addition, it was proposed by using partial delegation with warrant proxy signature scheme. Hence, a warrant m_w is also appearing in this scheme.

The strong proxy signature scheme proposed by Lee *et al* [LK99] is as follows:

(Proxy generation and delivery)

An original signer 'O' generates a random number k_A . After that, she computes $r_A \equiv g^{k_A} \pmod{p}$ and $s_A \equiv x_A h(m_w, r_A) + k_A \pmod{p-1}$. The warrant m_w should state application-dependent delegation information clearly such as the qualification of the proxy signer and allowed message content. Then, the original signer gives (r_A, s_A, m_w) to a proxy signer 'P' secretly.

O: Select a random, k_A ($1 \leq k_A \leq p-1$).

Compute $r_A = g^{k_A} \pmod{p}$.

Set $s_A \equiv x_A h(m_w, r_A) + k_A$ as a proxy key.

$O \rightarrow P(r_A, s_A, m_w)$ in a secure way.

(Proxy verification and proxy key generation)

Bob checks the validity of the following equation:

$$g^{s_A} \equiv r_A y_A^{h(m_w, r_A)} \pmod{p}.$$

If the checking passes, the proxy signer uses s_A to generate his own proxy key as

$$x_p \equiv s_A + x_B \pmod{q}.$$

and the implicitly public key is

$$y_p \equiv g^{x_p} \equiv y_A^{h(m_w, r_A)} y_B r_A \pmod{p}.$$

As the proxy signer qualification is stated in m_w explicitly, (r_A, s_A, m_w) can be exposed to a set of possible proxy signers in the proxy delivery stage. Only a qualified person can be a proxy signer.

(Proxy signature signing)

If a document m conforms to the message qualification stated in m_w , the proxy signer can use the x_p as a private key to create a signature ξ on behalf of the original signer. Then, $(m, \xi, m_w, y_A, y_B, r_A)$ is a valid proxy signature.



(Verification of the proxy signature)

First, a verifier computes the proxy public key as $y_p' \equiv y_A^{h(m_w, r_A)} y_B r_A$ using (m_w, y_A, y_B, r_A) . Using y_p' , the verifier verifies the validity of the proxy signature as $V(m, \xi, y_p') \stackrel{?}{=} \text{true}$.

Right after that, the verifier checks the conformance of proxy signature to the warrant m_w . Both the proxy signer Bob and message m_p should be qualified by m_w .

2.3. Forward-Secure Signature

Ross Andersen suggested the idea of a digital scheme with forward security in an invited lecture at the ACM CCS conference [And99]. The term “forward secrecy” was first used in the context of session key exchange protocols by Bellare and Miner [BM99]. The basic idea is that compromise of long-term keys does not compromise past session key, meaning that past actions are protected in some way against the loss of the current key. Furthermore, the core of forward secure digital signature schemes is the key updating method.



2.3.1. Background of forward-secure proxy signature

At the first, a user registers a public key PK and keeps private the corresponding secret key, which denotes SK_0 . The index “0” means a base secret key. The total time T is divided into t periods. While the public key is keeping the same in the whole periods of the total time T , the user evolves the secret key with time period t . When a period i begins, the user applies a function with an input SK_{i-1} , the secret key at last period, to generate SK_i and right after that the user deletes the previous secret key SK_{i-1} .

The function using in updating should be one-way function, whose feature is that an output is to compute from an input and inversely; and almost impossible to

calculate the input from this output without any additional information. Hence, the user can produce signatures using a different signing key, i.e. SK_i in period i .

The public key stays fixed throughout, so that the signature verification process is unchanged. In addition, the public key certification and management processes are unaltered, too.

2.3.2. Protocol of forward-secure signature

A forward-secure digital signature scheme is a kind of digital signature schemes and contains four phases, a key update phase, a key generation phase, a signature signing phase and a verification phase. A key update phase is a concept of key-revolution schemes to create a new key for the current period in which the duration of operation is divided into several periods with a different secret key for each period. We apply the Abdalla-Reyzin forward-secure digital signature scheme [AR00] to our proposed scheme, so we briefly describe the Abdalla-Reyzin forward-secure digital signature scheme firstly.

Let p_1 and p_2 be two primes of approximately equal size with $p_1 = p_2 = 3 \pmod{4}$ and $N = p_1 p_2$. The number N is a k -bits integer called *Blum* integer [Sti02]. The parameter v is a secure parameter. Assume that the valid duration for signature is divided into several periods, numbered $1, \dots, t$. The function h is an one-way hash function. An Abdalla-Reyzin forward-secure digital signature scheme includes four

phases - key generation, key update, signature signing, and verification. We described as follows:

(Key generation)

To generate a key pair, a signer should do the following step:

Step 1. Select a random $s_0 \in Z_N^*$ as a private key

Step 2. Compute a corresponding public key $u = 1/s_0^{v(t+1)} \bmod N$.

The original signer's key pair is (s_0, u) . The suffix '0' of private key s_0 indicates the basis state of a key-revolution scheme.



(Key update)

At current period j , the signer needs to update the private from s_{j-1} to s_j by $s_j = (s_{j-1})^{2^v} \bmod N$. Note that the suffix denotes the current period.

(Signature signing)

At the current period j , the signer want selects to sign a message M and execute the following steps:

Step 1. Select a random $k \in Z_N^*$.

Step 2. Compute $r = k^{2^{v(t+1-j)}} \bmod N$ and $e = h(j, r, M)$.

Step 3. Compute $\sigma = ks_j^e$.

Therefore, the signature on message M is (j, σ, e) .

(Verification)

A verifier can verify the validation of (j, σ, e) on message M to execute the following steps:

Step 1. If $\sigma = 0$ then reject the signature.

Step 2. Compute $r' = \sigma^{2^{v(t+1-j)}} u^e \pmod N$.

Step 3. If $e = h(j, r', M)$ then accept the signature, else reject it.

For fitting with the proposed scheme, we slightly modify the output parameter of signature algorithm and the input of verification algorithm.

2.4. Lamport One-Time Signature



One-time signature schemes were first proposed by Rabin [Rab79] and Lamport [Lam79] and based on the idea of committing public keys to secure keys using one-way functions. For more 25 years, Lamport one-time signature schemes have been proposed and investigated by many researchers. Indeed, one-time signature schemes have found many interesting applications, including on-line/off-line signatures, digital signatures with forward security properties, broadcast authentication protocols and proxy signatures etc.

In recent years, one-time signature schemes have attracted more and more attention, as an attractive alternative to the traditional signature schemes based on public key cryptography. One of the main advantages of one-time signature schemes

is their reliance on one-way functions that can be implemented using fast hash functions. The resulting signatures are the order of magnitude faster than signatures based on public cryptography applying on the resource-constrained, small devices, such as cellular phones, pagers, smart cards etc. The other of advantage of such a scheme is that it is generally quire fast. However, the scheme tends unwieldy when used to authenticate multiple messages because additional data needs to be generates to both sign and verify each new message. By contrast, with conventional signature schemes like RSA [RSA78], the same key pair can be used to authenticate multiple documents, which will face the threat of replay attacks.

In this section, we briefly review the Lamport one-time signature, which includes three algorithms- key generation, signature signing and verification. Suppose that $h : Y \rightarrow Z$ is a one-way hash function.



(Key generation)

Step 1. Select $2k$ elements $y_{i,j} \in Y$ at random with $1 \leq i \leq k$ and $j = 1,0$

where k is the length of message based on 2.

Step 2. Compute $z_{i,j} = h(y_{i,j})$ for all i, j .

Step 3. The key K consists of the $2k$ y 's and $2k$ z 's. The private key SK box

and the public key PK box are as follows:

$$SK = \begin{pmatrix} y_{1,0} & y_{2,0} & \dots & y_{k,0} \\ y_{1,1} & y_{2,1} & \dots & y_{k,1} \end{pmatrix}$$

$$PK = \begin{pmatrix} z_{1,0} & z_{2,0} & \dots & z_{k,0} \\ z_{1,1} & z_{2,1} & \dots & z_{k,1} \end{pmatrix}$$

(Signature signing)

To sign a k -bit message $m = m_1 \dots m_k$, we should do the following steps:

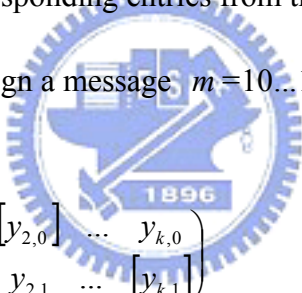
Step 1. The corresponding entries of the message $m_1 \dots m_k$ are $y_{1,m_1}, \dots, y_{k,m_k}$.

Step 2. We define the signature

$$sig(m_1 \dots m_k) = (y_{1,m_1}, \dots, y_{k,m_k}).$$

Step 3. We just select corresponding entries from the key box to create signature.

For example, we want to sign a message $m = 10 \dots 1$. The signature is



$$\begin{aligned} sig(m_1, \dots, m_k) &= \begin{pmatrix} y_{1,0} & [y_{2,0}] & \dots & y_{k,0} \\ [y_{1,1}] & y_{2,1} & \dots & [y_{k,1}] \end{pmatrix} \\ &= (y_{1,1} \quad y_{2,0} \quad \dots \quad y_{k,1}) \end{aligned}$$

on message $m_1 \dots m_k$.

(Verification)

To verify signature $(y_{1,1} \quad y_{2,0} \quad \dots \quad y_{k,1})$ on message $m_1 \dots m_k$, we check if

$$h(m_i) = y_{i,m_i} \text{ for } 1 \leq i \leq k \text{ holds.}$$

If it holds accept the signature, or reject it.

A message to be signed is a binary k -tuple. Each bit selects the corresponding value in the SK box as signed value. If the i^{th} message bit is m_i , the signature is y_{i,m_i} in the SK box. To verify the signature, we just check the hash value of each element is the corresponding value in the box.



Chapter 3 Proxy Signature in Public-Key Infrastructures (PKIs)

The notations are defined the same as in section 2.2.4. The proposed scheme is based on the proxy-protected approach. Only the proxy signer can create the proxy signature. Although the proxy-unprotected scheme is more efficient than the proxy-protected one, the proxy-unprotected scheme is only applicable when the original signer is honest. That means the proxy-protected schemes have also the ability to prevent cheating attempts plotted by the original signer, needless to mention about malicious proxy signers. We develop a proxy-protected scheme, which could be combined with the DSA (ECDSA) and it could be applied on Public Key Infrastructures (PKIs).

3.1 Proxy-Protected Scheme

There are four steps, proxy generation and delivery, proxy verification and proxy key generation, signing by the proxy signer and verification of the proxy signature in proxy-protected scheme. Let the original signer 'O' has key pair (x_A, y_A) where $y_A = g^{x_A} \bmod p$ and we describe above steps as follows:

(Proxy generation and delivery)

The proxy signer ' P ' selects a random k_0 , computes g' and sends it to the original signer. On receiving g' , the original signer creates s_A and sends (r_A, s_A) . The parameters g' and r_A are public information. The protocol is showed as follows:

P : Select a random, k_0 ($1 \leq k_0 \leq p-1$).

Compute $g' = g^{k_0} \bmod p$.

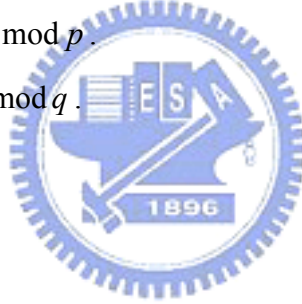
$P \rightarrow O$ g'

O : Compute $k_A \in_R Z_q^*$ and $r_A = g^{k_A} \bmod p$.

Compute $e = h(g^{k_A}) \bmod p$.

Set $s_A = (x_A e + k_A) \bmod q$.

$O \rightarrow P$ (r_A, s_A) .



(Proxy verification and proxy key generation)

The proxy signer checks the validity of (r_A, s_A) and computes a proxy key s_B .

The protocol is as follows:

P : Receive (r_A, s_A) from original signer

Check $r_A = g^{s_A} y_A^{-e'} \bmod p$ where $e' = h(r_A^{k_0})$.

If it holds process the follows, else reject it.

Computes $s_B = s_A k_0^{-1} \bmod q$.

Thus, the proxy key is s_B .

(Proxy signature signing)

The original signer signs on the message m as the DSA algorithm using the proxy key s_B . The protocol is follows:

P : To sign on message m , first compute $h(m)$.

Then select a random $k \in Z_q^*$.

Compute $r = (g^{tk} \bmod p) \bmod q$

Set $s = k^{-1}(h(m) + s_B r) \bmod q$.

This signing step is similar to the DSA scheme; and the proxy signature is the tuple (g', r_A, e', r, s) .



(Verification of the proxy signature)

To verify the proxy signature (g', r_A, e', r, s) on message m , a verifier should do the following steps:

V : Verify that $1 \leq r \leq q$ and $1 \leq s \leq q$; if not, then reject the signature.

Compute $w = s^{-1} \bmod q$.

Compute $u_1 = w \cdot h(m) \bmod q$, $u_2 = r w \bmod q$, and $u_3 = e' u_2 \bmod q$.

Compute $v = (g'^{u_1} r_A^{u_2} y_A^{u_3} \bmod p) \bmod q$.

Accept the signature if and only if $v = r$.

To verify the proxy signature (g', r_A, e', r, s) on message m , a verifier checks whether $v = r$, where $v = (g'^{u_1} r_A^{u_2} y_A^{u_3} \bmod p) \bmod q$.

In order to prove that the proposed scheme works correctly and explain that the proposed conforms to the requirements of the proxy signature schemes, there are two theorems as the follows:

Theorem 3.1: *If the proxy secret (r_A, s_A) is constructed correctly, then it will pass the verification by using $r_A = g^{s_A} y_A^{-e'}$ mod p .*

Proof:

Suppose the proxy secret (r_A, s_A) and $g' = g^{k_0}$ mod p is correct. We have

$$s_A = (x_A e + k_A) \text{ mod } q.$$

Then make the substitutions

$$e = h(g^{k_A}) = h((g^{k_0})^{k_A}) = h((g^{k_A})^{k_0}) = h(r_A^{k_0}) = e' \text{ mod } p.$$

We obtain the following:

$$s_A = (x_A e' + k_A) \text{ mod } q.$$

Rearrange the above equation

$$k_A = (s_A - x_A e') \text{ mod } q.$$

Raise both sides by g

$$g^{k_A} = g^{(s_A - x_A e')} \text{ mod } p,$$

$$r_A = (g^{s_A} \cdot g^{-x_A e'}) \text{ mod } p (\because r_A = g^{k_A} \text{ mod } p)$$

$$r_A = (g^{s_A} y_A^{-e'}) \text{ mod } p (\because y_A = g^{x_A} \text{ mod } p)$$

Thus, $r_A = (g^{s_A} y_A^{-e'}) \text{ mod } p$ as required. □

Suppose the proxy signer receives a proxy secret from the original signer correctly in the proxy key generation. The proxy signer cannot forge another proxy secret to

create a proxy key, because it is computationally infeasible to select another r_A to create a valid tuple of proxy secret. Moreover, the original signer also cannot forge the proxy key, because the generator is blinded by a factor of k_0 which is only known by the proxy signer. Thus, only the designed signer can create the valid proxy key. Therefore, the proposed scheme conforms to the property of *unforgeability*.

Theorem 3.2: *If the proxy signature is generated by the proxy signer correctly in the proposed scheme, then it will pass the proxy signature verification.*

Proof:

Suppose the proxy signature is correct. It implies that the delegation certification is correct such that we have a valid proxy signature

$$s = k^{-1} (h(m) + s_B r) \pmod q.$$

Rearrange the signature

$$k = s^{-1} (h(m) + s_B r) \pmod q$$

Substitute s_B

$$k = s^{-1} (h(m) + s_A k_0^{-1} r) \pmod q. (\because s_B = s_A k_0^{-1} \pmod q)$$

Substitute s_A

$$k = s^{-1} [h(m) + (x_A e + k_A) k_0^{-1} r] \pmod q. (\because s_A = (x_A e + k_A) \pmod q)$$

Raise both sides by g^k

$$g^{k^k} = (g^{s^{-1}h(m)} g^{k_A k_0^{-1} r s^{-1}} g^{x_A e k_0^{-1} r s^{-1}} \pmod p) \pmod q.$$

Substitute g^{k^k} by r , $g^{k_A k_0^{-1}}$ by r_A and $g^{x_A k_0^{-1}}$ by y_A

$$r = (g^{s^{-1}h(m)} r_A^{rs^{-1}} y_A^{ers^{-1}} \bmod p) \bmod q.$$

Let $w = s^{-1} \bmod q$, $u_1 = w \cdot h(m) \bmod q$, $u_2 = rw \bmod q$, and $u_3 = e'u_2 \bmod q$.

We yield

$$r = (g^{u_1} r_A^{u_2} y_A^{u_3} \bmod p) \bmod q \text{ as required.} \quad \square$$

The proxy signer uses the proxy key to sign on a document, but a verifier need to use the original signer's public key to verify the validation of the signature. The proxy key is created interactively by original and proxy signer such that from the signature, a verifier can be aware of the original signer agrees proxy signer on signing the message. This property is verifiability. From the theorems, the proposed scheme conforms to the proxy signature requirements from R1 to R4. For adapting to the DSA, We could not add any warrant information in the proposed scheme.

3.2. Security Analysis and Performance

The security of the proposed scheme is based on the difficulty of breaking the one-way hash function and the hardness of the discrete logarithm problem [MVS96]. In this section, we discussion some possible security attacks against the proposed scheme. We will explain that the proposed scheme can prevent from those attacks.

Attack1: An attacker may forge the proxy signature on the message m by selecting a random k and computing $r = g^{tk} \bmod p$.

Analysis of Attack1: The attacker needs to create a forgery signature $s = k^{-1}(h(m) + s_B r) \bmod q$. Because the proxy key is unknown by the attacker, it is computationally infeasible to determine s under the assumption of the discrete logarithm problem. The success probability is only $1/q$. However, it is negligible for large q .

Attack2: An attack might attempt to forge the proxy key s_B to create a proxy signature.

Analysis of Attack2: The attacker will face the discrete logarithm problem too. To solve s_B in $s_B = k_0^{-1}(x_A e + k_A) \bmod q$. It is still computationally infeasible under the assumption of the discrete logarithm problem.

Attack3: A malicious original signer attempt to forge the proxy signer to derive a valid proxy key.

Analysis of attack3: The proxy signer uses a blind factor k_0 to blind the generator g by $g' = g^{k_0} \bmod p$ [Sch95]. The original signer needs to solve k_0 from $g' = g^{k_0} \bmod p$. It is difficult to determine k_0 based on the hardness of the discrete logarithm problem. For the security reason, an original signer cannot derive the designed proxy signer's proxy key; otherwise the proxy signature cannot be

distinguished from the original signer and the proxy signer who create it. It is a 'proxy-protected' property.

Attack4: The malicious proxy signer attempts to impersonate an original signer to create a proxy secret.

Analysis of attack4: Firstly, the malicious proxy signer is computationally infeasible to create a random k_A from $r_A = g^{k_A} \pmod p$. Secondly, to solve e' from $r_A = g^{s_A} y_A^{-e'} \pmod p$ by knowing g and r_A is computationally infeasible.

In the propose scheme, the size of q is at 160 bits and the size of p is between 512 and 1024 bits. For the security reason, a 512-bit prime provides marginal security such that at least 768 bits is commended. Suppose p is a 768-bit integer and one modular exponentiation takes on 240 modular multiplications. In the Table 1, we compare the time complexity between the proposed scheme and the DSA. The major portion of time complexity is modular multiplications and modular inverses, thus we neglect the time complexity of hash function and modular additions. In the propose scheme, the time complexity of the proxy signature is the same as the DSA; while the time complexity of the proxy signature verification requires one modular exponentiation and two modular multiplications more than the DSA.

Table 3.1. The comparison of time complexity between proposed scheme and DSA

Scheme ↓	Proxy generation	Proxy verification	Signature	Verification
The proposed scheme	$241T_m$	$721T_m$	$242T_m + T_{inv}$	$725T_m + T_{inv}$
DSA	NA	NA	$242T_m + T_{inv}$	$483T_m + T_{inv}$

Note:

T_m : The number of modular multiplications.

T_{inv} : The number of modular inverse with 160-bit modulus.

3.3. Application on the ECDSA

In 2000, the ECDSA was approved as FIPS 186-2 [NIST00]. We apply the propose scheme to the ECDSA, called a proxy-protected ECDSA, which is a variant ECDSA with properties of proxy signatures.

3.3.1. Proxy-protected ECDSA

The parameters are defined in an elliptic curve E modulo a prime p as a public-key cryptography. The notations are follows:

Alice An original signer.

Bob A proxy signer.

Carol A verifier for proxy signature

p	A prime number.
E	An elliptic curve defined over F_p .
q	The number of points on E .
G	A point on E having prime order q .
x	A private key with $0 \leq x \leq q-1$.
Q	A public key with $Q = xG$ on E .
$h()$	An one-way hash function, SHA-1.

The original signer Alice has private key x and public key $Q = xG$ certificated by a certificate authority. Bob is a designated proxy signer. The protocol of proxy-protected ECDSA we describe as follows:

(Proxy generation and delivery)

Bob: Select a random, k_0 ($1 \leq k_0 \leq q-1$).

Compute $G' = k_0 G \pmod{q}$.

Bob → *Alice* G' .

Alice: Select a random integer, k_A ($1 \leq k_A \leq q-1$)

Compute $R_A = k_A G$.

Set $(x_1, y_1) = k_A G'$.

Compute $e = h(x_1)$ and set $s_A = (xe + k_A) \pmod{q}$.

Alice → *Bob* (R_A, s_A) .

(Proxy verification and proxy key generation)

Bob: Set $r_A = x_2$, where $(x_2, y_2) = R_A$.

Compute $e' = h(k_0 r_A) \bmod q$.

Compute $(x_2', y_2') = s_A G - e' Q$.

Accept the delegation if and only if $r_A = x_2'$.

Then, compute $s_B = s_A k_0^{-1} \bmod q$ as a proxy key.

(Proxy signature signing)

Bob: Select a random k ($1 \leq k < q - 1$).

Compute $(x_3, y_3) = kG'$.

Set $r = x_3$.

Compute $s = k^{-1}(h(m) + s_B r) \bmod q$.

If $r = 0$ or $s = 0$ then re-select a random k and run again.

The proxy signature for the message m is (G', R_A, e', r, s) .

(Verification of the proxy signature)

Carol: Verify that r and s are integers in interval $[1, q - 1)$.

Compute $w = s^{-1} \bmod q$.

Compute $u_1 = h(m)w \bmod q$.

Compute $u_2 = rw \bmod q$.

Compute $u_3 = e' u_2 \bmod q$.

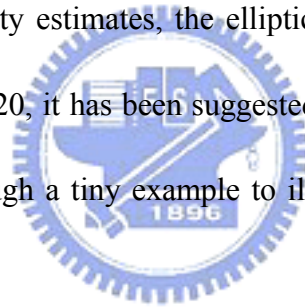
Compute $X = (x_3', y_3') = u_1 G' + u_2 R_A + u_3 Q$.

If $X = O$, then reject the signature, else accepts the signature if and only if $x_3' = x_3 = r$.

The proxy-protected ECDSA could be also deployed in ECDSA by taking parameters $G' = G$, $R_A = 0$ and $e' = 1$. Furthermore, the proxy-protected ECDSA also reaches the properties of strong proxy signature.

3.3.2. Example demonstration [Sti02]

In some reports on security estimates, the elliptic curve basing on cryptosystem will be secure till the year 2020, it has been suggested that one should take $p \approx 2^{160}$. In this section we work through a tiny example to illustrate the computations in the proxy-protected ECDSA.



Let E be the elliptic curve $y^2 = x^3 + x + 6$ over Z_{11} . The parameter q is the number of points in E . We first compute $x^3 + x + 6 \pmod{11}$ for $x \in Z_{11}$, and then try to solve the above equation for y . We can set

$$z = x^3 + x + 6 \pmod{11}$$

and test if z is a quadratic residue, or QR , by applying Euler's criterion.

If the modulo prime $p = 3 \pmod{4}$, we could yield the square roots of a quadratic residue z as following formula:

$$\pm z^{(11+1)/4} \pmod{11} = \pm z^3 \pmod{11}.$$

The results of the computing are listed in Table 3.2.

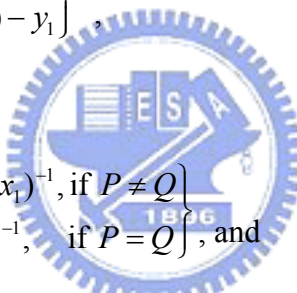
Because G is a generator, we can take the generator $G = (2,7)$; and compute the remaining multiples of G by applying the addition operation on E .

The addition operation on E is defined as follows:

Suppose $P_1 = (x_1, y_1)$, $P_2 = (x_2, y_2)$ are the point on E . If $x_2 = x_1$ and $y_2 = -y_1$, then $P_1 + P_2 = O$ where O is a special point, called point at infinity; otherwise

$P_1 + P_2 = (x_3, y_3)$, where

$$\begin{cases} x_3 = \lambda^2 - x_1 - x_2 \\ y_3 = \lambda(x_1 - x_3) - y_1 \end{cases},$$

$$\lambda = \begin{cases} (y_2 - y_1)(x_2 - x_1)^{-1}, & \text{if } P \neq Q \\ (3x_1^2 + a)(2y_1)^{-1}, & \text{if } P = Q \end{cases}, \text{ and}$$


a is in the elliptic curve $y^2 = x^3 + ax + b$ over Z_p such that $a = 1$.

Therefore, the next multiple is $2G = G + G$, $3G = 3G + G$, and so on. The results of these computations are tabulated in Table 3.3 [Sti02]. Suppose that Alice's private key is 3, so the public key is $Q = 3G = (8,3)$.

The proxy protocol is as follows:

(Proxy generation and delivery)

Bob : Select a random k_0 , said5;

and compute $G' = k_0 G = 5G = (3,6)$

Bob → Alice G'

Alice: Select a random k_A , said 4;

and compute $R_A = k_A G = 4G = (10,2)$;

set $k_A G' = 4(3,6) = (7,2) = (x_1, y_1)$.

Suppose that $e = h(7) = 5$. Alice computes

$$s_A = (xe + k_A) \bmod q = (3 * 5 + 4) \bmod 13 = 6 \text{ and}$$

forward $(R_A, s_A) = [(10,2), 6]$ to Bob.

(Proxy verification and proxy key generation)

Let $r_A = x_2 = 10$ where $(x_2, y_2) = R_A = (10,2)$. Then, Bob computes

$$e' = h(k_0 R_A) = h(7) = 5.$$

and accepts the delegation if $x_2' = 10 = x_2$ where

$$(x_2', y_2') = s_A G - e' Q = 6 * (2,7) - 5 * (8,3) = (10,2)$$

The proxy key is

$$s_B = s_A k_0^{-1} \bmod q = 6 * 5^{-1} \bmod 13 = 9.$$

(Proxy signature signing)

Suppose that the message is m , $h(m) = 8$ and $k = 9$. To sign the message, Bob computes

$$(x_3, y_3) = kG' = 9 * (3,6) = (7,9),$$

sets $r = x_3 = 7$ and creates proxy signature,

$$s = k^{-1}(h(m) + s_B r) \bmod q = 9^{-1}(8 + 9 * 7) \bmod 13 = 5.$$

The proxy signature is $(G', R, e', r, s) = [(3,6), (10,2), 5, 7, 5]$.

(Verification of the proxy signature)

The verifier does the follows:

$$w = s^{-1} \bmod q = 5^{-1} \bmod 13 = 8,$$

$$u_1 = h(m)w \bmod q = 8 * 8 \bmod 13 = 12,$$

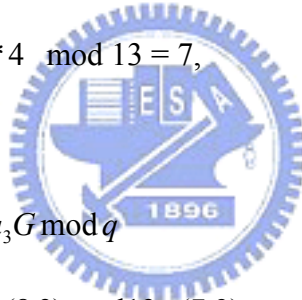
$$u_2 = rw \bmod q = 7 * 8 \bmod 13 = 4,$$

$$u_3 = e' u_2 \bmod q = 5 * 4 \bmod 13 = 7,$$

and

$$X = (x_1, y_1) = u_1 G' + u_2 R + u_3 G \bmod q$$

$$= 12 * (3,6) + 4 * (10,2) + 7 * (8,3) \bmod 13 = (7,9).$$



The verifier accepts the signature, because $x_1 = 7$. This example adequately shows the proxy-protected ECDSA can be used in practice. Nevertheless, the security of the proxy-protected ECDSA is as secure as the standard signature ECDSA.

We have proposed both proxy-protected signature on DSA and proxy-protected signature scheme and relative application on the ECDSA, called a proxy-protected DSA and proxy-protected ECDSA respectively. The proxy-protected DSA (ECDSA) is a variant of DSA (ECDSA), which satisfies not only the security of signature but

also the required secure properties of proxy signature and strong proxy signature. We use an example adequately to demonstrate the proxy-protected ECDSA in practice. Moreover, proxy-protected DSA (ECDSA) and conventional DSA (ECDSA) can be used in one scheme.



Table 3.2. Points on the elliptic curve $x^3 + x + 6 \pmod{11}$ [Sti02]

X	$x^3 + x + 6 \pmod{11}$	$y' = \pm z^3 \pmod{11}$	$(y')^2$	y	QR?
0	6	4,7	5		No
1	8	5,6	3		No
2	5	4,7	5	4,7	Yes
3	3	5,6	3	5,6	Yes
4	8	5,6	3		No
5	4	2,9	4	2,9	Yes
6	8	5,6	3		No
7	4	2,9	4	2,9	Yes
8	9	3,8	9	3,8	Yes
9	7	2,9	4		No
10	4	2,9	4	2,9	Yes

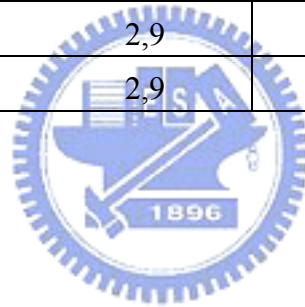


Table 3.3. The multiples of generator G [Sti02]

$G = (2,7)$	$2G = (5,3)$	$3G = (8,3)$	$4G = (10, 2)$
$5G = (3,6)$	$6G = (7,9)$	$7G = (7,2)$	$8G = (3,5)$
$9G = (10,9)$	$10G = (8,8)$	$11G = (5,9)$	$12G = (2,5)$

3.4. Applications on Public Key Infrastructures (PKIs)

3.4.1. PKI architecture

PKI is an authentication technology that provides means for replying parties to know that the publics actually belong to the parties. Using a public key cryptography, PKI enables the services including data confidentiality, data integrity. The main framework of PKI is defined in ITU-T X.509 Recommendation [CF⁺03,AF99]. The public Key Infrastructure X.509 working group of the Internet Engineering Task Force (IETF) has been developing X509-based PKI (PKIX) model that is suitable for deploying a certificate-based architecture on the Internet.

Figure 3.1 is a simplified view of the architectural model assumed by the PKIX specifications. The components in this model are End Entity, CA, RA, CRL issuer, and repository.

3.4.2. PKIX components

In this section, we briefly describe the components of the PKIX model.

- End entity: An end entity can be an end-user or a device such as a router, a

server or a process on anything can be identified in the subject name.

- Certification Authority (CA): CA is a component that can issue certificates. Certificates are signed by the issuing CA. CA are also responsible for issuing CRLs unless this been delegated to a separate CRL issuer. CA may involve registration tasks, but these are often delegated to the Registration authority (RA).
- Registration Authority (RA): The RA verifies the identity of the end entity attempting to register with the PKI. The RA should perform to verify that the subject has possession of the private key being registered and validate the parameters of public keys for registration.
- Repositories: A repository is a term used to denote any method for storing and retrieving PKI-related information such as certificates and CRLs.



3.4.3. PKIX management functions

Management protocols are required to support on-line interactions between PKI user and management entities. The functions need to be supported by management protocols in the proxy signature include:

- Registration: This is a process whereby a user first makes itself known t a CA or RA. The step is usually associated with the initial verification of the entities

identity. The processor could be accomplished directly with the CA or through an RA.

- Initialization: This step involves initializing the associated trust anchor with the end entities. Additional information such as applicable certificate policies may also be supplied.
- Certification: This is the process in which a CA issues a public key certificate for a user's public key, and publishes that certificate in a repository.

The PKIX management functions also include key pair recovery, key pair update, revocation request, etc. These are not main processes in proxy signature applied in PKI.



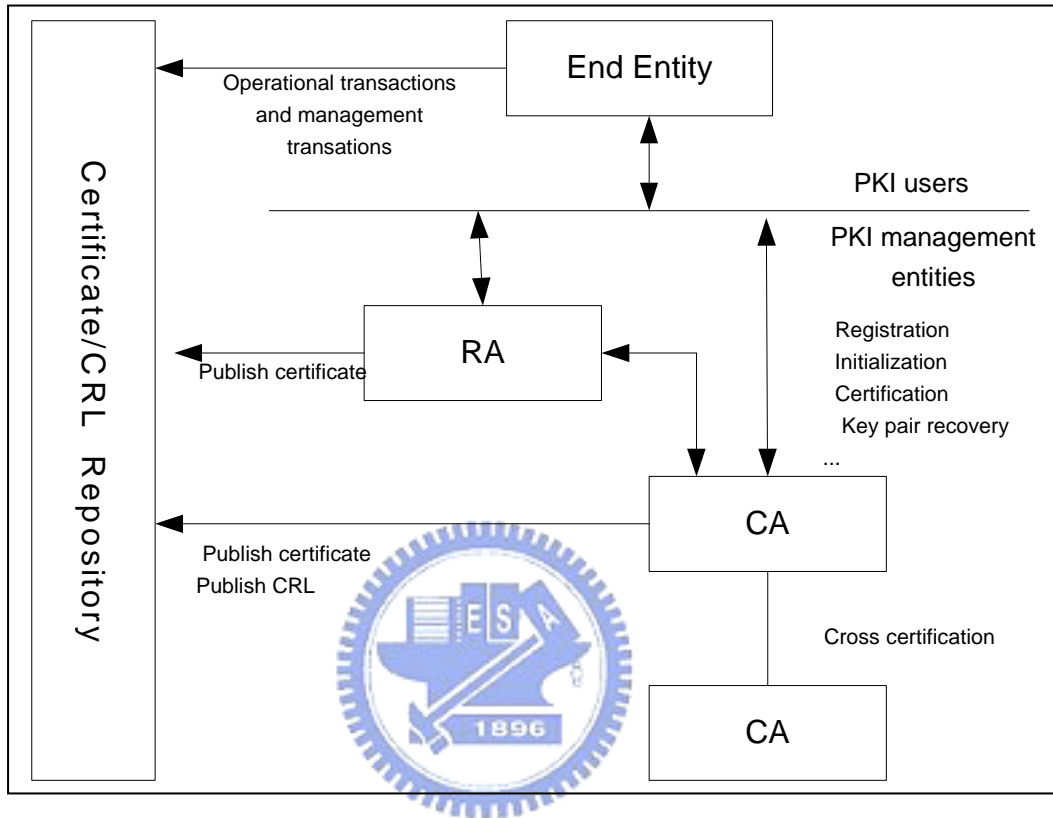
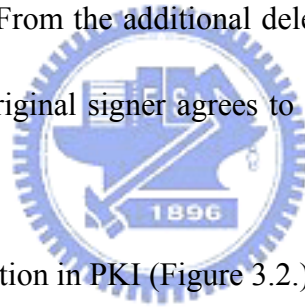


Figure 3.1. The PKIX Architecture Model

3.4.4. Proxy signature applied in PKIs

Our approach is to use X.509v3 certificate extension to indicate the relationship between an original signer and proxy signer by proxy parameters.

Suppose that the original signer is certified by a CA and has a public key certificate. The proxy signer enrolls into the PKI and then creates a proxy key with the original signer. At initialization step, the CA/RA needs to verify the relationship of the delegation from cryptographic technologies. The proxy key is derived from the original signer's private key. From the additional delegation information, the CA/RA must be convinced that the original signer agrees to delegate the signing capacity to the proxy signer.



The proxy signer initialization in PKI (Figure 3.2.) executes the following steps:

Step 1. The original signer and the proxy signer create a proxy protected proxy secret interactively.

Step 2. The proxy signer creates a proxy key pair.

Step 3. The proxy signer sends the public key information including the public information and certification request [RSA00] to the RA.

Step 4. On receiving the message from proxy signer, the RA verifies the identity of the proxy signer and the certificate policy. If they are qualified and sends the certificate request to the CA.

Step 5. The CA executes the process of signing to create a certificate.

Except the mechanisms provides the proxy signature scheme in PKI, the policy of CA must support the scheme. Due to deploying the proxy signature scheme, the constraints in the CA function will be raised.

When a verifier checks the validity of proxy signature, he uses the same algorithm as showed in Figure 3.3.



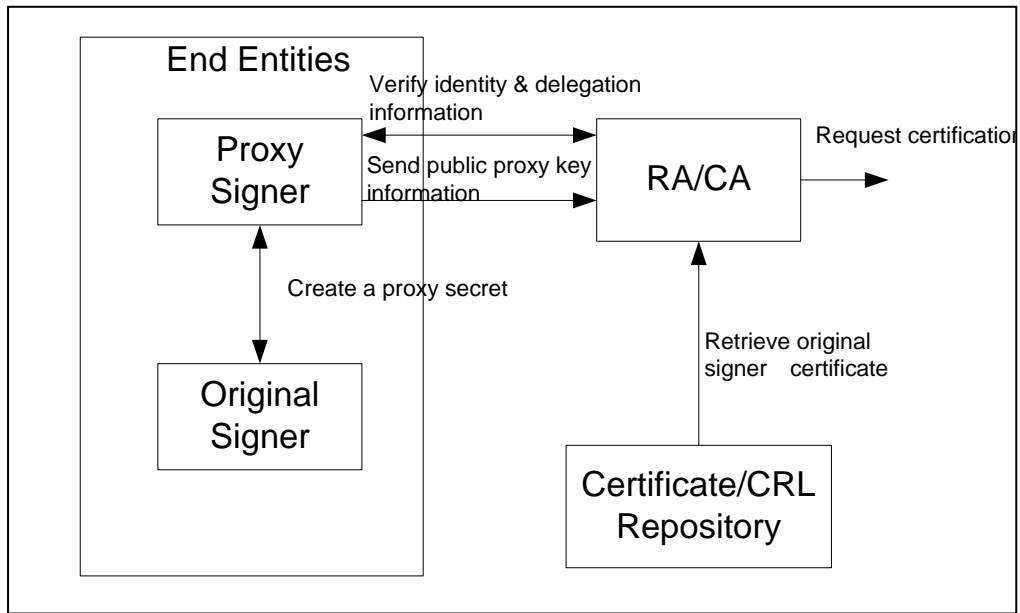


Figure 3.2 Proxy signer initialization in PKI

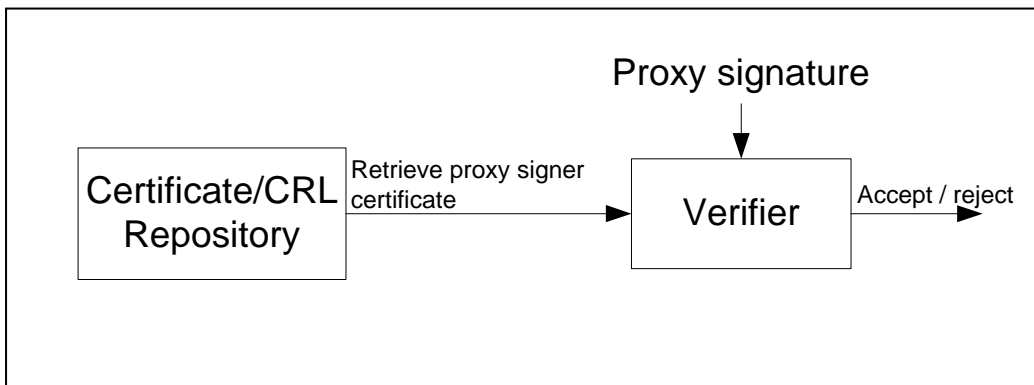


Figure 3.3 Verification of proxy signature in PKI

Chapter 4 Forward-Secure Proxy Signature

Generally speaking, many of proxy signature schemes can be proven secure under very reasonable assumptions. In many solutions, security guarantees last as long as the secret keys remain unrevealed. If a secret key is revealed, security is compromised and any signature created by the key is no longer trusted. In the literature of cryptography, many schemes are proposed to enhance security against the key exposure problem, such as threshold scheme [SLH99,Sun99], proactive schemes, and forward-secure scheme [AMN01,BM99,GQ88]. In a (m, n) threshold scheme, security is supposed under the condition that the adversary is restricted to comprise less than m of the n shares throughout the entire key lifetime. The proactive schemes are similar to threshold schemes except renewing all of the shares periodically.

Recently, some researchers propose proxy signature schemes deploying threshold schemes to enhance their security. However, threshold schemes and proactive schemes are group-oriented schemes, which are not suitable for a single signer or a single proxy signer. Consequently, proxy signature deploying forward-secure scheme is a novel approach. The object of forward-secure scheme is to protect signature security against the risk of key exposure without requiring effort of key distributions.

In this chapter, we propose a proxy signature scheme with forward-secure property in which we adopt the concept of forward-secure property for proxy

signatures against key exposure. In our scheme, the duration of proxy time is divided into several time periods. At each time period the proxy signer renews its proxy keys and deletes its previous proxy keys. Therefore, the proxy key is used only for a period to reduce the potential damage in case of the proxy key exposed. Once a proxy key is exposed, the attacker cannot forge a signature what is created before the time prior of key exposure.

A proxy signature scheme with forward secure property is a strong proxy signature scheme. Besides the requirements of strong proxy scheme from **R1** to **R5** (in chapter 2), It needs a more requirement 6 as follows:

R6 – forward secure property: disclosure of the present secret key does not compromise the secrecy of the earlier signature.



4.1. Forward-Secure Proxy signature Scheme

In the forward-secure proxy signature scheme, there is a system authority (SA) to authorize the identities and broadcast public information. The SA generates the parameters N , v and t [OS91] (referred to 2.3.2). The forward-secure proxy signature scheme involves several participants: an original signer ‘Alice’, a proxy signer ‘Bob’, and a verifier. There is a warrant m_w to constrain the relationship of an original signer and a proxy signer such as the identities of the protocol, the duration of

delegation, and the usage of proxy key.

4.1.1. Protocol of the forward-secure proxy signature scheme

Alice and Bob generate key pairs respectively. Then, Alice generates a new key that derives from Alice's private key and sends to Bob. On receiving the key from Alice, Bob creates a proxy key. At the next period, Bob updates the proxy key. To sign on a message, Bob uses the current proxy key; and then any verifier can verify the proxy signature uses both Alice's and Bob's public keys.

Alice selects a random $s_A \in Z_N^*$ as a private key and computes a corresponding public key $u_A = 1/s_{A_0}^{2^{v(t+1)}} \pmod N$; and then Bob selects a random $s_B \in Z_N^*$ as a private key and computes a corresponding public key $u_B = 1/s_{B_0}^{2^{v(t+1)}} \pmod N$. The SA certifies both of the public keys. The details of the protocol are described in Figure 4.1 and explained as follows.

(Proxy generation and delivery - Executed by between Alice and Bob)

Alice does the following steps.

Step 1. Select a random $k_A \in Z_N^*$ and computes $r_A = (1/k_A)^{2^{v(t+1)}} \pmod N$.

Step 2. Compute $e_A = H(m_W, r_A)$ and $\sigma_A = k_A s_A^{e_A} \pmod N$.

Step 3. Then, send the proxy secret (σ_A, m_W, r_A) to Bob in a secure manner.

(Proxy verification and proxy key generation - Executed by Bob)

Upon receiving (σ_A, m_W, r_A) , Bob does the following steps.

Step 1. If $\sigma_A = 0 \pmod{N}$ then reject the proxy key, else do the following steps.

Step 2. Then, compute $e_A = H(m_W, r_A)$ and $r_A' = (1/\sigma)^{2^{v(t+1)}} (1/u_A)^{e_A} \pmod{N}$.

Step 3. If $e_A = H(m_W, r_A')$ then do the following steps, else reject it.

Step 4. Compute $\sigma_{B_0} = \sigma_A s_B^{e_A} \pmod{N}$.

Therefore, at the beginning Bob generates a proxy key is (σ_{B_0}, m_W, r_A) .

(Proxy key update - Executed by Bob)

At the period j , Bob updates his private key $(j, \sigma_{B_j}, m_W, r_A)$ from $(j-1, \sigma_{B_{j-1}}, m_W, r_A)$ by $\sigma_{B_j} = (\sigma_{B_{j-1}})^{2^v} \pmod{N}$ and deletes previous private key $\sigma_{B_{j-1}}$.



(Proxy signature signing- Executed by Bob)

At the time period j , Bob signs on a message m and does the following steps:

Step 1. Select a random $k \in \mathbb{Z}_N^*$.

Step 2. Compute $r = k^{2^{v(t+1-j)}} \pmod{N}$, $e = H(m, r, j)$ and $\sigma = k (\sigma_{B_j})^e \pmod{N}$.

Therefore, the signature on message M is (j, σ, r, r_A) .

(Verification of the proxy signature)

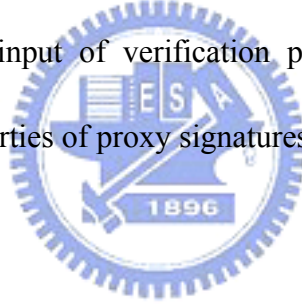
A verifier can verify the validity of the signature (j, σ, r, r_A) on the message M by following steps:

Step 1. If $\sigma = 0 \pmod{N}$ then reject the signature, else do the following steps.

Step 2. Compute $e_A = H(m_W, r_A)$ and $r' = \sigma^{2^{v(t+1)-j}} (r_A (u_A u_B)^{e_A})^e \pmod{N}$ in which the $r_A (u_A u_B)^{e_A}$ as a proxy public key.

Step 3. If $e = H(m, r', j)$ then accept the signature, else reject it.

In our forward-secure proxy signature scheme, we modified the Abdalla-Reyzin forward-secure digital signature scheme with the output of signature signing phase and the input of verification phase. The output of signature signing phase is (j, σ, r) instead of (j, σ, e) and then we add a parameter r_A to reform the signature as (j, σ, r, r_A) . However, the input of verification phase is relatively modified by (j, σ, r, r_A) to keep the properties of proxy signatures.



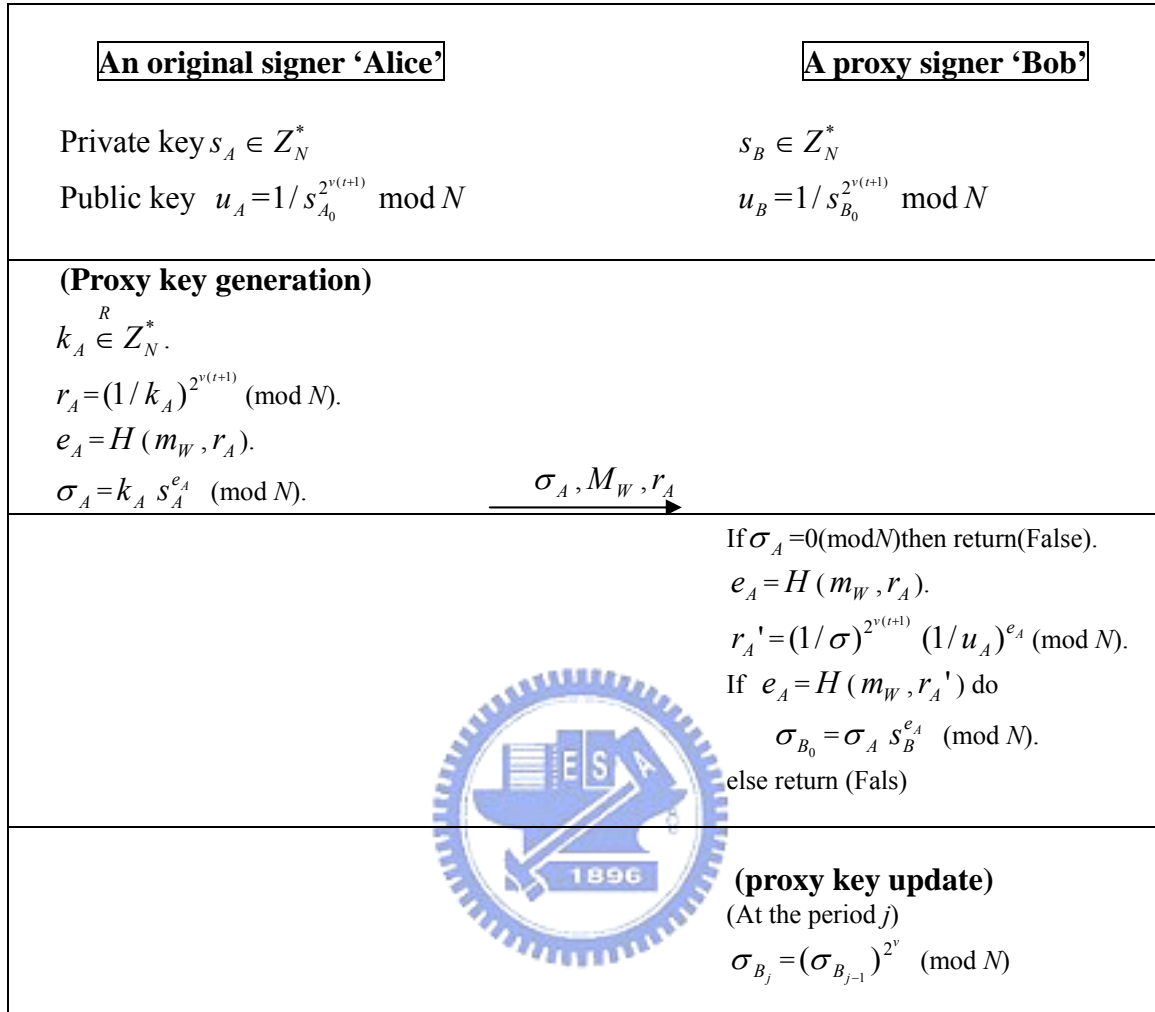


Figure 4.1.1 The protocol of the forward-secure proxy signature scheme (1)

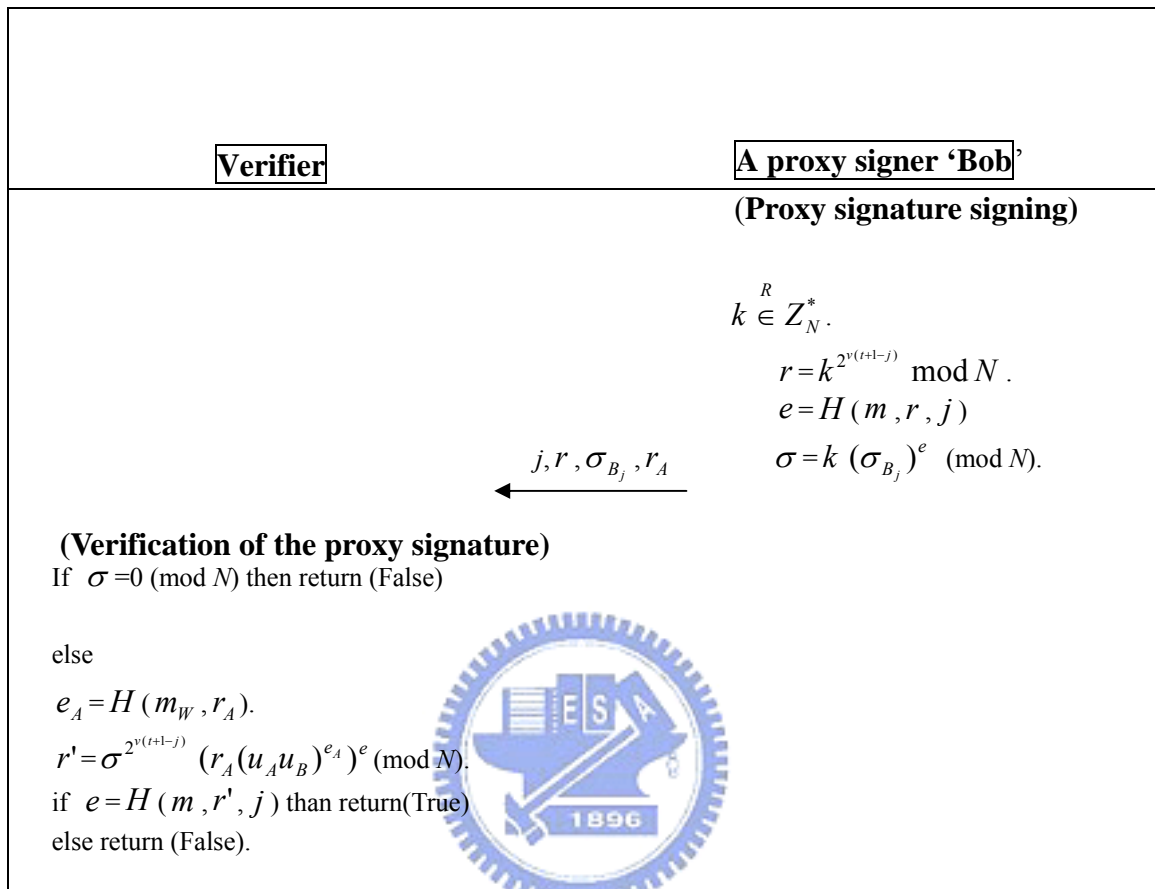


Figure 4.1.2 The protocol of the forward-secure proxy signature scheme (2)

4.1.2. Correctness and conformance of the forward-secure proxy signature scheme

The following lemma and theorem prove an original signer can delegate its right of signature and the forward-secure proxy signature works correctly.

Lemma 4.1: *If an original signer Alice delegates its signing capability to a proxy signer Bob by a proxy secret (σ_A, r_A) , then the proxy signer can verify proxy secret's validity.*

Proof:

An original signer's delegation works correctly by computing

$$\begin{aligned} r_A' &= (1/\sigma_A)^{2^{v(t+1)}} (1/u_A)^{e_A} \\ &= (1/k_A)^{2^{v(t+1)}} \\ &= r_A \pmod{N}. \end{aligned}$$

Thus $e_A = H(m_w, r_A')$, as require. □

Form the Lemma 4.1 Bob receive the proxy secret and checks the secret is correct.

The following theorem proves that the proposed signature works correctly.

Theorem 4.1: *If the forward-secure proxy signature (j, σ, r, r_A) on a message m is valid at the period j , it will pass proxy signature verification.*

Proof:

If the original signer delegate its signature capability correctly by *Lemma 1*, we will prove that $H(m, r, j) = H(m, r', j)$ by the following statements:

$$\begin{aligned}
r' &= \sigma^{2^{v(t+1-j)}} (r_A (u_A u_B)^{e_A})^e \pmod{N} \\
&\text{(Substitute } \sigma = k (\sigma_{B_j})^e \pmod{N}) \\
&= (k (\sigma_{B_j})^e)^{2^{v(t+1-j)}} (r_A (u_A u_B)^{e_A})^e \pmod{N} \\
&\text{(Substitute } \sigma_{B_j} = (\sigma_{B_{j-1}})^{2^v} \pmod{N}) \\
&= k^{2^{v(t+1-j)}} ((\sigma_{B_j})^{2^{v(t+1-j)}})^e (r_A (u_A u_B)^{e_A})^e \pmod{N} \\
&\text{(Substitute } \sigma_{B_0} = \sigma_A s_A^{e_A} \pmod{N}) \\
&= k^{2^{v(t+1-j)}} ((\sigma_{B_0})^{2^{v(t+1)}})^e (r_A (u_A u_B)^{e_A})^e \pmod{N} \\
&\text{(Substitute } \sigma_{B_0} = \sigma_A s_B^{e_A} \pmod{N}) \\
&= k^{2^{v(t+1-j)}} (\sigma_A^{2^{v(t+1)}} ((s_B^{2^{v(t+1)}})^{e_A})^e (r_A (u_A u_B)^{e_A})^e \pmod{N} \\
&\text{(Substitute } \sigma_A = k_A s_A^{e_A} \pmod{N}) \\
&= k^{2^{v(t+1-j)}} ((1/r_A)((1/u_A)(1/u_B))^{e_A})^e (r_A (u_A u_B)^{e_A})^e \pmod{N} \\
&= k^{2^{v(t+1-j)}} \pmod{N} \\
&= r
\end{aligned}$$

Therefore, $H(m, r, j) = H(m, r', j)$, as required. \square

From a signature, the original signer's agreement on the message m using a proxy key is verified explicitly if the proxy signature (j, σ, r, r_A) passes the checking successfully via *Theorem 4.1*. Furthermore, the proxy key is derived from the proxy

secret that is made by the original's private key correctly by *Lemma 4.1*. Therefore, from the signature, a verifier can identify the proxy signer's signing capability delegated by the original signer.

Identity information of the proxy signer is included in the signature. Anyone can determine the identity of the corresponding proxy signer because of the proxy signer's public key u_B .

4.1.3. Security analysis of the forward-secure proxy signature scheme

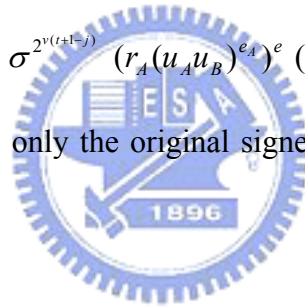
By reason of the security requirements, the length of the *blum* integer N is more than 1024 bits and the security parameter v is more than 6 in the forward-secure proxy signature scheme.

The forward-secure proxy signature scheme is a type of partial delegations in which the proxy secret σ_A is derived from the original signer's private key. Then, the proxy signer uses the proxy secret to create a proxy key σ_{B_0} for protecting the proxy signer. It is computationally infeasible to derive from σ_A to σ_{B_0} based on 2^v -th square root problem. Thus, only a designated proxy signer can create a valid proxy signature. The result leads the forward-secure proxy signature scheme to conforming to the property of *unforgeability*.

From the signature, the original signer's agreement on the signed message M

using a proxy key is also verified explicitly, if the proxy signature (m, e, σ) passes the checking successfully. The proxy key is derived from the proxy secret which is made by the original's private key. Therefore, from the signature a verifier can verify the proxy signer's signing capability delegated by the original signer. The design is for *verifiability*.

The issue of security in the forward-secure proxy signature scheme inherits from the Abdalla-Reyzin forward-secure digital signature scheme except the proxy key generation. An original signer delegates its signing capability to the proxy signer by creating 2^v -th root signature [AR00,MVS96]. It is computationally infeasible to solve the equation of $e = H(m, \sigma^{2^{v(r+1-j)}} (r_A(u_A u_B)^{e_A})^e \pmod{N}, j)$ by only known σ, r_A, r, j and m . Therefore, only the original signer can create the proxy secret to delegate its signing capability.



4.1.4. Comparison

We compare the proposed scheme with others proxy signature on the properties in Table 4.1. In [MUO₁₉₆], an original signer delegates an unlimited signing capability to a proxy signer and the delegation is transferable to others because it does not use warrant. In [LK99], the scheme use warrant to limit the proxy relationship and proxy signer does not update the proxy key such that without the forward-secure property. The proxy-protected DSA (proposed in Chapter 3) uses the PKI to be identified the

proxy signer by others and uses the policy in PKI to limit the delegation.

Table 4.1. The comparison of proxy signature schemes

Features	[MUO196]	[LK99]	Proxy protected DSA in PKI	Proposed Scheme
Verifiability	O	O	O	O
Strong unforgeability	X	O	O	O
Strong identifiability	X	O	O	O
Strong undeniability	X	O	O	O
Prevention of misuse	X	O	O	O
Limited delegation	X	O	O	O
Non-transferability	X	O	O	O
Forward-secure property	X	X	X	O

4.2. A Variant Forward-Secure Proxy Signature

However, the proposed forward-secure proxy scheme is non-designated proxy signature. In some application, the identities are need authenticated by each other. We propose another proxy signature scheme with time limitation which can obtain a time limitation property on delegation duration of proxy signatures. Furthermore, the verifier can identify which proxy signer signed on the document.

4.2.1. Protocol variant of forward-secure proxy signature

The system exists a system authority (SA) to create parameters and certify the public information. The system parameters are t , v and N as in the section 4.1. The function h is a one-way hash function with bit length of the output more than 160 bits and “||” denotes a concatenation of two strings. The proposed scheme includes five phases – proxy key generation, proxy key acceptance, proxy key update, proxy signature signing, and proxy verification. The protocol of the proposed scheme is described in Figure 4.2 and explained as follows:

An original signer Id_A (Id_A is a unique identifier) selects a random $s_A \in \mathbb{Z}_N^*$ as its private key and computes its corresponding public key is $u_A = (1/s_A)^{2^{v(t+1)}} \pmod{N}$.

A proxy signer Id_B (Id_B is also a unique identifier) selects a random $s_B \in \mathbb{Z}_N^*$ as its private key and compute its corresponding public key is $u_B = (1/s_B)^{2^{v(t+1)}} \pmod{N}$.

The above two public keys and identifiers are certified by a SA.

(Proxy key generation and delivery)

The original signer and the proxy signer create a proxy key interactively. They should the following steps:

Step 1. An original signer Id_A executes *Algorithm PxyGen*(Id_A, Id_B, s_A) in

Figure 4.3 to get a proxy secret (σ_A, r_A) and sends it to a designated proxy signer Id_B in a secure manner.

Step 2. For accepting the proxy secret, a proxy signer Id_B should obtain the original signer Id_A 's proxy secret (σ_A, r_A) .

Step 3. The proxy signer confirms the validity of (σ_A, r_A) by checking if $e_A = h(Id_A || Id_B || r_A')$ holds by *Algorithm PxyAccept* $(Id_A, Id_B, s_B, (\sigma_A, r_A))$ in Figure 4.4.

Step 4. Then the proxy signer computes $\sigma_{B_0} = \sigma_A (s_B)^{e_A} \pmod{N}$.

Thus, the proxy signer's proxy tuple is (σ_{B_0}, r_A) . The key σ_{B_0} with index '0' is at the basic state.

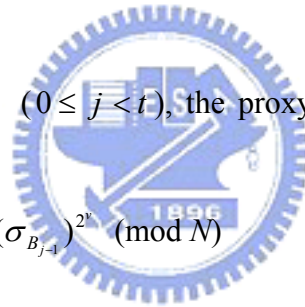
(Proxy key update)

At each current period j ($0 \leq j < t$), the proxy signer Id_B renews the proxy key from $\sigma_{B_{j-1}}$ to σ_{B_j}

Step 1. Computes $\sigma_{B_j} = (\sigma_{B_{j-1}})^{2^j} \pmod{N}$

Step 2. Deletes $\sigma_{B_{j-1}}$

The proxy key needs to be updated at every valid period



(Proxy signature signing)

Because at the period j ($0 < j \leq t$), the proxy key is σ_{B_j} , the proxy signer needs to sign a message M using *algorithm Sign* (m, σ_{B_j}, j) in Figure 4.5 and the output is (σ, r, j) . For the verifiability property, the output needs to add the parameter r_A such that the proxy signer's signature for m is $((\sigma, r, j), r_A)$.

(Verification of the proxy signature)

To verify the proxy signature $((\sigma, r, j), r_A)$ on a message m , a verifier should obtain Id_A , Id_B , u_A and u_B , and compute $e_A = h(Id_A || Id_B || r_A)$ and $r_A(u_A u_B)^{e_A}$ as a proxy public key $r_A(u_A u_B)^{e_A}$. Then the verifier executes *algorithm* $Vf(m, r_A(u_A u_B)^{e_A}, ((\sigma, r, j), r_A))$ in figure 4.6. If the output is ‘False’ reject the signature, else accept it.

The building blocks of the proposed scheme are as the same as the forward-secure proxy signature scheme except the identity of original and proxy signer.



4.2.2. Correctness

The following lemma and theorem prove an original signer delegates its right by signature and the proposed signature works correctly.

Lemma 4.2: *If an original signer Id_A delegates its signing capability to a proxy signer Id_B by proxy secret (σ_A, r_A) , then the proxy signer can verify its validity.*

Proof: An original signer’s delegation works correctly by computing $r_A' = (1/\sigma_A)^{2^{v(t+1)}} (1/u_A)^{e_A} = (1/k_A)^{2^{v(t+1)}} = r_A \pmod{N}$. Thus $e_A = H(Id_A || Id_B || r_A')$, as require. □

Theorem 4.2: *If the proposed signature $((\sigma, r, j), r_A)$ on a message M is valid at the period j , it will pass verification of proxy signature of the proposed scheme.*

Proof: If the original signer delegate its signature capability correctly by Lemma 1, we will prove that $H(M || r || j) = H(M || r' || j)$ by the following statements:

$$\begin{aligned}
r' &= \sigma^{2^{v(t+1-j)}} (r_A (u_A u_B)^{e_A})^e \pmod{N} \\
&= (k(\sigma_{B_j})^e)^{2^{v(t+1-j)}} (r_A (u_A u_B)^{e_A})^e \pmod{N} \\
&= k^{2^{v(t+1-j)}} ((\sigma_{B_j})^{2^{v(t+1-j)}})^e (r_A (u_A u_B)^{e_A})^e \pmod{N} \\
&= k^{2^{v(t+1-j)}} ((\sigma_{B_0})^{2^{v(t+1)}})^e (r_A (u_A u_B)^{e_A})^e \pmod{N} \\
&= k^{2^{v(t+1-j)}} (\sigma_A^{2^{v(t+1)}} ((s_B^{2^{v(t+1)}})^{e_A})^e (r_A (u_A u_B)^{e_A})^e \pmod{N} \\
&= k^{2^{v(t+1-j)}} ((1/r_A)((1/u_A)(1/u_B))^{e_A})^e (r_A (u_A u_B)^{e_A})^e \pmod{N} \\
&= k^{2^{v(t+1-j)}} \pmod{N} \\
&= r
\end{aligned}$$

Therefore, $H(M || r || j) = H(M || r' || j)$, as required. \square

4.2.3. Conformance with properties of proxy signature

(Verifiability) From a signature the original signer's agreement on the message M using a proxy key is verified explicitly, if the proxy signature $((\sigma, r, j), r_A)$ passes the checking successfully using by Theorem 4.2. Furthermore, the proxy key is

derived from the proxy secret and the proxy secret is derived by the original's private key by *Lemma 4.2*. Therefore, from the signature a verifier can identify the proxy signer's signing capability delegated by the original signer.

(Identifiability) Identity information of the proxy signer is included in the signature. Anyone can determine the identity of the corresponding proxy signer because the proxy signer's public key u_B and the unique identifier Id_B is required in order to check a proxy signature.

4.2.4. Security analysis

In the proxy key update phase, a proxy signer renews its proxy key $\sigma_{B_j} = (\sigma_{B_{j-1}})^{2^v} \pmod{N}$ and deletes $\sigma_{B_{j-1}}$ at the period j . It is computationally infeasible to get $\sigma_{B_{j-1}}$ from σ_{B_j} without knowing p_1 and p_2 . Moreover, an original signer Id_A delegates its signing capability to the proxy signer Id_B using creating 2^v -th root signature [OS91].

In this scheme, a proxy signature includes the value j of the period. A verifier can verify a proxy signature in the valid period by checking whether the value j of the period is in the delegation duration. Therefore, each period can be presented a proxy period. For example, a manager (an original signer) wants to delegate his signing capability to his secretary (a proxy signer) for a month. The time period could be set $t=30$ and each period has length one day. The proxy public key keeps for one month

and the proxy key is updated daily. When a proxy key finishes updating (i.e. it has been updated for all the periods.), it is revoked automatically.

The key exposure problem is a serious problem against the security of a strong proxy signature scheme. We have proposed a new proxy signature scheme to avoid the problems of key compromise. The proposed scheme is also a very helpful tool in which an original signer delegates its signing capability to a proxy signer with limitation of the duration.



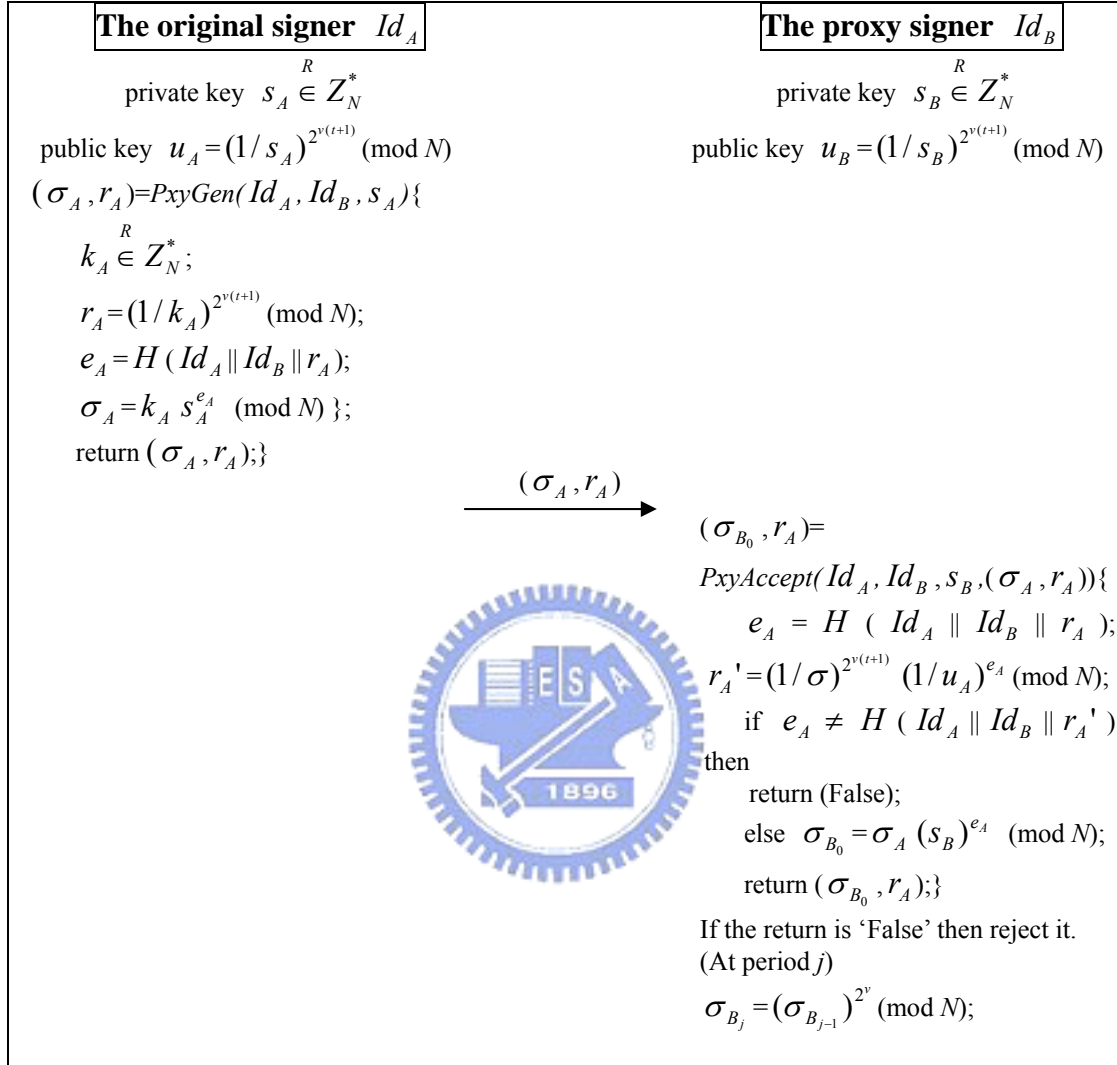


Figure 4.2.1 The protocol of a variant forward-secure proxy signature (1)

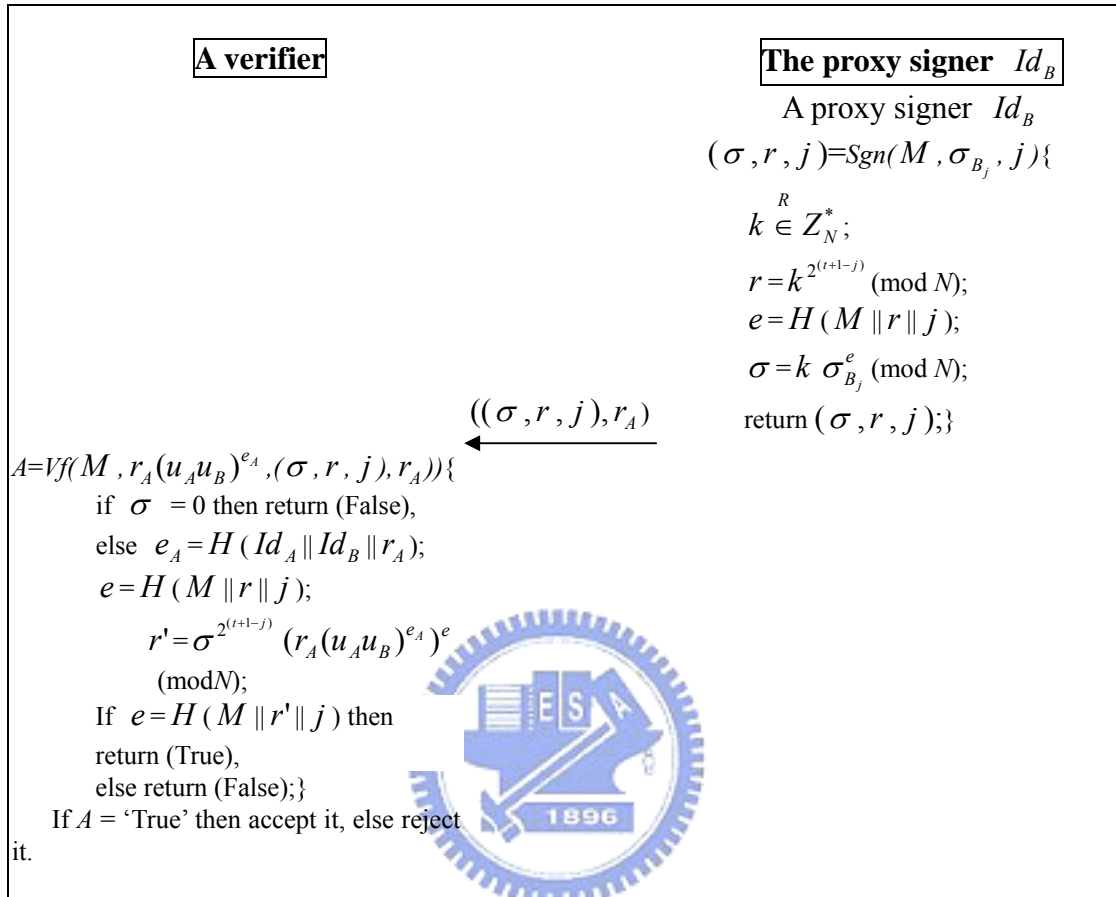


Figure 4.2.2 The protocol of a variant forward-secure proxy signature (2)

Algorithm PxyGen(Id_A : an original proxy signer's identifier, Id_B : a proxy signer's identifier, s_A : an original signer's private key): Proxy generation algorithm.

Summary: An original signer Id_A creates a proxy secret (σ_A, r_A) for proxy signer Id_B .

Step 1. Generate a random k_A ($k_A \in \mathbb{Z}_N^*$).

Step 2. Compute $r_A = (1/k_A)^{2^{v(t+1)}} \pmod{N}$,

Step 3. Compute $e_A = h(Id_A || Id_B || r_A)$ and $\sigma_A = k_A s_A^{e_A} \pmod{N}$

Step 4. Return proxy secret (σ_A, r_A) .

Figure 4.3. The algorithm of forward-secure proxy key generation

Algorithm PxyAccept(Id_A : an original proxy signer's identifier, Id_B : a proxy signer's identifier, s_B : a proxy signer's private key, (σ_A, r_A) : a proxy secret):
 Proxy acceptance algorithm.

Summary: The proxy signer Id_B accepts the proxy secret and creates proxy key (σ_{B_0}, r_A) .

Step 1. Compute $e_A = h(Id_A || Id_B || r_A)$, and $r_A' = (1/\sigma)^{2^{v(r+1)}} (1/u_A)^{e_A} \pmod{N}$.

Step 2. If $e_A \neq h(Id_A || Id_B || r_A')$ then return (False), else compute

$$\sigma_{B_0} = \sigma_A (s_B)^{e_A} \pmod{N}$$

Step 3. Return (σ_{B_0}, r_A) .



Figure 4.4. The algorithm of forward-secure proxy acceptance

Algorithm $\text{Sign}(M : \text{a message to be signed}, s_j : \text{a private key}, j : \text{a current time period})$: Forward-secure signature algorithm.

Summary: A signer generates a signature on a message m using private key s_j and a current period j .

Step 1. Select a random k ($k \in \mathbb{Z}_N^*$).

Step 2. Compute $r = k^{2^{t+1-j}} \pmod{N}$, $e = \text{h}(m \parallel r \parallel j)$, and $\sigma = k s_j^e \pmod{N}$.

Step 3. Return (σ, r, j) .

Figure 4.5 Forward-secure signature algorithm



Algorithm $Vf(m: \text{ a message, } u : \text{ a public key, } (\sigma, r, j): \text{ a signature}):$
 Forward-secure verification algorithm.

Summary: Any verifier can verify the signature (σ, r, j) on a message M using public key u .

Step 1. if $\sigma = 0$ then return (False), else compute $e = h(m \parallel r \parallel j)$ and

$$r' = \sigma^{2^{(t+1-j)}} u^e \pmod{N}.$$

If $e = h(m \parallel r' \parallel j)$ then return (True), else Return (False).

Figure 4.6 Forward-secure verification algorithm



Chapter 5 One-time Signature and its Application to Proxy Signature

In this chapter, we propose a new scheme to generalize the Lamport one-time signature and its applications. We group the message by power of 2. Then, each group of the message is encoded and signs individually by selecting the corresponding private keys from the private key box to create the signature. Thus, the general Lamport one-time signature scheme saves on the storage space of the public keys and the size of the signatures. Moreover, we propose an efficient solution for signing a long message to make the general Lamport one-time signature scheme more operative in practical. Furthermore, we improve general Lamport one-time signature scheme to save move storage space, and propose its application on proxy signature scheme. One-time proxy signatures are one-time signatures for which an original signer can delegate his signing capability to a proxy signer.

5.1. Improving Lamport One-Time Signature

In this section, we introduce an efficient scheme for one-time signature in which the length of public keys and signature will be reduced.

5.1.1. Protocol of variant Lamport one-time signature

The Lamport one-time signature scheme faces the long length of signatures that are the half of private key box. We propose a new scheme to reduce the size of signature and create a variant Lamport one-time signature. We consider that the Lamport one-time signature scheme is a special case of the variant Lamport one-time signature scheme. Suppose that $h:Y \rightarrow Z$ is a one-way hash function. The new scheme is described as follows:

(Key generation)

We should do the following steps:

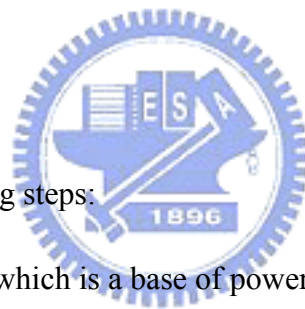
Step 1. Select a number v which is a base of power 2 like 2, 2^2 , 2^3 (say $v=2^e$)

Step 2. Compute the columns l of key array by encoding the message m based on v as $m=(m_1\dots m_l)_v$.

Step 3. Select l elements $y_{i,0} \in Y$ at random with $1 \leq i \leq l$ at the first row of the private key box. At each column i , select e elements with suffix by power of 2 as $y_{i,1}, y_{i,2^1}, \dots, y_{i,2^e}$.

Step 4. Compute the corresponding public key box by using hash function.

Thus private key box SK and the public key PK box as follows:



$$SK = \begin{pmatrix} y_{1,0} & y_{2,0} & \cdots & y_{l,0} \\ y_{1,1} & y_{2,1} & \cdots & y_{l,1} \\ y_{1,2^1} & y_{2,2^1} & \cdots & y_{l,2^1} \\ \vdots & \vdots & & \vdots \\ y_{1,2^e} & y_{2,2^e} & \cdots & y_{l,2^e} \end{pmatrix}$$

$$PK = \begin{pmatrix} z_{1,0} & z_{2,0} & \cdots & z_{l,0} \\ z_{1,1} & z_{2,1} & \cdots & z_{l,1} \\ z_{1,2^1} & z_{2,2^1} & \cdots & z_{l,2^1} \\ \vdots & \vdots & & \vdots \\ z_{1,2^e} & z_{2,2^e} & \cdots & z_{l,2^e} \end{pmatrix}$$

(Signature signing)

To sign the message m , we should do the following steps:

Step 1. Encode the message based on v as $m = (m_1 \dots m_l)_v$.

Step 2. If the digit of the message $m_i = 0$, then select the first row of corresponding entry $y_{i,0}$.

Step 3. Encode each digit based on 2 as $m_i = (u_1 u_2 \dots u_e)_2$.

Step 4. At each column i and e -bit message $m_i = (u_1 u_2 \dots u_e)_2$, If $u_j = 1$ select $y_{i,2^j}$, else discard it.

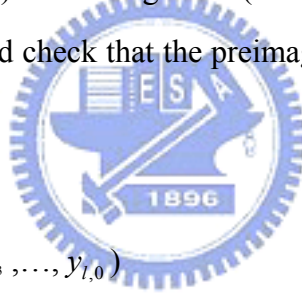
Thus, the signature of message m is the selected items in the private key box (a_1, a_2, \dots, a_n) .

For example $m = (3A \dots 0)_{16}$, we encode $3 = (11)_2$ and $A = (0101)_2$ and select the corresponding entry in the private key box as $(y_{1,1}, y_{1,2^1}, y_{2,2^1}, y_{2,2^3}, y_{l,0})$

$$\begin{aligned}
\text{sig}(3A\dots 0) &= \begin{pmatrix} y_{1,0} & y_{2,0} & \cdots & [y_{l,0}] \\ [y_{1,1}] & y_{2,1} & \cdots & y_{l,1} \\ [y_{1,2^1}] & [y_{2,2^1}] & \cdots & y_{l,2^1} \\ \vdots & \vdots & & \vdots \\ y_{1,2^3} & [y_{2,2^3}] & \cdots & y_{l,2^3} \end{pmatrix} \\
&= (y_{1,1}, y_{1,2^1}, y_{2,2^1}, y_{2,2^3}, \dots, y_{l,0})
\end{aligned}$$

(Verification)

To verify the signature (a_1, a_2, \dots, a_n) on message, we run the verification algorithm in the above section. For example, we check the signature $(y_{1,1}, y_{1,2^1}, y_{2,2^1}, y_{2,2^3}, \dots, y_{l,0})$ on message $m = (3A\dots 0)_{16}$. Select the corresponding entry in the public key box and check that the preimage of the selected entries are the signatures as the follows:



$$\text{vf}(y_{1,1}, y_{1,2^1}, y_{2,2^1}, y_{2,2^3}, \dots, y_{l,0})$$

$$= \begin{pmatrix} z_{1,0} & z_{2,0} & \cdots & [z_{l,0}] \\ [z_{1,1}] & z_{2,1} & \cdots & z_{l,1} \\ [z_{1,2^1}] & [z_{2,2^1}] & \cdots & z_{l,2^1} \\ \vdots & \vdots & & \vdots \\ z_{1,2^3} & [z_{2,2^3}] & \cdots & z_{l,2^3} \end{pmatrix}$$

In the variant Lamport one-time signature scheme, a message to be signed is based on the power of 2. The message is divided into l digits. Each digit of the message is signed individually. The signature is the corresponding entries of private box with 1's binary in each digit encoded by based of 2. The verification consists of checking the

each signature is the preimage of the corresponding public key entries.

In addition, the variant Lamport one-time signature scheme generalizes the Lamport one-time signature scheme. Thus, at encoding the message into the power of 2, the variant Lamport one-time signature scheme will reduce to a Lamport one-time signature scheme.

5.1.2. Efficiency

Lamport one-time signature scheme is not efficient on the length of private and public keys. The variant Lamport one-time signature scheme saves the space storage of the public keys and the signatures. We consider that the length of message is 320 bits and the variant Lamport one-time signature scheme based on 8 and based on 32 respectively. We compare the variant Lamport one-time signature scheme and the Lamport one-time signature scheme in the Table 5.1.

For example, we want to encode the public key based on 32 in the variant Lamport one-time signature scheme. We encode the message based on 32 and get $l=320/32=10$. The number of items in each column is $e+2=7$, since $e=5$ has 6 items and the special item when the message digit is 0. Therefore, we save about $(320-70)/320=78\%$ storage in storage space against the Lamport scheme.

If the length of signature is l in the Lamport one-time signature, it will take size

of $O(l)$ time complexity on verification but the variant Lamport one-time signature scheme, it only takes time complexity of size of $O(\log(n))$ on verification, since it encode on the base of power of 2.

Although, the variant Lamport one-time signature scheme is better than the Lamport one-time signature scheme, it is still not efficient to sign a very long message. We may improve the problem by hashing the message before signing. I.e., to signing message m , first we compute the hash value $\tilde{m} = f(m)$ by using hash function, then we sign the message \tilde{m} by the variant Lamport one-time signature scheme. The result leads that the signature is not increasing with the length of the message to be signed. Thus, the time complexity is $O(1)$ on verification.

We consider using the SHA-series hash function in the variant Lamport one-time signature scheme. The public key requires 35 items for signing a long message. If we want sign a message with the length of 320 bits. It will take 11% storage for the public keys against the Lamport scheme.

Table 5.1. Comparison of the variant Lamport one-time signature scheme and
Lamport one-time signature scheme

	# of public key items	# of signature items	# of verification items
Lampor signature	320	160	160
Variant Lamport one-time signature scheme based on 8	200	100	100
Variant Lamport one-time signature scheme based on 32	70	35	35

5.2. More Improving on Lamport One-Time Signature

In this section, we develop the Lamport- t one-time signature scheme ($L(t)$ for short). The scheme is more efficient than the above scheme.

5.2.1 The Definition of Lamport-t scheme

The $L(t)$ includes three algorithms, key generation, signature and verification, we define it as follows:

Definition 5.1

Suppose that $h : Y \rightarrow Z$ is a one-way hash function.

Let $T = \{0, 1, \dots, 2^t - 1\}$ where t is a dimensional parameter and the message m be the length of n bits and $k = \left\lceil \frac{n}{t} \right\rceil$ (where $\lceil \cdot \rceil$ is a ceiling function).

(Key generation)

Step 1. Select $2^t k$ elements $y_{i,j} \in Y$ at random with $1 \leq i \leq k$ and $j \in T$ where

k is the length of message based on 2^t .

Step 2. Compute $z_{i,j} = h(y_{i,j})$ for all i, j .

The key K consists of the $2^t k$ y 's and $2^t k$ z 's. The private key SK box and the public key PK box are as follows:

$$SK = \begin{pmatrix} y_{1,0} & y_{2,0} & \cdots & y_{k,0} \\ y_{1,1} & y_{2,1} & \cdots & y_{k,1} \\ \vdots & \vdots & \vdots & \vdots \\ y_{1,2^t-1} & y_{2,2^t-1} & \cdots & y_{k,2^t-1} \end{pmatrix}$$

$$PK = \begin{pmatrix} z_{1,0} & z_{2,0} & \cdots & z_{k,0} \\ z_{1,1} & z_{2,1} & \cdots & z_{k,1} \\ \vdots & \vdots & \vdots & \vdots \\ z_{1,2^t-1} & z_{2,2^t-1} & \cdots & z_{k,2^t-1} \end{pmatrix}$$

(Signature)

To sign a tk -bit message $m = m_1 \dots m_t m_{t+1} \dots m_{2t} \dots m_{kt}$, do the following steps:

Step 1. Encode message by base 2^t as $m = (u_1 u_2 \dots u_k)_{2^t}$ where

$$u_i = (m_{it+1} m_{it+2} \dots m_{it+t}) \text{ and } 0 \leq i \leq k-1.$$

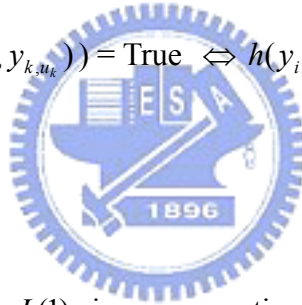
Step 2. Select corresponding entries from the private key box as the signature on the message m . The signature is the following definition.

$$\text{sig}(m) = (y_{1,u_1}, y_{2,u_2}, \dots, y_{k,u_k}).$$

(Verification)

To verify signature $(y_{1,u_1}, y_{2,u_2}, \dots, y_{k,u_k})$ on message m , anyone can check by the follows:

$$\text{vf}(m, (y_{1,u_1}, y_{2,u_2}, \dots, y_{k,u_k})) = \text{True} \Leftrightarrow h(y_{i,u_i}) = z_{i,u_i} \text{ for } 1 \leq i \leq k.$$



□

In the case of $t=1$, the $L(1)$ is a conventional Lamport one-time signature scheme. Thus, the $L(t)$ is another generalized Lamport one-time signature scheme.

A message to be signed is grouped by k 's columns. Each column of message is signed individually. If the i^{th} column of the message is u_i which is encoded by the base of 2^t , then the i^{th} column of the signature is the corresponding value of the private key in the private key box.

For example $m = (3E\dots 0)_{16}$, we select the corresponding entry in the private key box as $(y_{1,3}, y_{1,14}, \dots, y_{k,0})$

$$\begin{aligned}
\text{sig}(3E\dots 0) &= \left\{ \begin{array}{cccc} y_{1,0} & y_{2,0} & \cdots & [y_{k,0}] \\ \vdots & y_{2,1} & \cdots & y_{k,1} \\ [y_{1,3}] & \vdots & \cdots & \vdots \\ \vdots & [y_{2,14}] & \cdots & y_{k,14} \\ y_{1,15} & y_{2,15} & \cdots & y_{k,15} \end{array} \right\} \\
&= (y_{1,3}, y_{1,14}, \dots, y_{k,0})
\end{aligned}$$

The verification is checking that the each element in the column is the preimage of the public key in the public box by the hash function h .

Let us consider a message with length n bits and the parameter t such that n is divisible by t . The results of comparison of the conventional Lamport one-time signature scheme i.e. $L(1)$ and the $L(t)$ on the storage space are as the Table 5.2. The $L(t)$ saves the storage of the signature by t times, but increases the storage of the privates and public keys by $\frac{2^{t-1}}{t}$ against $L(1)$. In case of the $L(2)$, the size of the signature is the half of the Lamport one-time signature scheme, but the number of public keys is the same as $L(1)$.

Table 5.2. Comparison of the $L(1)$ and $L(t)$ with message n bits

Scheme	Signature items	Private keys	Public keys
Lamport's scheme $L(1)$	n	$2n$	$2n$
$L(2)$	$\frac{n}{2}$	$2n$	$2n$
$L(t)$	$\frac{n}{t}$	$2^t \frac{n}{t}$	$2^t \frac{n}{t}$

Note: n : The length of the message bits.

t : The dimensional parameter.

$k = \left\lceil \frac{n}{t} \right\rceil$ (where $\lceil \cdot \rceil$ is a ceiling function).

5.2.2. Security analysis and efficiency

The adversary attempts to forge the $L(t)$. If f is a one-way function with a non-negligible function $\varepsilon(\cdot)$, First, we show the adversary how to invert the $L(t)$ scheme. Assume the adversary forge the signature with probability δ . From the column of the signature, the terms selected from the private key box are about 2^t so that to break the signature is with probability $2^t \delta$. We know the hash function f with a non-negligible function, such that we have $2^t \delta < \varepsilon$ and therefore $\delta < \frac{2\varepsilon}{2^t}$. To break the $L(t)$ function is with probability $\frac{2\varepsilon}{2^t}$.

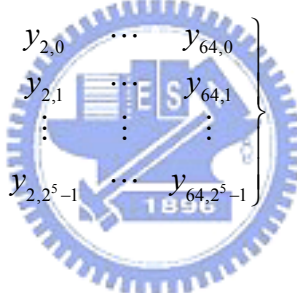
We consider using the SHA-1 hash function in the proposed scheme and the message length $n=320$ such as $t=5$ and $k=64$. The key pair $(SK_{L(5)}, PK_{L(5)})$ of $L(5)$ is as follows:

$$SK_{L(5)} = \begin{Bmatrix} y_{1,0} & y_{2,0} & \cdots & y_{64,0} \\ y_{1,1} & y_{2,1} & \cdots & y_{64,1} \\ \vdots & \vdots & \vdots & \vdots \\ y_{1,2^5-1} & y_{2,2^5-1} & \cdots & y_{64,2^5-1} \end{Bmatrix}$$

$$PK_{L(5)} = \left\{ \begin{array}{cccc} z_{1,0} & z_{2,0} & \cdots & z_{64,0} \\ z_{1,1} & z_{2,1} & \cdots & z_{64,1} \\ \vdots & \vdots & \vdots & \vdots \\ z_{1,2^5-1} & z_{2,2^5-1} & \cdots & z_{32,2^5-1} \end{array} \right\}$$

We use $L(5)$ to compare to the Lamport one-time signature scheme ($L(1)$) in the Table 5.3.

To improve the size of the public keys, we use the $L(5)$ with hash chains. A hash chain is to compute a hash value iteratively. After acting of the hash chain, the $L(5)$ is as follows:

$$SK_{L(5)_hash} = \left\{ \begin{array}{cccc} y_{1,0} & y_{2,0} & \cdots & y_{64,0} \\ y_{1,1} & y_{2,1} & \cdots & y_{64,1} \\ \vdots & \vdots & \vdots & \vdots \\ y_{1,2^5-1} & y_{2,2^5-1} & \cdots & y_{64,2^5-1} \end{array} \right\}$$


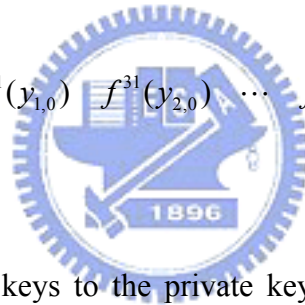
$$= \left\{ \begin{array}{cccc} y_{1,0} & y_{2,0} & \cdots & y_{64,0} \\ f(y_{1,0}) & f(y_{2,0}) & \cdots & f(y_{64,0}) \\ \vdots & \vdots & \vdots & \vdots \\ f^{31}(y_{1,0}) & f^{31}(y_{2,0}) & \cdots & f^{31}(y_{64,0}) \end{array} \right\}$$

$$PK_{L(5)_hash} = \{f^{31}(y_{1,0}) \quad f^{31}(y_{2,0}) \quad \cdots \quad f^{31}(y_{64,0})\}$$

We may improve the long message by hashing the message before signing. Use the SHA-1 for the message 320 bits and shorten the message to 160 bits. The private keys and the public keys will reduce 50%. The key pair is as follows:

$$SK_{L(5)_hash_long} = \begin{Bmatrix} y_{1,0} & y_{2,0} & \cdots & y_{32,0} \\ f(y_{1,0}) & f(y_{2,0}) & \cdots & f(y_{32,0}) \\ \vdots & \vdots & \vdots & \vdots \\ f^{31}(y_{1,0}) & f^{31}(y_{2,0}) & \cdots & f^{31}(y_{32,0}) \end{Bmatrix}$$

$$PK_{L(5)_hash_long} = \{f^{31}(y_{1,0}), f^{31}(y_{2,0}), \cdots, f^{31}(y_{32,0})\}$$



We adopt concept of fly keys to the private key generation. When signing the message, we iterate to use hash function at each row and choice corresponding items.

The private keys are as follows:

Table 5.3. The iterations of the private keys of $L(5)$

Initialization	$\{y_{1,0} \ y_{2,0} \ \cdots \ y_{32,0}\}$
First iteration	$\{y_{1,1} \ y_{2,1} \ \cdots \ y_{32,1}\}$
Second iteration	$\{y_{1,2} \ y_{2,2} \ \cdots \ y_{32,2}\}$
...	...
Final iteration	$\{y_{1,31} \ y_{2,31} \ \cdots \ y_{32,31}\}$

The signature requires 64 items in the $L(5)$. It improve 90% storage compared to the $L(1)$. Thus, the public keys improve 95% against the Lamport scheme. However, the private keys are still required a large amount of items. We proposed the $L(5)$ with hash chains for signing long messages to reduce the private keys to 1024 items. In practical, we can use the fly keys to reduce the storage space of the private keys such that we can improve the storage space to about 95%.

Table 5.4. The comparison of variant $L(t)$ s with message 320 bits

Scheme ↓	Signature items	Private key items	Public key items
Lamport's scheme ($L(1)$)	320	640	640
$L(5)$	64	2048	2048
$L(5)_{hash}$	64	2048	64
$L(5)_{hash_long}$	32	1024	32
$L(5)_{hash_long_flykey}$	32	32	32
Improvement	90%	95%	95%

Note: $L(5)_{hash} : L(5)$ with hash chains

$L(5)_{hash_long} : L(5)$ with hash chain for signing long messages

$L(5)_{hash_long_flykey} : L(5)$ with hash chain for signing long message by fly keys



5.2.3. Comparison

The Bos-Chaum scheme [BC93] is that signatures are shorter than with the Lamport scheme. Suppose we want to sign a n -bits message and we choose k larger enough so that [Sti02]

$$2^n \leq \binom{2k}{k}$$

We need to satisfy the above inequality. If we estimate the binomial coefficient [Sti02]

$$\binom{2k}{k} = \frac{(2k)!}{(k!)^2}$$

using Stirling's formula, we obtain the inequality

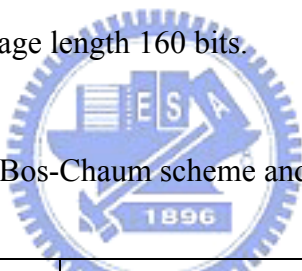
$$n \leq 2 - \frac{\log_2(n\pi)}{2}.$$

(Note: Here, the parameter n and k are swapped to [Sti02])

The Bos-Chaum scheme requires an injective function that associates a k -subset of a $2k$ -set with each possible binary n -tuple.

The comparison of Bos-Chaum scheme and $L(t)$ with the message 160 bit is in Table 5.5. We choose the message length 160 bits.

Table 5.5. The comparison of Bos-Chaum scheme and $L(t)$ with the message 160 bits



Item	Bos-Chaum scheme	<i>L(t)_hash_long_flykey</i>
Message bits	160	160
Public key items	166	32
Private key items	83	32
Signature items	83	32

We can take $k=83$ so that the public key items are 166 and the private items are 83 in the Bos-Chaum scheme. Our scheme is more efficient than the Bos-Chaum scheme and does not need the injection function.

5.3. One-time Signature Scheme Applied on Proxy

Signature

The one-time signature proposed above could be used in proxy signature. We using one-time signature design a proxy signature and the protocol are as the follows.

The system parameter tuple (p, q, g) is defined in Section 2.2.4. Suppose a message with length n bits and the parameter t such that n is divisible by t .

The original signer 'Alice' selects 2^k random numbers as private key box SK_A and the corresponding public key box PK_A with the public keys $z_{i,j} = f^{(2)}(w_{i,j}) \bmod p$ for $1 \leq i \leq k$ and $0 \leq j \leq 2^t - 1$ where function is a one way hash function and

$$f^{(n)} = \overbrace{f \circ f \dots \circ f}^n.$$



(Key generation)

$$SK_A = \begin{Bmatrix} w_{1,0} & w_{2,0} & \dots & w_{k,0} \\ w_{1,1} & w_{2,1} & \dots & w_{k,1} \\ \vdots & \vdots & \vdots & \vdots \\ w_{1,2^t-1} & w_{2,2^t-1} & \dots & w_{k,2^t-1} \end{Bmatrix}$$

$$PK_A = \begin{Bmatrix} z_{1,0} & z_{2,0} & \dots & z_{k,0} \\ z_{1,1} & z_{2,1} & \dots & z_{k,1} \\ \vdots & \vdots & \vdots & \vdots \\ z_{1,2^t-1} & z_{2,2^t-1} & \dots & z_{k,2^t-1} \end{Bmatrix}$$

(Proxy generation and delivery - Executed by Alice and Bob interactively)

The protocol does the following steps.

Step 1. Alice computes $y_{i,j} = f(w_{i,j})$ for $1 \leq i \leq k$ and $0 \leq j \leq 2^t - 1$ to create a proxy key box DK_A as follows:

$$DK_A = \begin{pmatrix} y_{1,0} & y_{2,0} & \cdots & y_{k,0} \\ y_{1,1} & y_{2,1} & \cdots & y_{k,1} \\ \vdots & \vdots & \vdots & \vdots \\ y_{1,2^t-1} & y_{2,2^t-1} & \cdots & y_{k,2^t-1} \end{pmatrix}$$

Step 2. Alice forwards DK_A to the designed proxy signer Bob in a secure manner.

Step 3. Bob checks the validation of proxy key box by

$$z_{i,j} \stackrel{?}{=} f(y_{i,j}) \text{ for } 1 \leq i \leq k \text{ and } 0 \leq j \leq 2^t - 1.$$

If it holds, accept it else reject it.

(Proxy signature signing and verification)

To sign the message m the proxy signer should do the Lamport-t signature scheme and use the Lamport-t verification scheme.

The above proposed scheme is a proxy-unprotected signature scheme. The original signer can forge

The proposed scheme is explicitly use a function f based on discrete logarithms [MVS96]. The original signer derived proxy keys from her private keys by function f such that the proxy signer cannot derive the proxy signer's private keys.

5.4. Discussion

We have proposed two generalized Lamport one-time signature schemes which can save storage space. The proposed schemes are used to sign a long message by hashing the message before signing to make the proposed schemes more efficient.

We expect that our schemes can be used to build more operative one-time signature schemes. The one-time signature applied on proxy signature satisfies the properties of proxy signature. Furthermore, we design a novel proxy signature scheme using proposed one-time signature schemes. This novel proxy signature scheme still needs large storage space unless using fly-key to reduce storage space. But, the fly-key needs enormous computation. It is trade-off problem remained as an open problem.

One-time proxy signature could be applied on mobile agents which are autonomous software entities that are able to migrate across different execution environment. An agent guarantees the security of customer's secret key by transmitting the data without transferring the signature function of the agent. For that proxy signature will be adapted regarding the secret computation of the secret key.

We compare the Kim's one-time proxy [KB⁺01] and Wang's one-time proxy signature [WP03] to the proposed scheme in Table 5.6. The Kim's one-time proxy scheme is based on Discrete logarithm which less efficient than the other two schemes.

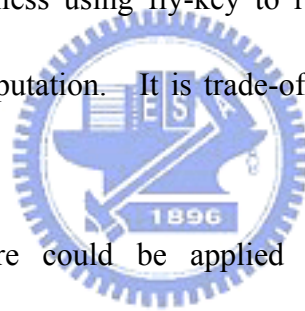
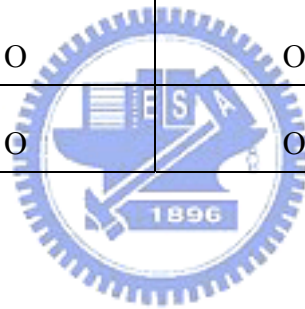


Table 5.6. The comparison of one-time proxy signatures

Property ↓	Kim one-time proxy signature	Wang one-time proxy signature	Proposed scheme
Cryptosystem	Discrete Logarithm	Hash Function	Hash Function
Proxy generation	Schnorr signature	Oblivious Transfer	Double hash
Signature scheme	Fail-stop signature	Lamport scheme	Lamport-t scheme
Verifiability	O	O	O
Unforgeability	O	O	O
Identifiability	O	O	O
Undeniability	O	O	O



Chapter 6 Conclusion

6.1. Conclusion

In this dissertation, we provide a new strong proxy signature scheme in DSA, a strong proxy signature scheme with forward secure property and a one-time proxy signature. This new strong proxy signature scheme in DSA satisfies all the requirements Lee, et al proposed. It provides another method to practice proxy signature, and we also design a procedure to apply it in the PKIs. The second scheme suggests a stronger proxy signature scheme and it combines strong proxy signature with forward secure property. It is more attacker-resistant because even a secret key of a proxy signer is compromised at some time period. The signatures which proxy signer signed for the original person still valid provided that the signatures are signed before key exposure. However, this is not sufficient for the use of a proxy signature. The key exposure problem in distributed environments is also a serious problem against the security of a strong proxy signature scheme. For this reason, we can adopt a strong proxy signature scheme with forward secure property to lengthen the lifetime of a digital signature.

On the other way, we have proposed the generalized Lamport- t one-time signature scheme and apply it on the proxy signature. We use the concepts of hashing chain, signing long message and fly key to improve the storage space of key and get great

results. We expect that our scheme can be used to build more operative one-time signature schemes. In case of the base scheme $L(1)$ of the proposed scheme, the proposed scheme is a conventional Lamport one-time signature scheme.

6.2. Future Work

XML signatures are digital signatures designed for use in XML transactions. The standard defines a schema for capturing the result of a digital signature operation applied to arbitrary data. Like non-XML-aware digital signatures (e.g., PKCS), XML signatures add authentication, data integrity, and support for non-repudiation to the data that they sign. However, unlike non-XML digital signature standards, XML signature has been designed to both account for and take advantage of the Internet and XML. A fundamental feature of XML signature is the ability to sign only specific portions of the XML tree rather than the complete document. The different components are authored at different times by different parties and each signing only those elements relate to itself. This flexibility will be suitable to be use in the proxy signer. The original signer and the proxy signer have the signatures respectively. Those results can be applied on electronic commerce.

In other way, a proxy signature can be used into a mobile agent who can be applied in the electronic commerce. Mobile agents are autonomous software entities

which are able to migrate across different execution environments. Mobility and autonomy make permanent connections unnecessary. There are following fundamental problems of executing mobile code. (1) Code and execute integrity, (2) Code privacy, and (3) Computing with secrets in public. Mobile agents are suitable for providing low-bandwidth connections and asynchronous communications. So far, the relative discussions on proxy signature are rare, and not applicable. The issue is a good interesting problem to research.



References

- [AF99] C. Adms, S. Farrell, “Internet X.509 public key infrastructure certificate management protocols,” March 1999.
- [AMN01] M. Abdalla, S. Miner and C. Namprempre, “Forward-secure threshold signature schemes,” LNCS 2020, Springer-Verlag, CT-RSA 2001, pp. 143-158, Apr. 2001.
- [And99] R. Anderson, Invited lecture, Fourth Annual Conference on Computer and Communications Security, ACM, 1997.
- [ANSI99] ANSI X9.63, “Public key cryptography for the financial services industry: Key agreement and key transport using elliptic curve cryptography,” Jan 1999.
- [AR00] M. Abdalla and L. Reyzin “A new forward-secure digital signature scheme”, Asiacrypto, pp. 116 -129, 2000.
- [BC⁺01] M. Burmester, V. Chrissikopoulos, P. Kotzanikolaou and E. Magkos, “Strong forward security”, Proceedings of the Sixteenth international conference on Information security, Kluwer International Federation For Information Processing Series (IFIP), pp. 109-122, 2001.
- [BC93] Jurjen N. Bos, David Chaum, “Provable unforgeable signatures,” Advances in Cryptology – Crypto’92, LNCS 740, Springer-Verlag, pp. 31-53, 1993.
- [BM99] M. Bellare and S. Miner, ”A forward-secure digital signature scheme,” Advances in Cryptology - Crypto’99 Proceedings, LNCS 1666, Springer-Verlag, pp. 431-438, Sep. 1999.
- [BPH02] L. Bassham, W. Polk, R. Housley, ”Algorithms and Identifiers for the Internet X.509 public key infrastructure certificate and Certificate Revocation List (CRL) profile,” RFC3279, April 2002.
- [CCH03] Tzer-Shyong Chen, Yu-Fang Chung, and Gwo-Shiuan Huang, “Efficient proxy multisignature schemes based on the elliptic curve cryptosystem,” Computers & Security, vol 22, no 6, pp 527-534,2003.
- [CF⁺03] S. Chokhan, W. Ford, R. Sabett, C. Merill, S. Wu, “ Internet X.509 public key infrastructure certificate policy and certificate practices framework,” RFC3647, November 2003.
- [Cha83] D. Chaum, “Blind signatures for untraceable payments,” Advances in

Cryptology - Proceedings of Crypto '82, pp. 199-203, 1983.

- [ElG85] T. ElGamal, "A public key cryptosystem and a signature scheme based on discrete logarithms," IEEE Transactions on Information Theory, vol. 31, no. 4, pp. 469-472, July 1985.
- [GQ88] L.C Guillou, J.J. Quisquater, "A paradoxical identity-based signature scheme resulting from zero-knowledge," Advances in Cryptology CRYPTO'88 LNCS Vol.403, pp.216-231, Aug.1988.
- [HS97] Stuart Haber and W. Scott Stornetta, "Secure names for bit-strings," Proceedings of the 4th ACM conference on Computer and Communication Security, April 1997.
- [HTT04] Min-Shiang Hwang, Shiang-Feng Tzeng, and Chwei-Shyong Tsai, "Generalization of proxy signature based on elliptic curves," Computer Standards & Interfaces, vol.26, pp.73-84, March 2004.
- [HWW01] Chien-Lung Hsu, Tzong-Sun Wu, Tzong-Chen Wu, "New nonrepudiable threshold proxy scheme with known signers," The Journal of System and Software 58, pp. 119-124, 2001.
- [KB⁺01] H. Kim, J. Baek, B. Lee, and K. Kim, "Secret computation with secrets for mobile agent using one-time proxy signature", SCIS'2001, 14C-3, pp. 845-850, Osio, Japan, 2001.
- [KPW97] S. Kim, S. Park, and D. Won, "Proxy signatures, revisited," Proceedings of ICICS'97, Springer-Verlag, LNCS 1334, pp. 223-232, 1997.
- [Kra00] H. Krawczyk. "Simple Forward-Secure Signatures From Any Signature Scheme ", Proceedings of the 7th ACM Conference on Computer and Communications Security, ACM Press, pp.108-115, 2000.
- [LA03] S. Lal and A. K. Awasthi, "Proxy blind signature scheme," to appear in Journal of Information Science and Engineering Cryptology ePrint Archive, Report 2003/072.
- [Lam79] L. Lamport, "Constructing digital signatures from a one-way function," Technical Report CSL-98, SRI International, 1979.

- [LC03] Wei-Bin Lee and Tzung-Her Chen, "Constructing a proxy signature scheme based on existing security mechanisms," *Information & Security International Journal*, vol 12, no. 2, pp.250-258, 2003.
- [LHW98] Narn-Yih Lee, Tzonelih Hwang, Chih-Hung Wang, "On Zhang's nonrepudiable proxy signature schemes," *ACISP 1998*, pp. 415-422, 1998.
- [LK99] B. Lee, and K. Kim, "Strong proxy signatures," *IEICE Trans. Fundamentals*, vol. E82-A, no.1, pp.1-11, Jan 1999.
- [LKK₁01] B. Lee, H. Kim and K. Kim, "Strong proxy signature and its applications," *Proceedings of SCIS 2001*, 11B-1, pp. 603-608, 2001.
- [LKK₂01] B. Lee, H. Kim, and K. Kim, "Secure mobile agent using strong non-designated proxy signature", *Proceedings of ACISP2001*, LNCS vol. 2119, Springer-Verlag, pp. 474-486, 2001.
- [LKK₃01] Byoungcheon Lee, Heesun Kim, and Kwangjo Kim, "Secure mobile agent using strong non-designated proxy signature," *LNCS*, 2001.
- [LTH03] Li-Hua Li, Shiang-Feng Tzeng and Min-Shiang Hwang, "Generalization of proxy signature-based on discrete logarithms," *Computer & Security*, vol. 22, no. 3, pp. 245-255, 2003.
- [MEE00] Elsayed Mohammed, A. E. Emarah and Kh. El-Shennawy, "A blind signature scheme based on ElGamal signature," *17th National Radio Science Conference*, Egypt, 2000.
- [Men93] Alfred Menezes, *Elliptic Curve Public Key Cryptosystems*, Kluwer Academic Publishers, 1993.
- [Mer87] R. C. Merkle, "A certified digital signature based on a conventional function," *Advances in Cryptology - Crypto'87*, LNCS 293, pp. 369-378, 1987.
- [MUO₁96] M. Mambo, K. Usuda, and E. Okamoto, "Proxy signatures: delegation of the power to sign messages, " *IEICE Trans. Fundamentals*, vol. E79-A, no.9, pp.1338-1354, 1996.
- [MUO₂96] M. Mambo, K. Usuda, and E. Okamoto. "Proxy signatures for

- delegating signing operation,” In Proceedings of the 3rd ACM Conference on Computer and Communications Security (CCS), 48C57. ACM, 1996.
- [MVS96] Alfred J. Menezes, Paul C. Van, Oorschot and Scott A. Vanstone, Handbook of Applied Cryptography, CRC Press, 1996.
- [Neu93] B. Clifford Neuman, “Proxy-based authorization and accounting for distributed systems,” International Conference on Distributed Computing Systems 1993.
- [NIST00] NIST. “Digital Signature Standard (DSS),” Federal Information Processing Standards Publication 186, November 1994. Revision (To include ECDSA) 186-2, January, 2000.
- [NIST02] NIST. “Secure hash standard,” Federal Information Processing Standards Publication FIPS PUB 180-2, Aug. 2002.
- [OS91] H. Ong and C. Schnorr, “Fast signature generation with a Fiat Shamir-like scheme”, Advances in Cryptology - Eurocrypt'90, LNCS vol.473, Springer Verlag , pp. 432-440, 1991.
- [Rab79] M. O. Rabin, “Digitalized signatures,” Foundations of Secure Communication, Academic Press, pp. 155-168, 1979.
- [RSA78] R. L. Rivest, A. Shamir, L. Adelman, “A method for obtaining digital signature and public key cryptosystem,” Comm. ACM 21 (2),pp. 120-126, 1978.
- [RSA00] RSA Laboratories, “PKCS #10: Certification request syntax specification,” RFC 2986, Version 1.7, November 2000.
- [Sch95] L.A.M. Schoenmakers, “An efficient payment system withstanding parallel attacks,” CWI, CS-R9522, 1995.
- [Sch00] B. Schneier, Applied cryptography, John Wiley & Sons, 2000.
- [SH04] H. M. Sun and B. T. Hsieh, “On the Security of the some proxy blind signature scheme,” Proceedings of the second workshop on Australasia Information Security, Data Mining and Web Intelligence, and Software Internationalisation, vol 32, pp 75-78, 2004.
- [SLH99] H. M. Sun, N. Y. Lee, and T. Hwang, " Threshold proxy signatures" IEE proceedings – Computers and Digital Techniques, vol. 146, no. 5, pp.

259-263, 1999.

- [Son01] D. Song, "Practical forward secure group signature schemes," 8th ACM Conference on Computer and Communications Security (CCS-8), 2001.
- [Sti02] Douglas R. Stinson, *Cryptography Theory and Practice*, 2nd, CRC Press, 2002.
- [Sun99] H. M. Sun, "An efficient nonrepudiable threshold proxy signature scheme with known signers," *Computer Communications*, vol. 22, no. 8, pp. 717-722, New York, IPC Science and Technology Press, 1999.
- [Sun00] H. M. Sun, "On the Design of Time-Stamped Proxy Signatures with Traceable Receivers", *IEE Proceedings - Computers and Digital Techniques*, accepted, London, IEE Press, 2000.
- [TLT02] Z. Tan, Z. Liu and C. Tang, "Digital proxy blind signature schemes based on DLP and ECDLP," *MM Research Preprints*' no. 21, MMRC, AMSS Academia Sinica, Beijing, pp. 212-217, 2002.
- [TJC03] Yuh-Min Tseng, Jinn-Ke Jan and Hung-Yu Chien, "Digital signature with message recovery using self-certified public keys and its variants," *Applied Mathematics and Computation* vol. 136, pp. 203-214, 2003.
- [SW02] K. Shum and V. K. Wei, "A strong proxy signature scheme with proxy signer privacy protection," *IEEE Eleventh International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises*, pp 55 –56, June 2002.
- [WP03] H. Wang and J. Pieprzyk, "Efficient one-time proxy signatures," *Asiacrypto'03*, LNCS 2894, pp. 507-522, 2003.
- [WS96] T. C. Wu and H. S. Sung, "An improved one-time digital signature scheme based on one-way function," *Journal of Information science and Engineering*, vol.12 no.3, pp. 387-395, 1996.
- [YBX00] Lijang Yi, Guoqiang Bai and Guizhen Xiao, "Proxy multi-signature scheme: A new type of proxy signature scheme," *Electronic Letters* 16th, vol. 36, no. 6, pp. 527-528, March 2000.
- [Zha97] K. Zhang, "Threshold proxy signature schemes," 1997 Information Security workshop, Japan, pp. 191-199, September, 1997.

- [Zho98] K. Zhong, "Efficient protocols for signing routing messages," Proceedings of NDSS, 1998.
- [ZW⁺04] Jianhong Zhang, Qianhong Wu, Jilin Wang, Yumin Wang, "An improved nominative proxy signature scheme for mobile communication," Proceedings of the 18th International Conference on Advance Information Networking and Application (AINA'04), 2004.

