國立交通大學

資訊科學與工程研究所

博士論文

具代理授權特性的可轉換鑑別加密方法

Convertible Authenticated Encryption Schemes with Proxy Delegation

研究生：林韓禹

指導教授：黃世昆、吳宗杉、葉義雄　教授

中華民國九十九年十二月

具代理授權特性的可轉換鑑別加密方法
Convertible Authenticated Encryption Schemes with Proxy Delegation

研究生：林韓禹　　Student： Han-Yu Lin

指導教授：黃世昆　　Advisors：Shih-Kun Huang
吳宗杉　　Tzong-Sun Wu
葉義雄　　Yi-Shiung Yeh

國立交通大學
資訊科學與工程研究所
博士論文

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

Computer Science

December 2010

Hsinchu, Taiwan, Republic of China

中華民國九十九年十二月

具代理授權特性的可轉換鑑別加密方法

學生：林韓禹

指導教授：黃世昆博士
吳宗杉博士
葉義雄博士

國立交通大學資訊科學與工程研究所博士班

摘　　要

在日常的生活中，數位簽章及公開金鑰加密是保護線上交易安全的二種常用機制。前者確保鑑別性與不可否認性，後者則保障機密性。

欲提供密碼方法同時具備機密性與鑑別性，鑑別加密方法是一較佳的選擇，與直接簽章再加密的方式相較，鑑別加密法可提升效率與降低通訊成本。此方法允許簽署者產生一鑑別加密訊息，使得僅特定驗證者有能力來解密此訊息並驗證其對應的簽章。可轉換鑑別加密方法不僅具備上述所提的特性，當發生事後的否認爭議時，更提供額外的簽章轉換機制使任意人信服簽署者的不誠實。

代理簽章方法允許一位被授權者，稱為代理簽署者，根據事先定義好的簽署策略，代表原始簽署者產生合法的代理簽章。在本論文中，作者提出三種具代理授權特性的可轉換鑑別加密方法，分別植基於 RSA、CDHP、BDHP 不同的密碼假設難題。所提之方法允許一位代理簽署者代表原始簽署者產生一合法的鑑別加密訊息，同時僅有一位特定接收者有能力解密並驗證其對應的代理簽章。由於轉換後的原始代理簽章會在訊息回復與驗證簽章的過程中被運算出來，因此簽章轉換的程序相當簡單，而且可由特定驗證者在不需額外計算或通訊成本的情況下獨立完成。我們也提出一個群體導向的變形方法，其允許一個由 n 位原始簽署者組成的群體授權他們的

簽署能力給一位代理簽署者，來代表此原始簽署群體產生鑑別加密訊息。為了方便大訊息的加密，作者進一步提出藉由將一個大訊息切割為多個小訊息區塊的具訊息鏈結的變形方法。

與之前的文獻相比，所提的方法不僅有較低的計算成本，同時亦提供較佳的功能性。此外，在抵抗調整式選擇密文攻擊的機密性安全需求與抵抗調整式選擇訊息攻擊的不可偽造性安全需求，也在 random oracle 模型下證明。

關鍵字：可轉換、鑑別加密、代理授權、機密性、公開金鑰系統

Convertible Authenticated Encryption Schemes with Proxy Delegation

Student：Han-Yu Lin

Advisors：Dr. Shih-Kun Huang
Dr. Tzong-Sun Wu
Dr. Yi-Shiung Yeh

Institute of Computer Science and Engineering
National Chiao Tung University

ABSTRACT

In modern daily life, digital signatures and public key encryptions are two commonly applied mechanisms for protecting the security of on-line transactions. The former ensures authenticity and non-repudiation while the latter guarantees confidentiality.

To simultaneously provide cryptographic schemes with confidentiality and authenticity, an authenticated encryption (AE) scheme is a better alternative for promoting efficiency and reducing communication overheads as compared to the straightforward sign-then-encrypt method. Such schemes allow a signer to produce an authenticated ciphertext, such that only a designated recipient has the ability to decrypt the ciphertext and verify its corresponding signature. Convertible authenticated encryption (CAE) schemes not only inherit the characteristic mentioned above, but also provide additional signature conversion mechanism to convince anyone of signer's dishonesty when a later dispute occurs.

Proxy signature schemes allow an authorized person called proxy signer to generate proxy signatures on behalf of an original signer according to the predefined signing policy. In this dissertation, the author proposes three CAE schemes with proxy delegation based on different cryptographic assumptions, i.e., RSA, CDHP, BDHP, respectively. The proposed schemes allow a proxy signer to generate a valid authenticated ciphertext on behalf of an original signer and only the intended recipient is capable of decrypting it and verifying the corresponding proxy signature. The signature conversion is rather simple and can be solely done by the designated recipient with neither extra computation costs nor communication overheads, since the converted proxy signature will be derived during the message recovery and signature verification phase. We also present a group-oriented variant which enables an original group consisting of *n* signers to delegate their signing power to a proxy signer such that the latter can generate an authenticated ciphertext on behalf of the former. For facilitating the encryption of a large message, the author further introduces the other variant with message linkages by dividing a large message into many small message blocks.

As compared with previous works, the proposed schemes not only have lower computation costs, but also provide better functionalities. Additionally, the security requirement of confidentiality against indistinguishability under adaptive chosen-ciphertext attacks (IND-CCA2) and that of unforgeability against existential forgery under adaptive chosen-message attacks (EF-CMA) are proved in the random oracle model.

Keywords: convertible, authenticated encryption, proxy delegation, confidentiality, public key system.

Acknowledgements

I would like to dedicate this dissertation to those who once helped me and share the joy with all people who care about me.

It takes me five and a half years to finish this dissertation at NCTU. For my first two years, I appreciate my late advisor, Prof. Yi-Shiung Yeh for his teaching and encouragement. Without asking me to deal with many laboratory affairs, he gave me lots of time to prepare qualification examinations. Prof. Yeh was a kind, sincere and conscientious teacher. Being a good example for students, he devoted himself to the study of security related issues and often worked until night at holidays. He always told us to think about possible research topics from wider and more different perspectives. I will remember what he had taught me forever.

For the last three and a half years, I appreciate Prof. Shih-Kun Huang and Prof. Tzong-Sun Wu for being willing to serve as my advisors. The former supports me to continue my unfinished researches and helps me with many department things at NCTU. The latter is not only my major dissertation advisor, but also my life mentor. He teaches me that always look on the bright side of life and be optimistic when facing any trouble. I sincerely appreciate all they have done for me.

Special thanks go to Prof. Wen-Guey Tzeng, Prof. Shi-Chun Tsai, Prof. Tzong-Chen Wu, Prof. Henry Ker-Chang Chang, Prof. Chu-Hsing Lin and Prof. Chi-Jen Lu for agreeing to serve on my dissertation committee. Without their valuable suggestions, this dissertation would not have been complete. I also thank all laboratory members for the wonderful time we had together. Especially, my senior officemate, Ting-Yu Huang, helped me a lot. Furthermore, a special thank to Prof. Chien-Chao Tseng and the department administrative staff for giving me required assistance throughout the process.

Most importantly, I do not know how to thank my dear families and beloved girl friend in words for their endless patience and unconditional support in either mentality or substances.

Contents

List of Tables

List of Figures

1. Introduction

With the rapid development of electronic commerce (eCommerce), the security of on-line transactions has received great attention. Generally speaking, cryptographic techniques can be adopted to protect communication content over the Internet. Public key encryption [DH76] and digital signature schemes [ElG85, RSA78, NR93] are two fundamental cryptographic mechanisms which primarily aim for providing confidentiality [HWT$^+$04, Jac91] and authenticity [Sta05], respectively. The digital signature scheme can further satisfy the requirement of non-repudiation [Sch98] to prevent signer's dishonesty.

1.1 Motivation

Some applications, however, like contract signings, electronic funds transfer (EFT), on-line auctions and credit card transactions require all the above security requirements simultaneously be achieved. A straightforward way would be sign-then-encrypt [VM97]. Yet, the approach is costly in terms of computation efforts and communication overheads. In some special circumstances, a proxy might be properly delegated to conduct these confidential transactions, e.g., proxy auctions and contract signings by an authorized proxy signer. Consider group-oriented applications such as a joint account owned by two or more individuals. To withdraw money from such an account, all owners must cooperatively sign a withdrawal receipt which can only be verified by the bank teller. In case that account owners are unable to sign personally, they can delegate their signing power to a proxy signer who can legitimately conduct transactions on behalf of them. It thus can be seen that the design of efficient and provably secure cryptographic schemes fulfilling such requirements is crucial and benefits the practical implementation.

1.2 Related Works

Since Diffie and Hellman [DH76] proposed the first public key system based on discrete logarithm problems (DLP) in 1976, public key systems have been extensively studied. In public key cryptosystems [Gir91, RSA78, Sha84], each one has a private key and its corresponding public one. To achieve the security requirements of confidentiality and data integrity [Sta05], one can use a recipient's public key to encrypt messages such that only the designated recipient can decrypt the ciphertext with his own private key. However, it might be even hard for an arbitrator to handle if a sender disclaims having transmitted the encrypted message. A digital signature scheme is applicable for that the signature is generated with a signer's unique private key and thereafter everyone can verify its validity with the signer's public key. It can be seen that only the actual owner of private key can produce a valid signature so as to prevent a dishonest signer from disclaiming, which is referred to as non-repudiation.

In 1994, Horster *et al.* [HMP94] proposed an authenticated encryption (AE) scheme further providing digital signature schemes with the property of confidentiality and only a designated recipient can verify the signature instead of everyone. Since only a designated recipient has the ability to decrypt the ciphertext and verify the corresponding signature, there might be a potential drawback that a signer repudiates his signature. In such circumstance, it is even difficult for an arbitrator to judge who is lying.

To deal with the case of a later dispute over repudiation, Araki *et al.* [AUI99] presented a convertible limited verifier signature scheme. However, the signature conversion of their scheme requires the assistance of signer and will incur extra computation efforts, which is considered to be inefficient and unworkable if a signer is reluctant to cooperate with. Besides, Zhang and Kim [ZK03] also pointed out that Araki *et al.*'s scheme could not withstand a universal forgery attack on an arbitrary chosen message.

In 2002, Wu and Hsu [WH02] proposed a convertible authenticated encryption (CAE) scheme, in which the signature conversion is rather simple and can be solely done by a designated recipient without extra computation efforts or communication overheads. Huang and Chang [HC03] further introduced an enhanced variant in the next year. However, both the Wu-Hsu and the Huang-Chang schemes cannot fulfill the security requirement of confidentiality, i.e., a ciphertext is computationally distinguishable with respect to two candidate messages. To eliminate such security weakness, Lv *et al.* [LWK05] addressed a secure and practical solution. In 2005, Wu *et al.* [WHL25] proposed generalized CAE schemes and adapted them based on elliptic curves [Kob87, Mil85] for facilitating gradually popular applications like smart cards [Hen94, RRK$^+$04, SP02], mobile phones and PDAs. In 2008, Chien [Chi08] proposed a selectively CAE scheme allowing either a signer or a designated recipient to perform signature conversion. In 2009, Lee *et al.* [LHT09] addressed a CAE scheme based on the ElGamal cryptosystem. Considering the RSA cryptosystem, Wu and Lin [WL09] also presented a CAE scheme based on RSA assumption. To fulfill the group-oriented application requirement, in 2008, Wu *et al.* [WHT^{+}08] and Chang [Cha08] proposed convertible multi-authenticated encryption (CMAE) schemes for group communication, respectively. In 2009, Tsai [Tsa09] presented a more efficient CMAE scheme with lower computation costs. Lin and Yeh [LY08] further proposed a threshold CAE scheme allowing any t or more signers to cooperatively generate a valid authenticated ciphertext on behalf of an original signing group. So far, lots of CAE variants [DC06, DCZ05, EA08, HLL^{+}05, LC04, LW08, LWH^{+}07, LWH^{+}08, WC05, WL08a, WL08b, WLC06, WLH^{+}07, ZD04] have been proposed.

In a separate development, Mambo *et al*. [MUO96a, MUO96b] extended the concept of digital signature and introduced the notion of proxy signatures. A proxy signature scheme allows an original signer to delegate his signing power to an authorized person called proxy signer such that the proxy signer can generate a valid proxy signature on behalf of an original one. As to the proxy delegation, it can be categorized into four different kinds as follows:

(i). *Full delegation* [MUO96a, MUO96b]: The proxy signer's signing key is the same as an original signer's private key so that all (proxy) signatures are generated with an identical private key. Consequently, it is difficult to convince any verifier that a proxy signature is indeed generated by the proxy signer. That is to say, it cannot offer secure mechanisms to protect any one of them from being framed by the other.

(ii). *Partial delegation* [MUO96a, MUO96b]: Based on the intractability of some security assumptions, e.g., factorization and discrete logarithm problems, a proxy signature key is computed from an original signer's private key while the latter cannot be derived from the former. Nevertheless, there might be a drawback that it requires an additional revocation protocol, as no information (e.g., the period of validity) is bonded to the delegation. Besides, it is difficult to identify the actual signer for a given signature, since a malicious original signer can easily impersonate a proxy one to forge a valid proxy signature.

(iii). *Delegation by warrant* [Neu93, Var91]: An original signer prepares a warrant containing some necessary proxy information, such as the period of validity and the identifiers of original and proxy signers, and then sends it to the proxy signer as his delegation authorization. The warrant could be viewed as an original signer's signature to convince any verifier of his agreement. However, it requires extra efforts to certify and transmit the warrant, which is costly in terms of computation efforts and communication overheads.

(iv). *Partial delegation with warrant* [KPW97]: This type preserves the merits of partial delegation and delegation by warrant. Equivalent to the second approach, it is computationally infeasible for a proxy signer to derive an original signer's private key from his proxy signature key. Moreover, to certify a warrant and validate a signature can be simultaneously carried out in a single step.

Obviously, the fourth approach, partial delegation with warrant, is more flexible and secure as compared with the first three. Because of its efficiency and security as compared with the other three, the author also adopts partial delegation with warrant to implement the proposed schemes. Up to the present, lots of variations of proxy signatures have been proposed [HC01, HLL00, HS00, HWW01, KPW97, LHW98, LWH02, SLH99, TYH04, WHL08, XC04a, XC04b, YX00]. These schemes can be classified into the five categories according to the signing policy and the number of original and proxy signers as follows:

(i). *Proxy multisignature* [YX00]: A group of two or more original signers delegates the signing power to a proxy signer. Then the proxy signer can generate a multisignature on behalf of the original group.

(ii). *Multi-proxy signature* [CC06, HS00, LWH02, WHL08, XC04b]: An original signer delegates his signing power to two or more proxy signers and all of them must cooperatively sign on behalf of the original signer.

(iii). *Threshold proxy signature* [HLL00, KPW97, LHW07, LHW98, SLH99, WCL+07]: In a (t, n) threshold proxy signature, an original signer delegates his signing power to n proxy signers such that any t or more of them can cooperatively generate a valid signature on behalf of the original signer.

(iv). *Multi-proxy multisignature* [HC01, XC04a]: A group composed of two or more original signers can delegate the signing power to a designated proxy group. All members in the proxy group must cooperatively generate a valid multisignature on behalf of the original group.

(v). *Threshold multi-proxy multisignature* [HWW01, LHL+01, TYH04]: In a (t, n) threshold multi-proxy multisignature, a group comprising two or more original signers can delegate the signing power to *n* proxy signers. Any *t* or more proxy signers can cooperatively generate a valid multisignature on behalf of the original group.

1.3 Our Contributions

In this dissertation, the author elaborates on the merits of CAE schemes and proxy signature schemes to propose three proxy CAE schemes named PCAE-(I), PCAE-(II) and PCAE-(III), respectively. To the best of our knowledge, the proposed PCAE-(I) scheme is the first provably secure one based on RSA assumption. The proposed PCAE schemes allow a delegated proxy signer to generate a valid authenticated ciphertext on behalf of an original signer such that only a designated recipient can recover the message and verify its embedded proxy signature. When the case of a later dispute over repudiation occurs, a designated recipient can solely convert the authenticated ciphertext into a publicly verifiable proxy signature without any computation or communication cost. Moreover, the author also presents two extensions. One is a group-oriented variant allowing one proxy signer to generate an authenticated ciphertext on behalf of an original signing group composed of *n* signers. The other is a variant with message linkages which enables the encryption of large messages. We also prove that the proposed schemes achieve the security requirement of confidentiality against indistinguishability under adaptive chosen-ciphertext attacks (IND-CCA2) and that of unforgeability against existential forgery under adaptive chosen-message attacks (EF-CMA) in the random oracle model. Compared with previous works, the proposed schemes not only have lower computation costs, but also provide better functionalities.

Table 1.3.1 summarizes the functionalities and security proofs among the proposed and related schemes including Elkamshoushy *et al.*' (EAM for short) [EAM06], the Zhang-Dong (ZD for short) [ZD04], Dai *et al.*'s (DYD for short) [DYD03], the Elkamchouchi-Abouelseoud (EA for short) [EA08], Duan *et al.*'s (DCZ for short) [DCZ05], the Li-Chen (LC for short) [LC04] and the Wang-Liu (WL for short) [WL05] schemes.

Table 1.3.2 further summarizes the computation costs in terms of required modular exponentiation computation among the proposed PCAE-(II) and those [EAM06, DYD03, ZD04] based on the computational Diffie-Hellman problem (CDHP).

Table 1.3.3 further summarizes the computation costs in terms of required bilinear pairing and multiplication computation among the proposed PCAE-(III) and those [EA08, DCZ05, LC04, WL05] based on the bilinear Diffie-Hellman problem (BDHP).

Table 1.3.1. Comparisons of functionalities and security proofs

Scheme Item	EAM	ZD	DYD	EA	DCZ	LC	WL	Ours
Against key exposure attack	O	O	O	O	×	O	O	O
Signature conversion	O	O	×	×	×	O	O	O
No conversion cost	×	×	×	×	×	O	O	O
Non-interactive conversion procedure	O	O	×	×	×	O	O	O
Formal Proof	×	×	×	×	×	×	×	O

Table 1.3.2. Comparisons of computation costs in terms of modular exponentiation computation

Scheme Item	EAM	DYD	ZD	PCAE-(II)
Costs for PCG phase*	$5T_e$	$3T_e$	$3T_e$	$2T_e$
Costs for ACG phase	$2T_e$	$2T_e$	$2T_e$	$2T_e$
Costs for SRV phase	$5T_e$	$7T_e$	$4T_e$	$4T_e$
Total Costs	$12T_e$	$12T_e$	$9T_e$	$8T_e$

Remark *: T_e stands for the time for performing one modular exponentiation computation.

Table 1.3.3. Comparisons of computation costs in terms of bilinear pairing and multiplication computation

Scheme Item	EA	DCZ	LC	WL[2]	PCAE-(III)
Costs for PCG phase[1]	$2T_B + 2T_M$	$2T_B + 3T_M$	$3T_B + 4T_M$	$3T_B + 3T_M$	$2T_B + T_M$
Costs for ACG phase	$2T_B + 2T_M$	$2T_B + 4T_M$	$2T_B + 2T_M$	$2T_B + 3T_M$	$2T_B + 2T_M$
Costs for SRV phase	$4T_B + 2T_M$	$4T_B + 4T_M$	$8T_B + 6T_M$	$5T_B + 4T_M$	$4T_B + 2T_M$
Total Costs	$8T_B + 6T_M$	$8T_B + 11T_M$	$13T_B + 12T_M$	$10T_B + 10T_M$	$8T_B + 5T_M$

Remarks: 1. T_B and T_M stands for the time for performing one bilinear pairing and one multiplication computation, respectively.

2. To obtain fair comparison results, we assume that only one proxy signer is

involved in the Wang-Liu scheme.

1.4 Organization of Dissertation

This dissertation consists of seven chapters. Chapter 1 is an introductory section describing the motivation, related works, our contributions and the organization of the dissertation. The rest of others are stated as follows:

In chapter 2, the author reviews some security notions and important cryptographic building blocks with respect to the proposed PCAE schemes.

In chapter 3, the author defines the formal model of PCAE schemes, including involved parties, composed algorithms and its security model.

In chapter 4, the author proposes a PCAE scheme based the RSA assumption, demonstrates its correctness and proves its security in the random oracle model.

In chapter 5, the author introduces a PCAE scheme based on CDHP, demonstrates its correctness and proves its security in the random oracle model. Moreover, a group-oriented variant for facilitating multiuser applications, and a variant with message linkages for benefiting the large message encryption are presented, respectively.

In chapter 6, the author addresses a PCAE scheme based on BDHP, demonstrates its correctness and proves its security in the random oracle model.

Finally, in chapter 7, the author makes a conclusion with regard to the significance of this dissertation and gives the future research.

2. Preliminaries

In this section, we first review some security notions and related cryptographic building blocks. The used notations are stated as follows:

Table 2.1. The used notations

Z_p	integers modulo p
Z_p^*	multiplicative group of integers modulo p
GF(p)	Galois field of p elements
$x \in Z_p$	element x in set Z_p
$x \in_R Z_p$	element x is a random integer in set Z_p
$x \leftarrow Z_p$	sampling element x uniformly in set Z_p
$\#Z_p$	number of elements in set Z_p
a mod b	modulo operation: reminder of a divided by b
$a \mid b$	integer b is divisible by integer a
$a \parallel b$	concatenation of a and b
$\|x\|$	bit-length of integer x, also absolute value of x
$\sum_{i=1}^{n} v_i, \sum_{i \in S} v_i$	sum of values v_i for $i = 1, 2, \ldots, n$, or for $i \in S$
$\prod_{i=1}^{n} v_i, \prod_{i \in S} v_i$	product of values v_i for $i = 1, 2, \ldots, n$, or for $i \in S$
$\log_b x$	logarithm to base b of x
$\oplus$	logical operation XOR
$\neg$	logical operation NOT
$\wedge$	logical operation AND
$\vee$	logical operation OR
$\forall$	for all
Pr[E]	probability of event E occurring

2.1 Security Notions [DK02, BLS03, MOV97, RSA78, Sta05]

We review some essential cryptographic security assumptions with respect to the proposed schemes as follows:

RSA Problem

Let $N = pq$ be the product of two large primes (p, q), and (e, d) two integers satisfying that $\gcd(e, \phi(N)) = 1$ and $ed = 1 \bmod \phi(N)$, where $\phi(\cdot)$ is the Euler totient function and $\phi(N) = (p - 1)(q - 1)$. Given $c = m^e \bmod N$ as input, the RSA problem is to output $m \in Z_N$ satisfying $m = c^d \bmod N$.

RSA Assumption

Let $\mathbf{G}$ be an RSA key generator which takes a security parameter 1^k as its input and outputs (N, e, d, p, q). Given an RSA instance $(N, e, c = m^e \bmod N)$, the advantage for any probabilistic polynomial-time adversary $\mathcal{A}$, every positive polynomial $P(\cdot)$ and all sufficiently large w to solve the RSA problem is at most $1/P(w)$, i.e.,

$$\Pr[\mathcal{A}(N, e, c) = m;\ c \leftarrow m^e \bmod N,\ m \leftarrow Z_N,\ (N, e, d, p, q) \leftarrow \mathbf{G}] \leq 1/P(w).$$

The probability is taken over the uniformly and independently chosen instance with a given security parameter k and over the random choices of $\mathcal{A}$.

Definition 2.1.1. *The (t, ε)-RSA assumption holds if there is no polynomial-time adversary that can solve the RSA problem in time at most t and with the advantage ε.*

Discrete Logarithm Problem; DLP

Let p and q be two large primes satisfying $q \mid p - 1$, and g a generator of order q over GF(p). The discrete logarithm problem is, given an instance (y, p, q, g), where $y = g^x \bmod p$ for some $x \in Z_q$, to derive x.

Discrete Logarithm (DL) Assumption

Let $I_k = \{(p, q, g) \in I \mid |p| = k\}$ with $k \in \mathrm{N}$, where I is the universe of all instances and $|p|$ represents the bit-length of p. For every probabilistic polynomial-time algorithm $\mathcal{A}$, every positive polynomial $P(\cdot)$ and all sufficiently large k, the algorithm $\mathcal{A}$ can solve the DLP with an advantage at most $1/P(k)$, i.e.,

$$\Pr[\mathcal{A}(y, p, q, g) = \mathrm{Log}_{p,\, q,\, g}(y);\ (p, q, g) \leftarrow I_k, y \leftarrow Z_p^*] \le 1/P(k).$$

The probability is taken over the uniformly and independently chosen instance with a given security parameter k and over the random choices of $\mathcal{A}$.

Definition 2.1.2. *The* (t, ε)*-DL assumption holds if there is no polynomial-time adversary that can solve the DLP in time at most t and with the advantage* ε*.*

Computational Diffie-Hellman Problem; CDHP

Let p and q be two large primes satisfying that $q \mid p - 1$ and g a generator of order q over GF(p). The computational Diffie-Hellman problem is, given an instance (p, q, g, g^a, g^b) for some $a, b \in Z_q$, to derive $g^{ab} \bmod p$.

Computational Diffie-Hellman (CDH) Assumption

Let $I_k = \{(p, q, g) \in I \mid |p| = k\}$ with $k \in \mathrm{N}$, where I is the universe of all instances and $|p|$ represents the bit-length of p. For every probabilistic polynomial-time algorithm $\mathcal{A}$, every positive polynomial $P(\cdot)$ and all sufficiently large k, the algorithm $\mathcal{A}$ can solve the CDHP with an advantage at most $1/P(k)$, i.e.,

$$\Pr[\mathcal{A}(p, q, g, g^a, g^b) = g^{ab};\ (p, q, g) \leftarrow I_k,\ a, b \leftarrow Z_q] \le 1/P(k).$$

The probability is taken over the uniformly and independently chosen instance with a given security parameter k and over the random choices of $\mathcal{A}$.

Definition 2.1.3. *The* (t, ε)*-CDH assumption holds if there is no polynomial-time adversary that can solve the CDHP in time at most t and with the advantage* ε*.*

Bilinear Pairing

Let $(\boldsymbol{G}_1, +)$ and $(\boldsymbol{G}_2, \times)$ denote two groups of the same prime order q and e: $\boldsymbol{G}_1 \times \boldsymbol{G}_1 \rightarrow \boldsymbol{G}_2$ be a bilinear map which satisfies the following properties:

(i) ***Bilinearity***:

$e(P_1 + P_2, Q) = e(P_1, Q)e(P_2, Q)$;

$e(P, Q_1 + Q_2) = e(P, Q_1)e(P, Q_2)$;

(ii) ***Non-degeneracy***:

If P is a generator of $\boldsymbol{G}_1$, then $e(P, P)$ is a generator of $\boldsymbol{G}_2$.

(iii)***Computability***:

Given $P, Q \in \boldsymbol{G}_1$, $e(P, Q)$ can be efficiently computed by a polynomial-time algorithm.

Bilinear Diffie-Hellman Problem; BDHP

The BDHP is, given an instance $(P, A, B, C) \in \boldsymbol{G}_1{}^4$ where P is a generator, $A = aP$, $B = bP$ and $C = cP$ for some $a, b, c \in Z_q^*$, to compute $e(P, P)^{abc} \in \boldsymbol{G}_2$.

Bilinear Diffie-Hellman (BDH) Assumption

For every probabilistic polynomial-time algorithm $\mathcal{A}$, every positive polynomial $Q(\cdot)$ and all sufficiently large k, the algorithm $\mathcal{A}$ can solve the BDHP with the advantage at most $1/Q(k)$, i.e.,

$$\Pr[\mathcal{A}(P, aP, bP, cP) = e(P, P)^{abc}; a, b, c \leftarrow Z_q^*, (P, aP, bP, cP) \leftarrow \mathbf{G}_1{}^4] \leq 1/Q(k).$$

The probability is taken over the uniformly and independently chosen instance and over the random choices of $\mathcal{A}$.

Definition 2.1.4. *The* (t, ε)*-BDH assumption holds if there is no polynomial-time adversary that can solve the BDHP in time at most t and with the advantage ε.*

2.2 Designated Verifier Signature Scheme

A digital signature scheme is one fundamental cryptographic technique which primarily aims for providing authenticity and non-repudiation [MWX02]. Since all public keys are either maintained by the system authority (SA) or stored in the public key directory, one can easily obtain the corresponding public key of the other to verify his/her signature. The actual signer thus can not deny his/her generated signature later. However, in some applications such as electronic voting [RN01, Sch99] and electronic auction [JS03, WCL08], the non-repudiation property is not desirable. With an eye to the above requirement, in 1990, Chaum and Antwerpen [CA90] proposed an undeniable signature scheme in which a signer must assist a verifier to validate a generated signature. It is obvious that any third party attempting to verify the signature has to reach an agreement with the signer in advance. That is to say, in an undeniable signature scheme, a signer has completely control over his generated signatures. In 1996, Jakobsson *et al.* [JSI96] came up with the notion of designated verifier proofs and in a sense proposed a designated verifier signature (DVS) scheme. In their scheme, a designated verifier can be convinced of the signer's identity regarding a given signature without the assistance of the actual signer. Yet, a designated verifier can not transfer the proofs to convince any third party, since he is also capable of generating another DVS which is computationally indistinguishable from the received one. In 2003, Wang [Wan03] formalized the notion of DVS scheme and further proposed a so-called strong designated verifier signature (SDVS) scheme in which a designated verifier's private key is directly involved in the validation equation. Consequently, anyone cannot even perform the validation equation without the knowledge of designated verifier's private key. In 2007, Lee and Chang [LC07] further combine SDVS schemes with message recovery signatures. More recently, they [LC09] pointed out that signer's ambiguity could be a vital property of secure SDVS schemes. Namely, even if a signer's private key is compromised, any attacker still cannot identify the actual signer for a given SDVS which has not been received by the designated verifier. Another SDVS scheme satisfying such a property is also proposed in their paper. Nevertheless, they give no formal proof. In 2004, Susilo *et al.* [SZM04] addressed the first identity-based SDVS scheme from bilinear pairings. Since then, several researchers [HSM$^+$08, KBD09, KSS06, ZM08, LW09] have devoted themselves to the design of pairing-based SDVS schemes. However, we find out that none of these schemes could fulfill the property of signer's ambiguity addressed by Lee and Chang [LC09].

Generally speaking, an SDVS scheme should satisfy the following security requirements [SKM03]:

(i) ***Unforgeability:*** It is computationally infeasible for any polynomial-time adversary to forge a valid SDVS without knowing the private key of either the signer or the designated verifier.

(ii) ***Non-Transferability:*** Based on the transcript simulation property in an SDVS scheme, a designated verifier can also generate another SDVS which is computationally indistinguishable from the received one. Therefore, a designated verifier cannot transfer the SDVS to any third party.

(iii) ***Signer's Ambiguity:*** It is difficult to determine the identity of signer from an actual signer and a designated verifier for a given SDVS.

Recently, Lin *et al.* [LWY10] proposed a DL based short strong designated verifier signature scheme. An SDVS scheme has two involved parties, a signer and a designated verifier. Each one is a probabilistic polynomial-time Turing machine (PPTM). The signer will generate an SDVS intended for the designated verifier. Consequently, the corresponding SDVS can only be validated by the designated verifier with his private key. An SDVS scheme is correct if a signer can generate a valid SDVS and only a designated verifier can be convinced of the signer's identity. Lin *et al.*'s SDVS scheme consists of the following algorithms:

– **Setup:** Taking as input 1^k where k is a security parameter, the algorithm generates system's public parameters *params*.

– **Signature-Generation (SG):** The SG algorithm takes as input system parameters *params*, a message, the public key of designated verifier and the private key of signer. It generates a corresponding SDVS δ.

– **Signature-Verification (SV):** The SV algorithm takes as input system parameters *params*, a message m, an SDVS δ, the private key of designated verifier and the public key of signer. It outputs **True** if δ is a valid SDVS for m. Otherwise, an error symbol $\perp$ is returned as a result.

– **Transcript-Simulation (TS):** The TS algorithm takes as input system parameters *params*, a message m, an SDVS δ and the private key of designated verifier. It outputs another valid SDVS δ^* for m.

The concrete construction of each algorithm is described as follows:

– **Setup:** Taking as input 1^k, the system authority (SA) selects two large primes p and q where $|q| = k$ and $q \mid (p - 1)$. Let g be a generator of order q and f: $Z_p^* \times Z_p^* \rightarrow Z_q$, F: $Z_q \rightarrow Z_q$ and H: $\{0, 1\}^* \times Z_q \rightarrow Z_q$ collision resistant hash functions. The system publishes public parameters $params = \{p, q, g, f, F, H\}$. Each user U_i chooses his private key $x_i \in Z_q$ and computes the public key as $y_i = g^{x_i} \bmod p$. In addition, he also announces a universal parameter $T_i = g^{c_i} \bmod p$ where $c_i \in_R Z_q$.

– **Signature-Generation (SG):** Let U_s and U_v separately be a signer and a designated verifier. For signing a message $m \in_R \{0, 1\}^*$, U_s first chooses $w \in_R Z_q$ to compute $Q = F(w)$ and

$$R = f(y_v^{\,w} \bmod p, y_v^{\,c_s} \bmod p), \tag{2.2.1}$$

$$S = (w - x_s H(m, Q, T_s)) \bmod q. \tag{2.2.2}$$

Then U_s delivers m along with its SDVS $\delta = (Q, R, S)$ to U_v.

– **Signature-Verification (SV):** Upon receiving (m, δ), U_v computes

$$Z_1 = y_v^{\,S} y_s^{\,x_v H(m, Q, T_s)} \bmod p, \tag{2.2.3}$$

$$Z_2 = T_s^{\,x_v} \bmod p, \tag{2.2.4}$$

and then verifies the signature by checking whether

$$R = f(Z_1, Z_2). \qquad (2.2.5)$$

We show that the verification of Eq. (2.2.5) works correctly. From the right-hand side of Eq. (2.2.5), we have

$$f(Z_1, Z_2)$$

$$= f(y_v^{S} y_s^{x_v H(m,\, Q,\, T_s)} \bmod p,\; T_s^{x_v} \bmod p) \qquad \text{(by Eqs. (2.2.3) and (2.2.4))}$$

$$= f(y_v^{S + x_s(H(m,\, Q,\, T_s)} \bmod p,\; T_s^{x_v} \bmod p)$$

$$= f(y_v^{S + x_s(H(m,\, Q,\, T_s)} \bmod p,\; y_v^{c_s} \bmod p)$$

$$= f(y_v^{w} \bmod p,\; y_v^{c_s} \bmod p) \qquad \text{(by Eq. (2.2.2))}$$

$$= R \qquad \text{(by Eq. (2.2.1))}$$

which leads to the left-hand side of Eq. (2.2.5).

– **Transcript-Simulation (TS):** To generate another SDVS δ^* intended for himself, U_v computes

$$S^* = S + 1 \bmod q, \qquad (2.2.6)$$

$$R^* = f(y_v Z_1 \bmod p,\, Z_2). \qquad (2.2.7)$$

Here, $\delta^* = (Q, R^*, S^*)$ is another valid SDVS for the message m. In fact, the probability that the computed $\delta^* = (Q, R^*, S^*)$ and the received $\delta = (Q, R, S)$ are identical is at most $1/2^k$, i.e., $\Pr[\delta^* = \delta] \leq 1/2^k$.

Motivated by Schnorr's signature scheme [Sch91], Lin *et al.*'s scheme can be regarded as a generic signature scheme. Therefore, we can directly apply the Forking lemma introduced by Pointcheval and Stern [PS00] to prove the security of their scheme. Concretely speaking, we can first obtain two equations

$$Z_1 = y_v^{S} y_s^{x_v H(m,\, Q,\, T_s)} \bmod p,$$

$$Z_1 = y_v^{S'} y_s^{x_v H'(m,\, Q,\, T_s)} \bmod p,$$

and then compute the private key x_s as $(S - S')/(H'(m, Q, T_s) - H(m, Q, T_s))$. Theorem 2.2.1 gives more detailed security proof and advantage analyses to show the tight relation between the security of their SDVS scheme and the hardness of the DLP.

Theorem 2.2.1. *Lin et al.'s SDVS scheme is* $(t, q_F, q_H, q_{SG}, q_{SV}, \varepsilon)$*-secure against existential forgery on adaptive chosen-message attacks (EU-CMA) in the random oracle model if there is no probabilistic polynomial-time adversary that can* (t', ε')*-break the DLP, where*

$$\varepsilon' \geq (q_F{}^{-1})(\varepsilon - 2^{-k}) + ((q_F - 1)q_F{}^{-1})(4^{-1}(\varepsilon - 2^{-k})^3(q_F{}^{-1} + q_H{}^{-1})),$$

$$t' \approx t + t_\lambda(2q_{SG} + 2q_{SV}).$$

Here t_λ *is the costs for performing a modular exponentiation over a finite field.*

Proof: Please refer to [LWY10] for the full version.

Table 2.2.1. Comparisons of previous SDVS schemes

Scheme **Item**	**JSI**	**SKM**	**YL**	**LWY**	**LC-1**	**LC-2**
Unforgeability	×	O	O	O	O	O
Non-Transferability	O	O	O	O	O	O
Signer's Ambiguity	×	×	×	O	×	O
Provable Security	×	×	O	O	×	×
Signature Length	$3\|p\| + 3\|q\|$	$3\|q\|$	$\|p\| + \|q\|$	$3\|q\|$	$2\|p\| + 2\|q\|$	$\|p\| + \|q\|$
#Exponentiation for entire scheme	16	6	3	5	12	7

Table 2.2.1 summarizes the comparison of previous SDVS schemes including Jakobsson *et al.*'s (JSI for short) [JSI96], Saeednia *et al.*'s (SKM for short) [SKM03], the Yang-Liao (YL for short) [YL10], Lin *et al.*'s (LWY for short) [LWY10] and two presented by Lee and Chang separately in 2007 (LC-1 for short) [LC07] and 2009 (LC-2 for short) [LC09]. Although the Yang-Liao scheme has the lowest computation costs, the signature length of their scheme is longer than that of Lin *et al.*'s. Most importantly, their scheme cannot satisfy the requirement of signer's ambiguity addressed in [LC09], which is regarded as an essential

property of secure SDVS schemes. To sum up, Lin *et al.*'s SDVS scheme not only provides better functionalities, but also has lower computation costs and shorter signature length.

2.3 Convertible Authenticated Encryption Scheme

Considering the RSA cryptosystem, in 2009, Wu and Lin [WL09] presented a CAE scheme based on RSA assumption. A CAE scheme has two involved parties, a signer and a designated recipient. Each one is a polynomial-time-bounded probabilistic Turing machine (PPTM). A signer will generate an authenticated ciphertext and deliver it to a designated recipient. Yet, a dishonest signer might repudiate his generated ciphertext. Finally, the designated recipient decrypts the ciphertext and verifies the signature. The Wu-Lin scheme consists of the following algorithms:

– **Setup:** Taking as input 1^k where k is a security parameter, the algorithm generates system's public parameters *params*.

– **Authenticated-Ciphertext-Generation (ACG):** The ACG algorithm takes as input system parameters *params*, a message m, the public key of designated recipient and the private key of signer. It generates a corresponding authenticated ciphertext δ.

– **Signature-Recovery-and-Verification (SRV):** The SRV algorithm takes as input system parameters *params*, an authenticated ciphertext δ, the private key of designated recipient and the public key of signer. It outputs a message m and its converted signature Ω if the authenticated ciphertext δ is valid. Otherwise, an error symbol $\perp$ is returned as a result.

The concrete construction of each algorithm is described as follows:

– **Setup:** Initially, each user chooses two large primes (p_i, q_i), computes $N_i = p_i q_i$, selects e_i relatively prime to $\phi(N_i)$ and then derives d satisfying that $ed = 1 \bmod \phi(N)$. Here, (N_i, e_i) and (p_i, q_i, d_i) are public and private keys of each user, respectively. Let h: $\{0, 1\}^k \times \{0, 1\}^k \to \{0, 1\}^k$ be a collision resistant hash function, where $|k| = 160$ bits and $|k| < |N_i| \approx 2048$ bits.

– **Authenticated-Ciphertext-Generation (ACG):** For signing a message m, a signer U_s chooses an integers $c \in \{0, 1\}^k$ and computes

$$R = mc^{c} \bmod N_v, \tag{2.3.1}$$

$$T = c^{e_v} \bmod N_v, \tag{2.3.2}$$

$$S = h(m, c)^{d_s} \bmod N_s, \tag{2.3.3}$$

and then delivers the authenticated ciphertext $\delta = (S, R, T)$ to a designated recipient U_v.

– **Signature-Recovery-and-Verification (SRV):** Upon receiving δ, U_v first computes

$$c = T^{d_v} \bmod N_v. \tag{2.3.4}$$

He then recovers the message m as

$$m = Rc^{-c} \bmod N_v, \tag{2.3.5}$$

and checks the redundancy embedded in m. U_v can further verify the signature by checking if

$$S^{e_s} = h(m, c) \bmod N_s. \tag{2.3.6}$$

We show that U_v then can correctly recover the message m with embedded redundancy by Eq. (2.3.5). From the right-hand side of Eq. (2.3.5), we have

$$\begin{aligned} & Rc^{-c} \\ &= (mc^{c})c^{-c} && \text{(by Eq. (2.3.1))} \\ &= m \pmod{N_v} \end{aligned}$$

which leads to the left-hand side of Eq. (2.3.5).

If the authenticated ciphertext (S, R, T) is correctly generated, it will pass the test of Eq. (2.3.6). From the right-hand side of Eq. (2.3.6), we have

$$\begin{aligned} & h(m, c) \\ &= (h(m, c)^{d_s})^{e_s} \\ &= S^{e_s} \pmod{N_s} && \text{(by Eq. (2.3.3))} \end{aligned}$$

which leads to the left-hand side of Eq. (2.3.6).

Since the secret parameter c is obtained during the verification of authenticated ciphertext, the recipient can easily reveal the converted signature (S, c) along with the message m in case of a later repudiation. One can see that the conversion process is efficient as it will not incur extra computation costs or communication overheads. Anyone can perform Eq. (2.3.6) to verify the correctness of converted signature.

The IND-CCA2 and the EF-CMA security for their scheme can be proved in the random oracle model as Theorems 2.3.1 and 2.3.2, respectively.

Theorem 2.3.1. (Proof of Confidentiality) *The Wu-Lin scheme is* (t, q_h, q_{ACG}, q_{SRV}, ε)*-secure against indistinguishability under adaptive chosen-ciphertext attacks* (*IND-CCA2*) *in the random oracle model if there is no probabilistic polynomial-time adversary that can* (t', ε')*-break the RSA problem, where*

$$\varepsilon' \geq (q_h{}^{-1})(2\varepsilon - \frac{q_{SRV}}{2^k}),$$

$$t' \approx t + t_\lambda(q_h + q_{ACG} + q_{SRV}).$$

Here t_λ *is the average running time of one oracle-query.*

Proof: Please refer to [WL09] for the full version.

Theorem 2.3.2. (Proof of Unforgeability) *The Wu-Lin scheme is* (t, q_h, q_{ACG}, ε)*-secure against existential forgery under adaptive chosen-message attacks* (*EF-CMA*) *in the random oracle model if there is no probabilistic polynomial-time adversary that can* (t', ε')*-break the RSA problem, where*

$$\varepsilon' \geq (q_h{}^{-1})(\varepsilon - 2^{-k}),$$

$$t' \approx t + t_\lambda(q_h + q_{ACG}).$$

Here t_λ *is the average running time of one oracle-query.*

Proof: Please refer to [WL09] for the full version.

2.4 Proxy Signcryption Scheme

In 1997, Zheng [Zhe97] proposed a so-called signcryption scheme which is suitable for confidential applications. A signcryption scheme only allows a designated recipient to verify a signer's signature instead of everyone for the purpose of confidentiality. In 1998, Petersen and Michels [PM98] also proposed another signcryption variant modified from an authenticated encryption scheme. Yet, He and Wu [HW99] pointed out that their scheme is vulnerable to the forgery attack. To deal with a later dispute that a signer repudiates his signature, Zheng [Zhe97] introduced an arbitration mechanism by using the zero-knowledge protocol [BJY97, Cha90]. However, the arbitration mechanism is inefficient as it will increase extra computation efforts and communication overheads. In 1998, Bao and Deng [BD98] addressed an efficient way to handle a repudiation dispute. Their scheme enables a designated recipient to convert a signcrypted message into an ordinary signature for public verification without imposing extra burdens on computation or communication cost. In 2002, Baek *et al.* [BSZ02] introduced the formal security proof model for a signcryption scheme in the random oracle model. The next year, Boyen [Boy03] proposed a provably secure identity-based signcryption scheme with ciphertext anonymity. In 2005, Hwang *et al.* [HLS05] proposed an elliptic curve based signcryption scheme with forward secrecy for facilitating the gradually widely used mobile applications.

Considering proxy delegation, in 2010, Lin *et al.* [LWH10] proposed an efficient proxy signcryption scheme based on bilinear pairings. A proxy signcryption scheme mainly has three involved parties, an original signer, a proxy signer and a designated recipient. All parties are probabilistic polynomial-time Turing machines (PPTM). An original signer delegates his signing power to a proxy signer by issuing a proxy credential. After that, the latter can generate a signcrypted message on behalf of the former and sends it to a designated recipient. Finally, the designated recipient decrypts the message and verifies the proxy signature. A proxy signcryption scheme is correct if a proxy signer can generate a valid signcrypted message on behalf of an original signer and only a designated recipient is capable of decrypting it and verifying the proxy signature. Lin *et al.*'s scheme consists of the following algorithms:

– **Setup:** Taking as input 1^k where k is a security parameter, the algorithm generates system's public parameters *params*.

– **Proxy-Credential-Generation (PCG):** The PCG algorithm takes as input the private key of original signer and outputs a corresponding proxy credential for a proxy signer.

– **Signcrypted-Message-Generation (SMG):** The SMG algorithm takes as input a plaintext m, a proxy credential, the public key of designated recipient and the private key of proxy signer. It generates a corresponding signcrypted message δ.

– **Signature-Recovery-and-Verification (SRV):** The SRV algorithm takes as input a signcrypted message δ, the private key of designated recipient and the public keys of original and proxy signers. It outputs a plaintext m and its converted ordinary proxy signature Ω if the signcrypted message δ is valid. Otherwise, an error symbol ⊥ is returned.

The concrete construction of each algorithm is described as follows:

– **Setup:** Taking as input 1^k, the system authority (SA) selects two groups $(\boldsymbol{G}_1, +)$ and $(\boldsymbol{G}_2, \times)$ of the same prime order q with $|q| = k$. Let P be a generator of order q over $\boldsymbol{G}_1$, e: $\boldsymbol{G}_1 \times \boldsymbol{G}_1 \rightarrow \boldsymbol{G}_2$ a bilinear pairing and h_1: $\{0, 1\}^k \times \boldsymbol{G}_1 \rightarrow Z_q$, h_2: $\boldsymbol{G}_1 \rightarrow \boldsymbol{G}_1$ and h_3: $\boldsymbol{G}_2 \times \boldsymbol{G}_1 \rightarrow \{0, 1\}^k$ collision resistant hash functions. The system publishes $params = \{\boldsymbol{G}_1, \boldsymbol{G}_2, q, P, e, h_1, h_2, h_3\}$. Each user U_i chooses his private key $x_i \in_R Z_q$ and computes the corresponding public one as $Y_i = x_iP$.

– **Proxy-Credential-Generation (PCG):** Let U_o be an original signer delegating his signing power to a proxy signer U_p. U_o first chooses an integer $d \in Z_q$ to compute

$$N = dP, \tag{2.4.1}$$

$$\sigma = x_o + d(m_w) \bmod q, \tag{2.4.2}$$

where m_w is a warrant consisting of the identifiers of original signer, proxy signer and designated recipient, the delegation duration and so on. The proxy credential (σ, N, m_w) is then sent to U_p. Upon receiving (σ, m_w, N), U_p first checks its validity by verifying whether

$$\sigma P = Y_o + m_wN. \tag{2.4.3}$$

If it does not hold, (σ, m_w, N) is requested to be sent again.

We first show that the verification of Eq. (2.4.3) works correctly. From the left-hand side of Eq. (2.4.3), we have

$$
\begin{aligned}
&\sigma P \\
&= (x_o + d(m_w))P && \text{(by Eq. (2.4.2))} \\
&= x_o P + d(m_w)P \\
&= Y_o + m_w N && \text{(by Eq. (2.4.1))}
\end{aligned}
$$

which leads to the right-hand side of Eq. (2.4.3).

– **Signcrypted-Message-Generation (SMG):** For signcrypting a plaintext $m \in_R \{0, 1\}^k$ on behalf of the original signer U_o, U_p chooses $r \in_R Z_q$ to compute

$$R = rP, \tag{2.4.4}$$

$$S = r(h_1(m, R) + x_p + \sigma)^{-1}P, \tag{2.4.5}$$

$$V = e(h_2(\sigma Y_v), x_p Y_v), \tag{2.4.6}$$

$$X = E_V(S), \tag{2.4.7}$$

$$Y = h_3(V, R) \oplus m, \tag{2.4.8}$$

and then delivers the warrant m_w and the signcrypted message δ = (R, X, Y, N) to a designated recipient U_v, where E_V denotes a symmetric encryption function with key V.

– **Signature-Recovery-and-Verification (SRV):** Upon receiving (R, X, Y, N), U_v first computes

$$V = e(h_2(x_v(Y_o + m_w N)), x_v Y_p), \tag{2.4.9}$$

to recover the plaintext m as

$$m = h_3(V, R) \oplus Y \tag{2.4.10}$$

and checks the redundancy embedded in m. U_v further computes S as

$$S = D_V(X) \tag{2.4.11}$$

and verifies the proxy signature by checking if

$$e(h_1(m, R)P + Y_p + Y_o + m_wN, S) = e(P, R). \tag{2.4.12}$$

Note that D_V is a corresponding symmetric decryption function with key V.

We demonstrate that with received (R, X, Y, N) and the warrant m_w, a designated recipient can correctly recover the plaintext and verify the embedded proxy signature with Eq. (2.4.12). From the left-hand side of Eq. (2.4.12), we have

$$\begin{aligned}
&e(h_1(m, R)P + Y_p + Y_o + m_wN, S)\\
&= e(h_1(m, R)P + Y_p + Y_o + m_wN, r(h_1(m, R) + x_p + \sigma)^{-1}P) && \text{(by Eq. (2.4.5))}\\
&= e((h_1(m, R) + x_p + x_o + d(m_w))P, r(h_1(m, R) + x_p + x_o + d(m_w))^{-1}P)\\
& && \text{(by Eqs. (2.4.1) and (2.4.2))}\\
&= e(P, rP)\\
&= e(P, R) && \text{(by Eq. (2.4.4))}
\end{aligned}$$

which leads to the right-hand side of Eq. (2.4.12).

Since a converted proxy signature $\Omega = (S, R, N)$ is derived during the verification process, a designated recipient U_v can easily announce it together with (m, m_w) in case of a later dispute over repudiation. Accordingly, anyone can check Eq. (2.4.12) to realize proxy signer's dishonesty.

The IND-CCA2 and the EF-CMA security for their scheme can be proved in the random oracle model as Theorems 2.4.1 and 2.4.2, respectively.

Theorem 2.4.1. (Proof of Confidentiality) *Lin et al.'s scheme is* $(t, q_{h_1}, q_{h_2}, q_{h_3}, q_{PCG}, q_{SMG}, q_{SRV}, \varepsilon)$*-secure against indistinguishability under adaptive chosen-ciphertext attacks*

(*IND-CCA2*) *in the random oracle model if there is no probabilistic polynomial-time adversary that can* (t', ε')*-break the BDHP, where*

$$\varepsilon' \geq ({q_{h_3}}^{-1})(2\varepsilon - q_{SRV}(2^{-k})),$$

$$t' \approx t + t_{\lambda}(q_{SMG} + 2q_{SRV}).$$

Here t_{λ} *is the time for performing one bilinear pairing operation.*

Proof: Please refer to [LWH10] for the full version.

Theorem 2.4.2. (Proof of Unforgeability) *Lin et al.'s scheme is* (t, q_{h_1}, q_{h_2}, q_{h_3}, q_{PCG}, q_{SMG}, ε)*-secure against existential forgery under adaptive chosen-message attacks* (*EF-CMA*) *in the random oracle model if there is no probabilistic polynomial-time adversary that can* (t', ε')*-break the BDHP, where*

$$\varepsilon' \geq (\varepsilon - (q_{h_2} + 1)/2^{k})/(q_{h_2}q_{h_3}),$$

$$t' \approx t + t_{\lambda}(q_{SMG}).$$

Here t_{λ} *is the time for performing one bilinear pairing operation.*

Proof: Please refer to [LWH10] for the full version.

Table 2.4.1 summarizes the comparison of previous signcryption schemes including the Elkamchouchi-Abouelseoud [EA08] (EA for short), Duan *et al.*'s (DCZ for short) [DCZ05], the Li-Chen (LC for short) [LC04], the Wang-Cao (WC for short) [WC05], the Duan-Cao (DC for short) [DC06] and Lin *et al.*'s (LWH for short) [LWH10] schemes in terms of functionalities and security proofs. Note that the Elkamchouchi-Abouelseoud and Duan *et al.*'s schemes are vulnerable to the key exposure attack, i.e., once the private key of proxy signer is compromised, an attacker can easily recover the plaintext without the knowledge of designated recipient's private key. From this table, it can be seen that Lin *et al.*'s scheme not only provides better functionalities, but also has provable security.

Table 2.4.1. Comparisons of previous signcryption schemes

Scheme Item	EA	DCZ	LC	WC	DC	LWH
Pairing-based scheme	O	O	O	O	O	O
Against key exposure attack	×	×	O	O	O	O
Proxy delegation	O	O	O	O	×	O
Partial delegation with warrant	×	O	O	O	×	O
Public verifiability	×	O	O	O	O	O
No conversion cost	×	O	O	O	O	O
Complete proof of confidentiality	×	×	×	×	O	O
Complete proof of unforgeability	×	×	×	×	O	O

Table 2.4.2 further summarizes the comparison of computation costs in number of the most time-consuming operations, i.e., bilinear pairing computation. To obtain fair comparison results, the Duan-Cao scheme is excluded in Table 2.4.2, since their scheme does not have the property of proxy delegation. From the comparison results shown in Table 2.4.2, one can see that Lin *et al*.'s scheme outperforms compared ones and hence is more suitable for practical implementation.

Table 2.4.2. Comparisons of computation costs for previous proxy signcryption schemes

scheme Item	EA	DCZ	LC	WC	LWH
#Bilinear pairing for PCG	2	3	3	2	0
#Bilinear pairing for SMG	2	2	2	1	1
#Bilinear pairing for SRV	4	4	8	3	3
Total costs for the entire scheme	8	9	13	6	4

3. Formal Model of the PCAE Scheme

In this section, we first state involved parties of a PCAE scheme and then address its algorithms and security model.

3.1 Involved Parties

A proxy CAE scheme has three involved parties, an original signer, an authorized proxy signer and a designated recipient. Each one is a probabilistic polynomial-time Turing machine (PPTM). An original signer will compute and transmit a proxy credential to a proxy signer. The latter is responsible for producing an authenticated ciphertext on behalf of the former while a dishonest proxy signer might repudiate having generated his ciphertext. Finally, a designated recipient decrypts the ciphertext and verifies the proxy signature. A proxy CAE scheme is correct if a proxy signer can generate a valid authenticated ciphertext and only a designated recipient is capable of decrypting it and verifying the proxy signature.

3.2 Algorithms

The proposed proxy CAE (PCAE) scheme consists of following algorithms:

– **Setup:** Taking as input 1^k where k is a security parameter, the algorithm generates system's public parameters *params*.

– **Proxy-Credential-Generation (PCG):** The PCG algorithm takes as input system parameters *params*, a warrant and the private key of original signer. It outputs a corresponding proxy credential.

– **Authenticated-Ciphertext-Generation (ACG):** The ACG algorithm takes as input system parameters *params*, a proxy credential, a message m, the public key of designated recipient and the private key of proxy signer. It generates a corresponding authenticated ciphertext δ.

– **Signature-Recovery-and-Verification (SRV):** The SRV algorithm takes as input system parameters *params*, an authenticated ciphertext δ, the private key of designated recipient and the public keys of original and proxy signers. It outputs a message m and its converted proxy signature Ω if the authenticated ciphertext δ is valid. Otherwise, an error symbol $\perp$ is returned as a result.

3.3 Security Models

Two crucial security requirements of proposed proxy CAE schemes are message confidentiality and unforgeability. The widely accepted notion for the security of message confidentiality comes from the definition of indistinguishability-based security, i.e., an adversary attempts to distinguish a target ciphertext with respect to two candidate plaintexts. In the taxonomy of cryptanalysis, there are three kinds of attacks: ciphertext-only attack, chosen-ciphertext attack (CCA) and adaptive chosen-ciphertext attack (CCA2). An adversary in ciphertext-only attack cannot make any query while that in CCA can query the plaintext for his chosen ciphertext once. An adversary in CCA2 is the most advantageous as he can adaptively make new queries based on previous results. We therefore consider an adversary in CCA2 against our proposed schemes in the security requirement of message confidentiality. In addition to SRV queries, we also give an adversary the ability to make PCG and ACG queries. When it comes to the security requirement of unforgeability, we usually refer to an adversary in adaptive chosen-message attack (CMA). Such an adversary attempts to forge a valid authenticated ciphertext for his chosen message and is permitted to adaptively make PCG and ACG queries in our defined security notion. We describe several game models for the above two crucial security requirements as Definitions 3.3.1 to 3.3.4, respectively.

Then we can formally prove the security of our schemes in the random oracle model. Namely, one-way hash functions are simulated as random oracles controlled by a challenger who is responsible for answering an adversary's queries in the defined game model. Note that simulated results of each random query should be computationally indistinguishable from those generated by a real scheme. Basically, the concept of security proof is a security reduction. That is to say, we can reduce a well-known cryptographic problem such as CDHP to our proposed schemes meaning that if there is any adversary winning the game in CCA2 or CMA, a challenger that takes the adversary's advantages is able to break CDHP. We define these notions as follows:

Definition 3.3.1. (IND-onetime secure) *A proxy CAE scheme is said to achieve the security requirement of confidentiality against indistinguishability* (*IND-onetime*) *if there is no* *probabilistic polynomial-time adversary $\mathcal{A}$ with a non-negligible advantage in the following game played with a challenger $\mathcal{B}$:*

Setup: The challenger $\mathcal{B}$ first runs the Setup(1^k) algorithm and sends system's public parameters *params* to the adversary $\mathcal{A}$.

Phase 1: $\mathcal{A}$ can only ask random oracles adaptively, i.e., each query might be based on the result of previous queries.

Challenge: $\mathcal{A}$ produces two messages, m_0 and m_1, of the same length. The challenger $\mathcal{B}$ flips a coin $\lambda \leftarrow \{0, 1\}$ and generates an authenticated ciphertext δ^* for m_λ. The ciphertext δ^* is then delivered to $\mathcal{A}$ as a target challenge.

Guess: At the end of the game, $\mathcal{A}$ outputs a bit λ'. The adversary $\mathcal{A}$ wins this game if $\lambda' = \lambda$. We define $\mathcal{A}$*'s* advantage as $Adv(\mathcal{A}) = |\ \Pr[\lambda' = \lambda] - 1/2\ |$.

Definition 3.3.2. (IND-CPA2 secure) *A proxy CAE scheme is said to achieve the security requirement of confidentiality against indistinguishability under adaptive chosen-plaintext attacks (IND-CPA2) if there is no probabilistic polynomial-time adversary $\mathcal{A}$ with a non-negligible advantage in the following game played with a challenger $\mathcal{B}$:*

Setup: The challenger $\mathcal{B}$ first runs the Setup(1^k) algorithm and sends system's public parameters *params* to the adversary $\mathcal{A}$.

Phase 1: $\mathcal{A}$ can issue several queries adaptively, i.e., each query might be based on the result of previous queries:

– *Proxy-Credential-Generation (PCG) queries:* $\mathcal{A}$ issues a PCG query with respect to an original and a proxy signers. $\mathcal{B}$ returns a corresponding proxy credential.

– *Authenticated-Ciphertext-Generation (ACG) queries:* $\mathcal{A}$ chooses a message m and then gives $\mathcal{B}$ a proxy credential along with a message m. $\mathcal{B}$ returns a corresponding authenticated ciphertext δ to $\mathcal{A}$.

Challenge: $\mathcal{A}$ produces two messages, m_0 and m_1, of the same length. The challenger $\mathcal{B}$ flips a coin $\lambda \leftarrow \{0, 1\}$ and generates an authenticated ciphertext δ^* for m_λ. The ciphertext δ^* is then delivered to $\mathcal{A}$ as a target challenge.

Phase 2: $\mathcal{A}$ can issue new queries as those in Phase 1.

Guess: At the end of the game, $\mathcal{A}$ outputs a bit λ'. The adversary $\mathcal{A}$ wins this game if $\lambda' = \lambda$. We define $\mathcal{A}$'*s* advantage as $Adv(\mathcal{A}) = |\ \Pr[\lambda' = \lambda] - 1/2\ |$.

Definition 3.3.3. (IND-CCA2 secure) *A proxy CAE scheme is said to be semantically secure against adaptive chosen ciphertext attacks (IND-CCA2) if there is no probabilistic polynomial-time adversary $\mathcal{A}$ with a non-negligible advantage in the following game played with a challenger $\mathcal{B}$:*

Setup: The challenger $\mathcal{B}$ first runs the Setup(1^k) algorithm and sends system's public parameters *params* to the adversary $\mathcal{A}$.

Phase 1: $\mathcal{A}$ can issue several queries adaptively, i.e., each query might be based on the result of previous queries:

– *Proxy-Credential-Generation (PCG) queries:* $\mathcal{A}$ issues a PCG query with respect to an original and a proxy signers. $\mathcal{B}$ returns a corresponding proxy credential.

– *Authenticated-Ciphertext-Generation (ACG) queries:* $\mathcal{A}$ chooses a message m and then gives $\mathcal{B}$ a proxy credential along with a message m. $\mathcal{B}$ returns a corresponding authenticated ciphertext δ to $\mathcal{A}$.

– *Signature-Recovery-and-Verification (SRV) queries:* $\mathcal{A}$ submits an authenticated ciphertext δ along with a warrant m_w to $\mathcal{B}$. If δ is valid, $\mathcal{B}$ returns a recovered message m and its converted proxy signature Ω. Otherwise, an error symbol $\perp$ is outputted as a result.

Challenge: The adversary $\mathcal{A}$ produces two messages, m_0 and m_1, of the same length. The challenger $\mathcal{B}$ flips a coin $\lambda \leftarrow \{0, 1\}$ and generates an authenticated ciphertext δ^* for m_λ. The ciphertext δ^* is then delivered to $\mathcal{A}$ as a target challenge.

Phase 2: The adversary $\mathcal{A}$ can issue new queries as those in Phase 1 except an SRV query for the target ciphertext.

Guess: At the end of the game, $\mathcal{A}$ outputs a bit λ'. The adversary $\mathcal{A}$ wins this game if $\lambda' = \lambda$. We define $\mathcal{A}$*'s* advantage as $Adv(\mathcal{A}) = |\ \Pr[\lambda' = \lambda] - 1/2\ |$.

Definition 3.3.4. (EF-CMA secure) *A proxy CAE scheme is said to achieve the security requirement of unforgeability against existential forgery under adaptive chosen-message attacks* (*EF-CMA*) *if there is no probabilistic polynomial-time adversary* $\mathcal{A}$ *with a non-negligible advantage in the following game played with a challenger* $\mathcal{B}$:

Setup: $\mathcal{B}$ first runs the Setup(1^k) algorithm and sends system's public parameters *params* to the adversary $\mathcal{A}$.

Phase 1: $\mathcal{A}$ adaptively issues PCG and ACG queries as those in Phase 1 of Definition 3.3.3.

Forgery: Finally, $\mathcal{A}$ arbitrarily chooses a message m and produces an authenticated ciphertext δ^* which is not outputted by ACG query. The adversary $\mathcal{A}$ wins if δ^* is valid.

4. PCAE-(I) Scheme

In this section, we demonstrate the proposed first proxy CAE (abbreviated to PCAE-(I)) scheme.

4.1 Construction

– **Setup:** Initially, each user chooses two large primes (p_i, q_i), computes $N_i = p_i q_i$, selects e_i relatively prime to $\phi(N_i)$ and then derives d_i satisfying that $e_i d_i = 1 \bmod \phi(N_i)$. Here, ($N_i$, e_i) and (p_i, q_i, d_i) are public and private keys of each user, respectively. Let h_1: $\{0, 1\}^k \rightarrow \{0, 1\}^k$ and h_2: $\{0, 1\}^k \times Z_{N_o} \times \{0, 1\}^k \rightarrow \{0, 1\}^k$ be collision resistant hash functions, where $|k| = 160$ bits and $|k| < |N_i| \approx 2048$ bits.

– **Proxy-Credential-Generation (PCG):** Let U_o be an original signer delegating his signing power to a proxy signer U_p. U_o distributes a proxy credential σ and m_w to U_p with the following steps:

Step 1 U_o first uses his private key d_o to compute

$$\sigma = h_1(m_w)^{d_o} \bmod N_o, \tag{4.1.1}$$

where m_w is a warrant consisting of identifiers of original signer, proxy signer and designated recipient, the delegation duration and so on. Note that σ is regarded as the signature of m_w.

Step 2 Then U_o sends the proxy credential σ and m_w to U_p via a secure channel.

Step 3 Upon receiving it, U_p acquires U_o's public key (N_o, e_o) to verify whether

$$\sigma^{e_o} = h_1(m_w) \bmod N_o. \tag{4.1.2}$$

If it holds, U_p is convinced of the validity of received proxy credential; else, (σ, m_w) is requested to be sent again.

– **Authenticated-Ciphertext-Generation (ACG):** For signing a message m on behalf of the original signer U_o, U_p first chooses two integers r, $c \in \{0, 1\}^k$ and then computes

$$Q = r^{e_o} \bmod N_o, \tag{4.1.3}$$

$$R = \sigma r \bmod N_o, \tag{4.1.4}$$

$$D = mQ^c \bmod N_o, \tag{4.1.5}$$

$$T = c^{e_v} \bmod N_v, \tag{4.1.6}$$

$$S = h_2(m, Q, c)^{d_p} \bmod N_p. \tag{4.1.7}$$

Here, the proxy authenticated ciphertext $\delta = (S, R, T, D)$ and m_w are delivered to a designated recipient U_v.

– **Signature-Recovery-and-Verification (SRV):** Upon receiving δ and m_w, U_v first computes

$$c = T^{d_v} \bmod N_v, \tag{4.1.8}$$

$$Q = R^{e_o} h_1(m_w)^{-1} \bmod N_o. \tag{4.1.9}$$

He then recovers the message m as

$$m = DQ^{-c} \bmod N_o, \tag{4.1.10}$$

and checks the redundancy embedded in m. U_v can further verify the proxy signature by checking if

$$S^{e_p} = h_2(m, Q, c) \bmod N_p. \tag{4.1.11}$$

In case of the proxy signer’s repudiation, the designated recipient can solely release the converted proxy signature $\Omega = (S, R, c)$ along with (m, m_w) without extra computation efforts or communication overheads. To verify the correctness of the converted proxy signature, anyone can first compute Q with Eq. (4.1.9) and then verify the proxy signature by checking Eq. (4.1.11).

4.2 Correctness

We first show that in the proposed PCAE-(I) scheme, the proxy signer U_p can correctly verify the proxy credential with Eq. (4.1.2). From the left-hand side of Eq. (4.1.2), we have

$$
\begin{aligned}
&\sigma^{e_o} \\
&= h_1(m_w)^{d_o e_o} && \text{(by Eq. (4.1.1))} \\
&= h_1(m_w)^{(e_o d_o - 1) + 1} \\
&= h_1(m_w) h_1(m_w)^{(e_o d_o - 1)} \\
&= h_1(m_w) h_1(m_w)^{t\phi(N_o)} \\
&= h_1(m_w) (h_1(m_w)^{\phi(N_o)})^t \\
&= h_1(m_w)(1)^t && \text{(by Euler's Theorem [Sta06])} \\
&= h_1(m_w) \pmod{N_o}
\end{aligned}
$$

which leads to the right-hand side of Eq. (4.1.2).

With Eq. (4.1.10), the designated recipient U_v can correctly recover the message M with embedded redundancy. We first show that U_v can correctly obtain the shared secret Q. From the right-hand side of Eq. (4.1.9), we have

$$
\begin{aligned}
&R^{e_o} h_1(m_w)^{-1} \\
&= (\sigma r)^{e_o} h_1(m_w)^{-1} && \text{(by Eq. (4.1.4))} \\
&= (\sigma r)^{e_o} \sigma^{-e_o} && \text{(by Eq. (4.1.2))} \\
&= r^{e_o} \\
&= Q \pmod{N_o} && \text{(by Eq. (4.1.3))}
\end{aligned}
$$

which leads to the left-hand side of Eq. (4.1.9).

With the shared secret Q, U_v then can correctly recover the message m. From the right-hand side of Eq. (4.1.10), we have

$$DQ^{-c}$$

$$= mQ^{c}Q^{-c} \qquad \text{(by Eq. (4.1.5))}$$

$$= m \pmod{N_o}$$

which leads to the left-hand side of Eq. (4.1.10).

If an authenticated ciphertext (S, R, T, D) is correctly generated, it will pass the test of Eq. (4.1.11). From the right-hand side of Eq. (4.1.11), we have

$$h_2(m, Q, c)$$

$$= (h_2(m, Q, c)^{d_p})^{e_p}$$

$$= S^{e_p} \pmod{N_p} \qquad \text{(by Eq. (4.1.7))}$$

which leads to the left-hand side of Eq. (4.1.11).

4.3 Security Proofs

We prove that the proposed PCAE-(I) scheme achieves the IND-CCA2 and the EF-CMA security in the random oracle model as Theorems 4.3.1 and 4.3.2, respectively.

Theorem 4.3.1. (Proof of Confidentiality) *The proposed PCAE-(I) scheme is (t, q_{h_1}, q_{h_2}, q_{PCG}, q_{ACG}, q_{SRV}, ε)-secure against indistinguishability under adaptive chosen-ciphertext attacks (IND-CCA2) in the random oracle model if there is no probabilistic polynomial-time adversary that can (t', ε')-break the RSA problem, where*

$$\varepsilon' \geq (q_{h_2}{}^{-1})(2\varepsilon - \frac{q_{SRV}}{2^k}),$$

$$t' \approx t + t_\lambda(q_{h_1} + q_{h_2} + q_{PCG} + q_{PACG} + q_{PSRV}).$$

Here t_λ is the average running time of one oracle-query.

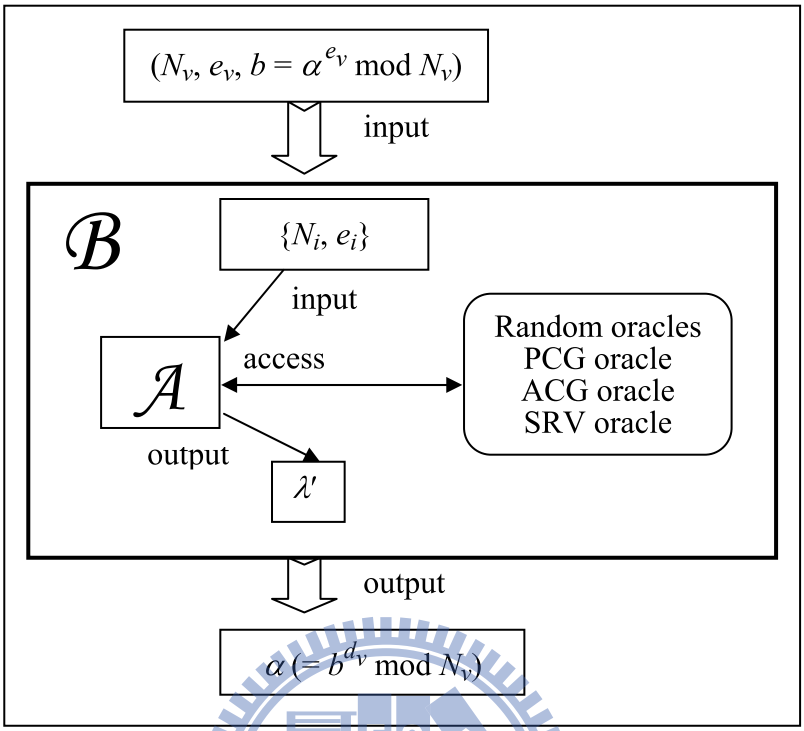


Fig. 4.3.1. The proof structure of confidentiality in Theorem 4.3.1

Proof: Fig. 4.3.1 depicts the proof structure of this theorem. Suppose that a $(t, q_{h_1}, q_{h_2}, q_{PCG}, q_{PACG}, q_{PSRV}, \varepsilon)$-PPTM $\mathcal{A}$ can break the proposed proxy CAE scheme with a non-negligible advantage ε under the adaptive chosen-ciphertext attack after running in time at most t and asking at most q_{h_i} h_i random oracle (for i = 1 and 2), q_{PCG} PCG, q_{PACG} PACG and q_{PSRV} PSRV queries. Then we can take $\mathcal{A}$ as a subroutine to construct another (t', ε')-algorithm $\mathcal{B}$ that solves the RSA problem with respect to the designated recipient's key pair in time at most t' and with the probability ε'. The algorithm $\mathcal{B}$ is said to (t', ε')-break the RSA problem. Let all involved parties and parameters be defined the same as those in Section 4.1. The objective of $\mathcal{B}$ is to obtain α (= b^{d_v} mod N_v) by taking (N_v, e_v, $b = \alpha^{e_v}$ mod N_v) as inputs. In this proof, $\mathcal{B}$ simulates a challenger to $\mathcal{A}$ in the following game.

Setup: The challenger $\mathcal{B}$ runs the Setup(1^k) algorithm and sends system's public parameters *params* = $\{N_i, e_i\}$ to the adversary $\mathcal{A}$.

Phase 1: $\mathcal{A}$ issues the following queries adaptively:

– h_1 *oracle:* When $\mathcal{A}$ asks an h_1 oracle of $h_1(m_w)$, $\mathcal{B}$ returns **$\mathcal{O}$-Sim(I)_*h*₁**(m_w). The simulated random oracle **$\mathcal{O}$-Sim(I)_*h*₁** operates as Fig. 4.3.2. Note that the function **insert**(N, b) will insert the value b into the array N.

```
oracle 𝒪-Sim(I)_h1(m_w)    // Let Q_h1[$q_{h_1}$] and A_h1[$q_{h_1}$][2] be two arrays.
for $i = 0$ to $q_{h_1} - 1$
    if (Q_h1[$i$] = $s$) then    // It is an old query.
        exit for;
    else if (Q_h1[$i$] = null) then    // It is a new query.
        insert(Q_h1, $s$); insert(A_h1, ($\sigma \in_R Z_{N_o}$, $v_1 = \sigma^{e_o} \bmod N_o$)); exit for;
    end if
next $i$
return A_h1[$i$][1];
```

Fig. 4.3.2. Algorithm of the simulated random oracle **$\mathcal{O}$-Sim(I)_*h*₁**

– h_2 *oracle:* When $\mathcal{A}$ asks an h_2 oracle of $h_2(m, Q, c)$, $\mathcal{B}$ returns **$\mathcal{O}$-Sim(I)_*h*₂**(m, Q, c). The simulated random oracle **$\mathcal{O}$-Sim(I)_*h*₂** operates as Fig. 4.3.3.

```
oracle 𝒪-Sim(I)_h2($m$, $Q$, $c$)    //Let Q_h2[$q_{h_1}$] and A_h2[$q_{h_1}$][2] be two arrays.
for $i = 0$ to $q_{h_2} - 1$
    if (Q_h2[$i$][0] = $m$) and (Q_h2[$i$][1] = $Q$) and (Q_h2[$i$][2] = $c$) then
        exit for;    // It is an old query.
    else if (Q_h2[$i$][0] = null) then    // It is a new query.
        insert(Q_h2, ($m$, $Q$, $c$)); insert(A_h2, ($S \in_R \{0, 1\}^k$, $v_2 = S^{e_p} \bmod N_p$));
        exit for;
    end if
next $i$
return A_h2[$i$][1];
```

Fig. 4.3.3. Algorithm of the simulated random oracle **$\mathcal{O}$-Sim(I)_*h*₂**

– *PCG queries:* When $\mathcal{A}$ makes a PCG query, $\mathcal{B}$ chooses a proper m_w and then returns (m_w, **$\mathcal{O}$-Sim(I)_*PCG***(m_w)). The simulated PCG oracle **$\mathcal{O}$-Sim(I)_*PCG*** operates as Fig. 4.3.4.

```
oracle $\mathcal{O}$-Sim(I)_PCG($m_w$)
$v_1 \leftarrow \mathcal{O}$-Sim(I)_$h_1$($m_w$)
for $i = 0$ to $q_{h_1} - 1$
    if (A_$h_1[i][1] = v_1$) then
        return A_$h_1[i][0]$;
    end if
next $i$
```

Fig. 4.3.4. Algorithm of the simulated PCG oracle **$\mathcal{O}$-Sim(I)_*PCG***

– *ACG queries:* When $\mathcal{A}$ makes an ACG query for some message m, $\mathcal{B}$ returns **$\mathcal{O}$-Sim(I)_*ACG***(m) as the result. The simulated ACG oracle **$\mathcal{O}$-Sim(I)_*ACG*** operates as Fig. 4.3.5.

```
oracle $\mathcal{O}$-Sim(I)_ACG($m$)
Choose $r, c \in \{0, 1\}^k$ and a proper $m_w$;
$(\sigma, m_w) \leftarrow \mathcal{O}$-Sim(I)_PCG($m_w$);
Compute $Q = r^{e_o} \bmod N_o$;
        $R = \sigma r \bmod N_o$;
        $D = mQ^c \bmod N_o$;
        $T = c^{e_v} \bmod N_v$;
$v_2 = \mathcal{O}$-Sim(I)_$h_2$($m, Q, c$);
for $i = 0$ to $q_{h_2} - 1$
    if (A_$h_2[i][1] = v_2$) then
        $S =$ A_$h_2[i][0]$;
    end if
next $i$
return $\delta = (S, R, T, D)$ along with $m_w$;
```

Fig. 4.3.5. Algorithm of the simulated ACG oracle **$\mathcal{O}$-Sim(I)_*ACG***

– *SRV queries:* When $\mathcal{A}$ makes an SRV query for a proxy authenticated ciphertext $\delta = (S, R, T, D)$ along with m_w, $\mathcal{B}$ returns **$\mathcal{O}$-Sim(I)_*SRV***(δ, m_w) as the result. The simulated SRV oracle **$\mathcal{O}$-Sim(II)_*SRV*** operates as Fig. 4.3.6. Note that the function **check**(N, b) will return a Boolean value depending on whether the value b is stored in the array N.

```
oracle $\mathcal{O}$-Sim(II)_SRV($\delta$, $m_w$)    //$\delta = (S, R, T, D)$
$Q = R^{e_o}(\mathcal{O}\text{-Sim(II)\_}h_1(m_w)^{-1}) \bmod N_o$;
if (check(A_$h_2$, $S$) = true) then
    for $j = 0$ to $q_{h_2} - 1$
        if (A_$h_2[j][0] = S$) then
            $M$ = Q_$h_2[j][0]$; $Q$ = Q_$h_2[j][1]$; $c$ = Q_$h_2[j][2]$; exit for;
        end if
    next $j$
    if ($m = DQ^{-c} \bmod N_o$) then
        return $\{m, \Omega = (S, R, c)\}$;
    else
        return $\perp$;
    end if
else
    return $\perp$;
end if
```

Fig. 4.3.6. Algorithm of the simulated SRV oracle $\mathcal{O}$-**Sim(I)_*SRV***

Challenge: $\mathcal{A}$ generates two messages, m_0 and m_1, of the same length. The challenger $\mathcal{B}$ flips a coin $\lambda \leftarrow \{0, 1\}$ and generates a proxy authenticated ciphertext $\delta^* = (S^*, R^*, T^*, D^*)$ for m_λ by running the simulated **Sim(I)_Challenge**(m_λ). The algorithm of **Sim(I)_Challenge** operates as Fig. 4.3.7.

```
algorithm Sim(I)_Challenge($m_\lambda$)
$(\sigma, m_w) \leftarrow \mathcal{O}$-Sim(I)_PCG($m_w$);    //$m_w$ is a properly chosen warrant.
Choose $r \in_R \{0, 1\}^k$ and compute $Q^* = r^{e_o} \bmod N_o$ and $R^* = \sigma r \bmod N_o$;
Choose $D^* \in_R Z_{N_o}$;
   // i.e., implicitly define $D^* = m_\lambda Q^{*\alpha}$;
Set $T^* = b$; // implicitly define $c = \alpha$ in Eq. (4.1.6)
Choose $S^* \in_R \{0, 1\}^k$ and compute $v_2 = S^{*e_p} \bmod N_p$;
insert(Q_$h_2$, ($m_\lambda$, $Q^*$, $\nabla$)) where $\nabla$ denotes the null symbol;
insert(A_$h_2$, ($S^*$, $v_2$));
   // i.e., implicitly define $h_2(m_\lambda, Q^*, \alpha) = v_2$. Note that the third parameter in Q_$h_2$ array
      is normally $c$ (= $\alpha$), but $\mathcal{B}$ does not know it.
return $\{m_w^*, \delta^* = (S^*, R^*, T^*, D^*)\}$;
```

Fig. 4.3.7. Algorithm of the simulated **Sim(I)_Challenge**

Phase 2: $\mathcal{A}$ issues new queries as those stated in Phase 1 except an SRV query for the target challenge δ^*.

Guess: Finally, $\mathcal{A}$ outputs a bit λ' as the result. If $\lambda' = \lambda$, $\mathcal{A}$ wins this game. We define $\mathcal{A}$'s advantage as $Adv(\mathcal{A}) = |\ \Pr[\lambda' = \lambda] - 1/2\ |$.

Analysis of the game: For each PCG or ACG query, $\mathcal{B}$ always returns a valid and computationally indistinguishable proxy credential or proxy authenticated ciphertext without unexpected terminations. Hence, the simulations of PCG and ACG queries could be regarded as perfect. Considering the simulation of SRV queries, $\mathcal{B}$ might return an error symbol $\perp$ for a valid proxy authenticated ciphertext δ if $\mathcal{A}$ has the ability to guess a correct random value without asking corresponding $h_2(m, Q, c)$ random oracle in advance. Let SRV_ERR, AC-V and QH$_2$ separately denote the events that $\mathcal{B}$ returns $\perp$ for some valid δ during the entire game, a ciphertext δ submitted by $\mathcal{A}$ is valid, and $\mathcal{A}$ has ever asked corresponding $h_2(m, Q, c)$ oracle beforehand. Then we can express the probability that $\mathcal{B}$ returns an error symbol for a valid ciphertext as $\Pr[\text{AC-V} \mid \neg\text{QH}_2] \leq 2^{-k}$. Because $\mathcal{A}$ is allowed to make at most q_{SRV} SRV queries, we can represent the probability of SRV_ERR as

$$\Pr[\text{SRV\_ERR}] \leq \frac{q_{SRV}}{2^k}. \qquad (4.3.1)$$

In the challenge phase, $\mathcal{B}$ has returned a simulated proxy authenticated ciphertext $\delta^* = (S^*, R^*, T^*, D^*)$ where $T^* = b$, which implicitly implies

$$c = T^{*\,d_v} = b^{d_v} = \alpha \bmod N_v.$$

When $\mathcal{A}$ happens to ask $h_2(m_\lambda, Q^*, \alpha)$ oracle in Phase 2, denoted by QH$_2$*, the simulation aborts as $\mathcal{B}$ does not know α. On the contrary, if the entire simulation game does not abort, denoted by ($\neg$Ab), $\mathcal{A}$ gains no advantage in guessing λ due to the randomness of the output of the random oracle, i.e.,

$$\Pr[\lambda' = \lambda \mid \neg\text{Ab}] = 1/2. \qquad (4.3.2)$$

Derived from the expression of $\Pr[\lambda' = \lambda]$, we have

$$\begin{aligned}
\Pr[\lambda' = \lambda] &= \Pr[\lambda' = \lambda \mid \neg\mathrm{Ab}]\Pr[\neg\mathrm{Ab}] + \Pr[\lambda' = \lambda \mid \mathrm{Ab}]\Pr[\mathrm{Ab}] \\
&\leq (1/2)\Pr[\neg\mathrm{Ab}] + \Pr[\mathrm{Ab}] && \text{(by Eq. (4.3.2))} \\
&= (1/2)(1 - \Pr[\mathrm{Ab}]) + \Pr[\mathrm{Ab}] \\
&= (1/2) + (1/2)\Pr[\mathrm{Ab}]. && (4.3.3)
\end{aligned}$$

Besides, we can also derive that

$$\begin{aligned}
\Pr[\lambda' = \lambda] &\geq \Pr[\lambda' = \lambda \mid \neg\mathrm{Ab}]\Pr[\neg\mathrm{Ab}] \\
&= (1/2)(1 - \Pr[\mathrm{Ab}]) \\
&= (1/2) - (1/2)\Pr[\mathrm{Ab}]. && (4.3.4)
\end{aligned}$$

Combining inequalities (4.3.3) and (4.3.4), we get

$$|\Pr[\lambda' = \lambda] - 1/2| \leq (1/2)\Pr[\mathrm{Ab}]. \qquad (4.3.5)$$

Recall that $\mathcal{A}$'s advantage is defined as $|\Pr[\lambda' = \lambda] - 1/2|$. Since $\mathcal{A}$ has non-negligible probability ε to break the proposed scheme, we therefore can obtain

$$\begin{aligned}
\varepsilon &= |\Pr[\lambda' = \lambda] - 1/2| \\
&\leq (1/2)\Pr[\mathrm{Ab}] && \text{(by Eq. (4.3.5))} \\
&= (1/2)(\Pr[\mathrm{QH}_2{*} \vee \mathrm{SRV\_ERR}]) \\
&\leq (1/2)(\Pr[\mathrm{QH}_2{*}] + \Pr[\mathrm{SRV\_ERR}]) \\
&\leq (1/2)(\Pr[\mathrm{QH}_2{*}] + \frac{q_{SRV}}{2^k}) && \text{(by Eq. (4.3.1))}
\end{aligned}$$

Rewriting the above inequality, we have $\Pr[\mathrm{QH}_2{*}] \geq 2\varepsilon - \frac{q_{SRV}}{2^k}$. If the event QH_2* happens, we claim that the correct answer α (= b^{d_v} mod N_v) will be stored in some entry of Q_h_2 array. Consequently, $\mathcal{B}$ would have non-negligible probability

$$\varepsilon' \geq (q_{H_2}{}^{-1})(2\varepsilon - \frac{q_{SRV}}{2^k})$$

to solve the RSA problem. The running time required for $\mathcal{B}$ is $t' \approx t + t_\lambda(q_{H_1} + q_{H_2} + q_{PCG} + q_{ACG} + q_{SRV})$.

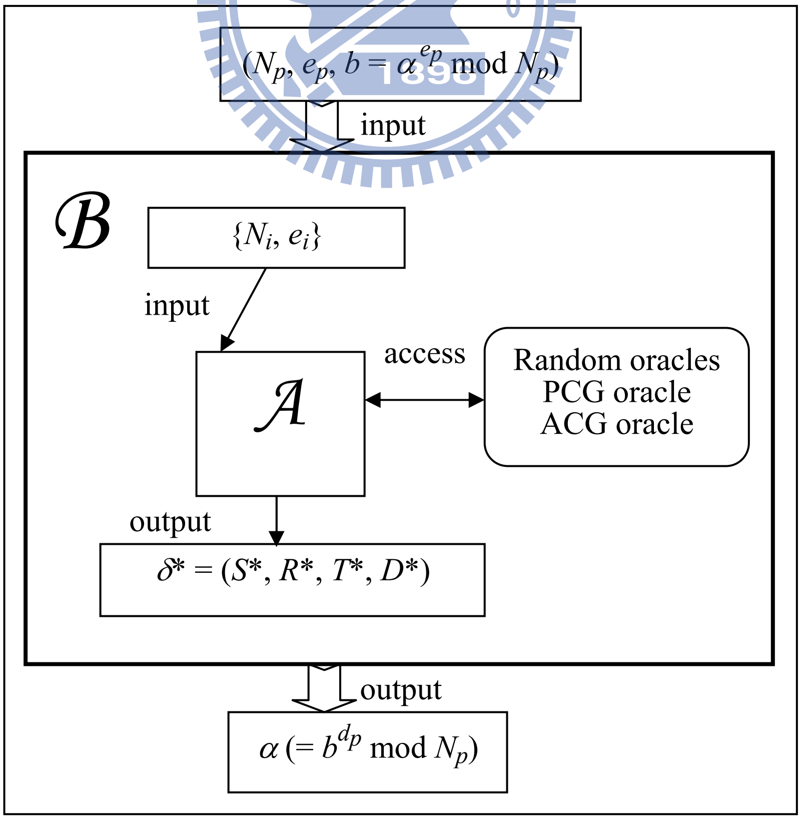
Q.E.D.

Theorem 4.3.2. (Proof of Unforgeability) *The proposed PCAE-(I) scheme is* $(t, q_{h_1}, q_{h_2}, q_{PCG}, q_{ACG}, \varepsilon)$*-secure against existential forgery under adaptive chosen-message attacks (EF-CMA) in the random oracle model if there is no probabilistic polynomial-time adversary that can* (t', ε')*-break the RSA problem, where*

$$\varepsilon' \geq ({q_{h_2}}^{-1})(\varepsilon - 2^{-k}),$$

$$t' \approx t + t_\lambda(q_{h_1} + q_{h_2} + q_{PCG} + q_{PACG}).$$

Here t_λ *is the average running time of one oracle-query.*



$(N_p, e_p, b = \alpha^{e_p} \bmod N_p)$
input
$\mathcal{B}$
$\{N_i, e_i\}$
input
access
$\mathcal{A}$
Random oracles
PCG oracle
ACG oracle
output
$\delta^* = (S^*, R^*, T^*, D^*)$
output
$\alpha\,(= b^{d_p} \bmod N_p)$

Fig. 4.3.8. The proof structure of unforgeability in Theorem 4.3.2

Proof: Fig. 4.3.8 depicts the proof structure of this Theorem. Suppose that a (t, q_{h_1}, q_{h_2}, q_{PCG}, q_{ACG}, ε)-PPTM $\mathcal{A}$ can break the proposed PCAE-(I) scheme with a non-negligible advantage ε under adaptive chosen-message attacks after running in time at most t and asking at most q_{h_i} h_i random oracle (for i = 1 and 2), q_{PCG} PCG and q_{ACG} ACG queries. Then we can take $\mathcal{A}$ as a subroutine to construct another (t', ε')-algorithm $\mathcal{B}$ that solves the RSA problem with respect to the proxy signer's key pair in time at most t' and with the probability ε'. Let all involved parties and parameters be defined the same as those in Section 4.1. The objective of $\mathcal{B}$ is to obtain α (= b^{d_p} mod N_p) by taking (N_p, e_p, $b = \alpha^{e_p}$ mod N_p) as inputs. In this proof, $\mathcal{B}$ simulates a challenger to $\mathcal{A}$ in the following game.

Setup: The challenger $\mathcal{B}$ runs the Setup(1^k) algorithm and sends system's public parameters *params* = {N_i, e_i} to the adversary $\mathcal{A}$.

Phase 1: $\mathcal{A}$ adaptively asks h_i random oracle (for i = 1 and 2), PCG and ACG queries as those defined in Theorem 4.3.1. Note that in the j-th h_2 random oracle, $\mathcal{B}$ directly returns b as the answer, where j is a random positive integer less than or equal to q_{h_2}.

Forgery: Finally, $\mathcal{A}$ outputs a proxy authenticated ciphertext δ^* = (S^*, R^*, T^*, D^*) for his arbitrarily chosen message m^*. If the proxy authenticated ciphertext is valid, $\mathcal{A}$ wins the game.

Analysis of the game: According to the analyses of Theorem 4.3.1, we know that the simulation of each PCG or ACG query will be normally terminated. Besides, $\mathcal{B}$ answers each h_1 or h_2 random oracle with a computationally indistinguishable value without collision. Let AC-V and QH_2 separately be the events that the outputted proxy authenticated ciphertext δ^* = (S^*, R^*, T^*, D^*) is valid and $\mathcal{A}$ has ever asks corresponding $h_2(m^*, Q^*, c^*)$ random oracle. The probability that $\mathcal{A}$ can guess a correct random value without asking h_2 random oracle is not greater than 2^{-k}. Since $\mathcal{A}$ has a non-negligible advantage ε to break the proposed scheme under adaptive chosen-message attacks, we have

$$\begin{aligned}\varepsilon &= \Pr[\text{AC-V}] \\ &\le \Pr[\text{AC-V} \mid \text{QH}_2] + \Pr[\text{AC-V} \mid \neg\text{QH}_2]\end{aligned}$$

$$\leq \Pr[\text{AC-V} \mid \text{QH}_2] + 2^{-k}.$$

Further writing the above inequality, we can also obtain

$$\Pr[\text{AC-V} \mid \text{QH}_2] \geq \varepsilon - 2^{-k}.$$

Seeing that in the j-th h_2 random oracle, $\mathcal{B}$ directly returned b as the result, we claim that when the event (AC-V | QH$_2$) occurs, $\mathcal{B}$ would have the probability of $({q_{h_2}}^{-1})$ to output

$$S^* = \alpha = b^{d_p} \bmod N_p.$$

Therefore, we can express the probability of $\mathcal{B}$ to solve the RSA problem as

$$\varepsilon' \geq ({q_{h_2}}^{-1})(\varepsilon - 2^{-k}).$$

The running time required for $\mathcal{B}$ is $t' \approx t + t_\lambda(q_{h_1} + q_{h_2} + q_{PCG} + q_{ACG})$.

Q.E.D.

According to Theorem 4.3.2, the proposed PCAE-(I) scheme is secure against existential forgery attacks. That is to say, the delegated proxy signer cannot repudiate having generated his authenticated ciphertext. Hence, we obtain the following corollary.

Corollary 4.3.1. The proposed PCAE-(I) scheme satisfies the security requirement of non-repudiation.

5. PCAE-(II) Scheme

In this section, we demonstrate the proposed second proxy CAE (abbreviated to PCAE-(II)) scheme.

5.1 Construction

– **Setup:** Taking as input 1^k, the system authority (SA) selects two large primes p and q satisfying $q \mid (p-1)$, and a generator g of order q, where $|q| = k$. Let h_1: $\{0, 1\}^k \times Z_p^* \rightarrow Z_q$, h_2: $\{0, 1\}^k \times Z_p^* \times Z_p^* \rightarrow Z_q$, h_3: $\{0, 1\}^k \rightarrow Z_p^*$ and h_4: $Z_p^* \rightarrow \{0, 1\}^k$ be collision resistant hash functions. The system publishes public parameters $params = \{p, q, g, h_1, h_2, h_3, h_4\}$. Each user U_i chooses his private key $x_i \in Z_q^*$ and computes the public key as $y_i = g^{x_i} \bmod p$.

– **Proxy-Credential-Generation (PCG):** Let U_o be an original user delegating his signing power to a proxy signer U_p. U_o first chooses $d \in_R Z_q$ to compute

$$T = g^d \bmod p, \tag{5.1.1}$$

$$\sigma = d - x_o h_1(m_w, T) \bmod q, \tag{5.1.2}$$

where m_w is a warrant consisting of the identifier of original signer, proxy signer and designated recipient, the delegation duration and so on. (σ, m_w, T) is then sent to U_p. Upon receiving (σ, m_w, T), U_p computes C as Eq. (5.1.3) and performs Eq. (5.1.4) to check its validity.

$$C = {y_o}^{h_1(m_w, T)} \bmod p, \tag{5.1.3}$$

$$T = g^{\sigma} C \pmod p). \tag{5.1.4}$$

If it does not hold, (σ, m_w, T) is requested to be sent again.

– **Authenticated-Ciphertext-Generation (ACG):** For signing a message $m \in_R \{0, 1\}^k$ on behalf of an original signer U_o, U_p chooses $r \in_R Z_q$ to compute

$$R = g^r h_3(m) \bmod p, \tag{5.1.5}$$

$$K = y_v{}^{\sigma} \bmod p, \tag{5.1.6}$$

$$s = r - x_p h_2(m, C, R) \bmod q, \tag{5.1.7}$$

$$r_1 = s(K \bmod q) \bmod q, \tag{5.1.8}$$

$$r_2 = h_4(K) \oplus m, \tag{5.1.9}$$

and then delivers the warrant m_w and the authenticated ciphertext $\delta = (r_1, r_2, R, T)$ to a designated recipient U_v.

– **Signature-Recovery-and-Verification (SRV):** Upon receiving (δ, m_w), U_v first computes

$$C = y_o{}^{h_1(m_w,\, T)} \bmod p, \tag{5.1.10}$$

$$K = (TC^{-1})^{x_v} \bmod p, \tag{5.1.11}$$

$$s = (K \bmod q)^{-1} r_1 \bmod q, \tag{5.1.12}$$

$$m = r_2 \oplus h_4(K), \tag{5.1.13}$$

and then checks the redundancy embedded in m. U_v can further verify the proxy signature by checking if

$$R = g^s y_p{}^{h_2(m,\, C\,,\, R)} h_3(m) \bmod p. \tag{5.1.14}$$

When the case of a later dispute over repudiation occurs, U_v can reveal the converted proxy signature $\Omega = (R, s, T)$, the warrant m_w and the original message m to prove proxy signer's dishonesty without any additional computation effort or communication overhead. Thus, anyone can verify the converted proxy signature with the assistance of Eqs. (5.1.10) and (5.1.14).

5.2 Correctness

We first show that the verification of Eq. (5.1.4) works correctly. From the right-hand side of Eq. (5.1.4), we have

$$
\begin{aligned}
& g^{\sigma}C \\
&= g^{\sigma} {y_o}^{h_1(m_w,\, T)} && \text{(by Eq. (5.1.3))} \\
&= g^{d - x_o h_1(m_w,\, T)} {y_o}^{h_1(m_w,\, T)} && \text{(by Eq. (5.1.2))} \\
&= g^{d} \\
&= T \pmod{p} && \text{(by Eq. (5.1.1))}
\end{aligned}
$$

which leads to the left-hand side of Eq. (5.1.4).

With the private key x_v and received T, the designated recipient can correctly compute the shared secret K. From the right-hand side of Eq. (5.1.11), we have

$$
\begin{aligned}
& (TC^{-1})^{x_v} \\
&= (g^{\sigma})^{x_v} && \text{(by Eq. (5.1.4))} \\
&= {y_v}^{\sigma} \\
&= K \pmod{p} && \text{(by Eq. (5.1.6))}
\end{aligned}
$$

which leads to the left-hand side of Eq. (5.1.11).

If the authenticated ciphertext (r_1, r_2, R, T) is correctly generated, it will pass the test of Eq. (5.1.14). From the right-hand side of Eq. (5.1.14), we have

$$
\begin{aligned}
& g^{s} {y_p}^{h_2(m,\, C\,,\, R)} h_3(m) \\
&= g^{r - x_p h_2(m,\, C\,,\, R)} {y_p}^{h_2(m,\, C\,,\, R)} h_3(m) && \text{(by Eq. (5.1.7))} \\
&= g^{r} h_3(m) \\
&= R \pmod{p} && \text{(by Eq. (5.1.5))}
\end{aligned}
$$

which leads to the left-hand side of Eq. (5.1.14).

5.3 Security Proofs

We prove that the proposed PCAE-(II) scheme achieves the IND-CCA2 and the EF-CMA security in random oracle models as Theorems 5.3.1 and 5.3.2, respectively.

Theorem 5.3.1. (Proof of Confidentiality) *The proposed scheme is* (t, q_{h_1}, q_{h_2}, q_{h_3}, q_{h_4}, q_{PCG}, q_{ACG}, q_{SRV}, ε)*-secure against indistinguishability under adaptive chosen-ciphertext attacks* (*IND-CCA2*) *in the random oracle model if there is no probabilistic polynomial-time adversary that can* (t', ε')*-break the CDHP, where*

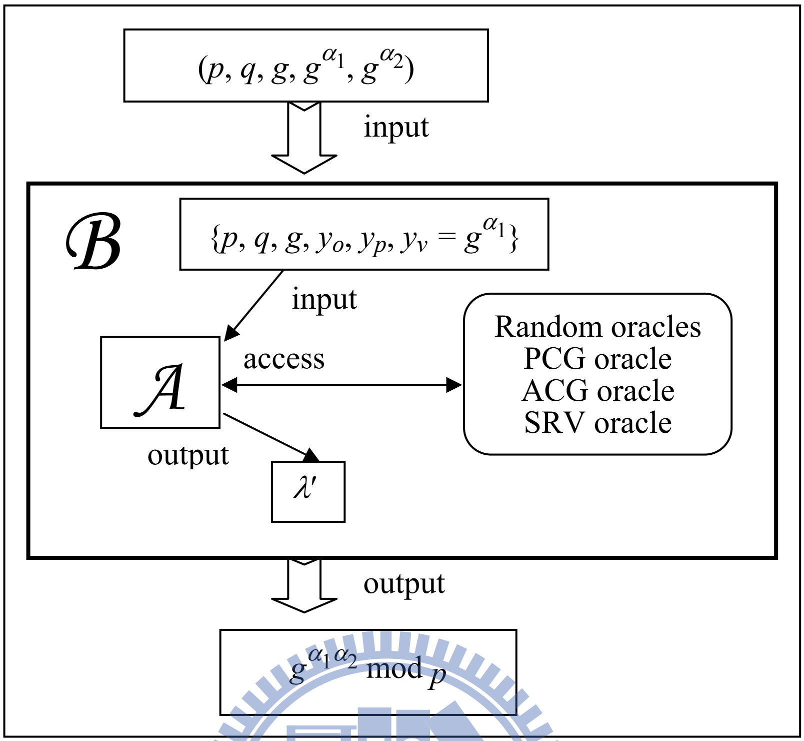
$$\varepsilon' \geq ({q_{h_4}}^{-1})(2\varepsilon - \frac{q_{SRV}(q_{h_2} + q_{h_4} + 1)}{2^k}),$$

$$t' \approx t + t_\lambda(q_{h_3} + 2q_{PCG} + 4q_{ACG} + 3q_{SRV} + 3).$$

Here t_λ *is the time for performing a modular exponentiation over a finite field.*

Proof: Fig. 5.3.1 depicts the proof structure of this Theorem. Suppose that a probabilistic polynomial-time adversary $\mathcal{A}$ can (t, q_{h_1}, q_{h_2}, q_{h_3}, q_{h_4}, q_{PCG}, q_{ACG}, q_{SRV}, ε)-break the proposed scheme with a non-negligible advantage ε under adaptive chosen-ciphertext attacks after running in time at most t and asking at most q_{h_i} h_i random oracle (for i = 1 to 4), q_{PCG} PCG, q_{ACG} ACG and q_{SRV} SRV queries. Then we can construct another algorithm $\mathcal{B}$ that (t', ε')-breaks the CDHP by taking $\mathcal{A}$ as a subroutine. Let all involved parties and parameters be defined the same as those in Section 5.1. The objective of $\mathcal{B}$ is to obtain ($g^{\alpha_1\alpha_2}$ mod p) by taking (p, q, g, g^{α_1}, g^{α_2}) as inputs. In this proof, $\mathcal{B}$ simulates a challenger to $\mathcal{A}$ in the following game.

Setup: The challenger $\mathcal{B}$ first runs the Setup(1^k) algorithm to obtain system's public parameters *params* = {p, q, g} and sets $y_v = g^{\alpha_1}$. Then, *params* and (y_o, y_p, $y_v = g^{\alpha_1}$) are sent to the adversary $\mathcal{A}$.



$(p, q, g, g^{\alpha_1}, g^{\alpha_2})$
input
$\mathcal{B}$
$\{p, q, g, y_o, y_p, y_v = g^{\alpha_1}\}$
input
access
$\mathcal{A}$
Random oracles
PCG oracle
ACG oracle
SRV oracle
output
λ'
output
$g^{\alpha_1\alpha_2}$ mod p

Fig. 5.3.1. The proof structure of confidentiality in Theorem 5.3.1

Phase 1: $\mathcal{A}$ issues the following queries adaptively:

– h_1 *oracle:* When $\mathcal{A}$ queries an h_1 oracle of $h_1(m_w, T)$, $\mathcal{B}$ returns $\mathcal{O}$**-Sim(II)_*h*₁**(m_w, T). The simulated random oracle $\mathcal{O}$**-Sim(II)_*h*₁** operates as Fig. 5.3.2.

```
oracle O-Sim(II)_h_1($m_w$, $T$)    // Let Q_$h_1$[$q_{h_1}$][2] and A_$h_1$[$q_{h_1}$] be two arrays.
 for $i = 0$ to $q_{h_1} - 1$
     if (Q_$h_1$[$i$][0] = $m_w$) and (Q_$h_1$[$i$][1] = $T$) then    // It is an old query.
         exit for;
     else if (Q_$h_1$[$i$][0] = null) then    // It is a new query.
         insert(Q_$h_1$, ($m_w$, $T$)); insert(A_$h_1$, $v_1 \in_R Z_q$); exit for;
     end if
 next $i$
 return A_$h_1$[$i$];
```

Fig. 5.3.2. Algorithm of the simulated random oracle $\mathcal{O}$**-Sim(II)_*h*₁**

– h_2 *oracle:* When $\mathcal{A}$ queries an h_2 oracle of $h_2(m, C, R)$, $\mathcal{B}$ returns $\mathcal{O}$-**Sim(II)_h_2**(m, C, R). The simulated random oracle $\mathcal{O}$-**Sim(II)_h_2** operates as Fig. 5.3.3.

```
oracle $\mathcal{O}$-Sim(II)_$h_2(m, C, R)$   // Let Q_$h_2[q_{h_2}][3]$ and A_$h_2[q_{h_2}]$ be two arrays.
  for $i = 0$ to $q_{h_2} - 1$
      if (Q_$h_2[i][0] = m$) and (Q_$h_2[i][1] = C$) and (Q_$h_2[i][2] = R$) then
          exit for;   // It is an old query.
      else if (Q_$h_2[i][0]$ = null) then   // It is a new query.
          insert(Q_$h_2$, $(m, C, R)$); insert(A_$h_2$, $v_2 \in_R Z_q$); exit for;
      end if
  next $i$
  return A_$h_2[i]$;
```

Fig. 5.3.3. Algorithm of the simulated random oracle $\mathcal{O}$-**Sim(II)_h_2**

– h_3 *oracle:* When $\mathcal{A}$ queries an h_3 oracle of $h_3(K)$, $\mathcal{B}$ returns $\mathcal{O}$-**Sim(II)_h_3**(K). The simulated random oracle $\mathcal{O}$-**Sim(II)_h_3** operates as Fig. 5.3.4.

```
oracle $\mathcal{O}$-Sim_$h_3(m)$   // Let Q_$h_3[q_{h_3}]$ and A_$h_3[q_{h_3}][3]$ be two arrays.
  for $i = 0$ to $q_{h_3} - 1$
      if (Q_$h_3[i] = m$) then   // It is an old query.
          exit for;
      else if (Q_$h_3[i]$ = null) then   // It is a new query.
          insert(Q_$h_3$, $m$); Choose $v_3 \in_R Z_q$;
          if (check(Q_$h_2$, $m$)) = true) then // $h_2(m, *, *)$ has been queried.
              for $j = 0$ to $q_{h_2} - 1$
                  if (Q_$h_2[j][0] = m$) then
                      $R$ = Q_$h_2[j][2]$; insert(A_$h_3$, $(v_3, R, V_3 = Rg^{v_3})$); exit for;
                  end if
              next $j$
          else // $h_2(m, *, *)$ has never been queried.
              insert(A_$h_3$, $(v_3 \in_R Z_q, 1, V_3 = g^{v_3})$);
          end if
          exit for;
      end if
  next $i$
  return A_$h_3[i][2]$;
```

Fig. 5.3.4. Algorithm of the simulated random oracle $\mathcal{O}$-**Sim(II)_h_3**

– *h_4 oracle:* When $\mathcal{A}$ queries an h_4 oracle of $h_4(K)$, $\mathcal{B}$ returns **$\mathcal{O}$-Sim(II)_*h_4***(K). The simulated random oracle **$\mathcal{O}$-Sim(II)_*h_4*** operates as Fig. 5.3.5.

```
oracle O-Sim(II)_h4(K)    // Let Q_h4[$q_{h_4}$] and A_h4[$q_{h_4}$] be two arrays.
for $i = 0$ to $q_{h_4} - 1$
    if (Q_h4[$i$] = $K$) then    // It is an old query.
        exit for;
    else if (Q_h4[$i$] = null) then    // It is a new query.
        insert(Q_h4, $K$); insert(A_h4, $v_4 \in_R \{0, 1\}^k$); exit for;
    end if
next $i$
return A_h4[$i$];
```

Fig. 5.3.5. Algorithm of the simulated random oracle **$\mathcal{O}$-Sim(II)_*h_4***

– *PCG queries:* When $\mathcal{A}$ makes a PCG query, $\mathcal{B}$ chooses a proper m_w and then returns (m_w, **$\mathcal{O}$-Sim(II)_*PCG***(m_w)) as a result. The simulated PCG oracle **$\mathcal{O}$-Sim(II)_*PCG*** operates as Fig. 5.3.5.

```
oracle O-Sim(II)_PCG($m_w$)
do
    Choose $\sigma$, $v_1 \in_R Z_q$;
    Compute $T = g^{\sigma} y_o^{v_1} \bmod p$;
while (check(Q_h1, ($m_w$, $T$)) = true)
insert(Q_h1, ($m_w$, $T$)); insert(A_h1, $v_1$);    // define $h_1(m_w, T) = v_1$
return ($\sigma$, $T$);
```

Fig. 5.3.6. Algorithm of the simulated PCG oracle **$\mathcal{O}$-Sim(II)_*PCG***

– *ACG queries:* When $\mathcal{A}$ makes an ACG query for some message m, $\mathcal{B}$ returns **$\mathcal{O}$-Sim(II)_*ACG***(m) as a result. The simulated ACG oracle **$\mathcal{O}$-Sim(II)_*ACG*** operates as Fig. 5.3.7.

```
oracle $\mathcal{O}$-Sim(II)_ACG($m$)
  Choose a proper $m_w$;
  $(\sigma, T)$ = $\mathcal{O}$-Sim(II)_PCG($m_w$); $V_3$ = $\mathcal{O}$-Sim_$h_3$($m$); $v_1$ = $\mathcal{O}$-Sim_$h_1$($m_w$, $T$);
  Compute $C = y_o^{v_1} \bmod p$; $K = y_v^{\sigma} \bmod p$;
  do
      Choose $s, v_2 \in_R Z_q$; Compute $R = g^s y_p^{v_2} V_3 \bmod p$;
  while (check(Q_$h_2$, ($m$, $C$, $R$)) = true)
  insert(Q_$h_2$, ($m$, $C$, $R$)); insert(A_$h_2$, $v_2$);    // define $h_2(m, C, R) = v_2$
  Compute $r_1 = s(K \bmod q) \bmod q$; $r_2$ = $\mathcal{O}$-Sim_$h_4$($K$) $\oplus m$;
  return $\delta = (r_1, r_2, R, T)$ along with $m_w$;
```

Fig. 5.3.7. Algorithm of the simulated ACG oracle $\mathcal{O}$-**Sim(II)_*ACG***

– *SRV queries:* When $\mathcal{A}$ makes an SRV query for some authenticated ciphertext δ with a warrant m_w, $\mathcal{B}$ returns $\mathcal{O}$-**Sim(II)_*SRV***(δ, m_w) as the result. The simulated SRV oracle $\mathcal{O}$-**Sim(II)_*SRV*** operates as Fig. 5.3.8. Note that the symbol '*' denotes wildcard.

```
oracle $\mathcal{O}$-Sim(II)_SRV($\delta$, $m_w$)    // $\delta = (r_1, r_2, R, T)$
  $v_1$ = $\mathcal{O}$-Sim(II)_$h_1$($m_w$, $T$); Compute $C = y_o^{v_1} \bmod p$;
  if (check(Q_$h_2$, (*, $C$, $R$)) = true) then // $h_2$(*, $C$, $R$) has ever been queried.
      $m$ = Q_$h_2$[$j$][0];    // Assume that Q_$h_2$[$j$][1] = $C$ and Q_$h_2$[$j$][2] = $R$.
      $v_4 = r_2 \oplus m$;
      if (check(A_$h_4$, $v_4$)) = true) then
          $K$ = Q_$h_4$[$j$];    // Assume that A_$h_4$[$j$] = $v_4$.
          $s = r_1(K \bmod q)^{-1} \bmod q$;
          if ($R = g^s y_p^{h_2(m,\, C,\, R)} h_3(m) \bmod p$) then
              return ($m$, $R$, $s$, $T$);
          else
              return ⊥;
          end if
      else
          return ⊥;
      end if
  else // $h_2$(*, $C$, $R$) has never been queried.
      return ⊥;
  end if
```

Fig. 5.3.8. Algorithm of the simulated SRV oracle $\mathcal{O}$-**Sim(II)_*SRV***

Challenge: $\mathcal{A}$ generates two messages, m_0 and m_1, of the same length. The challenger $\mathcal{B}$ flips a coin $\lambda \leftarrow \{0, 1\}$ and produces an authenticated ciphertext $\delta^* = (r_1^*, r_2^*, R^*, T^*)$ for m_λ by running the simulated **Sim(II)_Challenge**(m_λ). The algorithm of **Sim(II)_Challenge** operates as Fig. 5.3.9.

```
algorithm Sim(II)_Challenge($m_\lambda$)
  Choose a proper $m_w^*$;
  do
      Choose $v_1 \in_R Z_q$; Compute $C^* = y_o^{v_1} \bmod p$ and $T^* = (g^{\alpha_2})C^* \bmod p$;
  while (check(Q_$h_1$, ($m_w^*$, $T^*$)) = true)
  insert(Q_$h_1$, ($m_w^*$, $T^*$)); insert(A_$h_1$, $v_1$);    // define $h_1(m_w^*, T^*) = v_1$
  $V_3^*$ = $\mathcal{O}$-Sim(II)_$h_3$($m_\lambda$);
  do
      Choose $s^*, v_2 \in_R Z_q$; Compute $R^* = g^{s^*} y_p^{v_2} V_3^* \bmod p$;
  while (check(Q_$h_2$, ($m_\lambda$, $C^*$, $R^*$)) = true)
  insert(Q_$h_2$, ($m_\lambda$, $C^*$, $R^*$)); insert(A_$h_2$, $v_2$);    // define $h_2(m_\lambda, C^*, R^*) = v_2$
  Choose $r_1^* \in_R Z_q$ and $v_4 \in_R \{0, 1\}^k$;
  Compute $r_2^* = v_4 \oplus m_\lambda$;
     // Implicitly define $h_4(K^*) = v_4$, where $K^* = g^{\alpha_1\alpha_2}$ and $\mathcal{B}$ does not know it.
  return $\delta^* = (r_1^*, r_2^*, R^*, T^*)$ and $m_w^*$;
```

Fig. 5.3.9. Algorithm of the simulated **Sim(II)_Challenge**

Phase 2: $\mathcal{A}$ makes new queries as those stated in Phase 1 except an SRV query for the target ciphertext δ^*.

Analysis of the game: Consider above simulations of PCG and ACG queries. We can observe that the simulated proxy credential and authenticated ciphertext are computationally indistinguishable from those generated by a real scheme. Hence, we refer simulations of PCG and ACG queries to be perfect. Then we evaluate the simulation of SRV queries. From the algorithms of $\mathcal{O}$-**Sim(II)_*SRV***, one can find out that it is possible for an SRV query of some valid $\delta = (r_1, r_2, R, T)$ to return an error symbol $\perp$ on condition that $\mathcal{A}$ has the ability to produce δ without asking corresponding $h_2(m, C, R)$ or $h_4(K)$ random oracles in advance.

Let SRV_ERR be the event that an SRV query returns an error symbol $\perp$ for some valid δ during the entire game and AC-V an event that an authenticated ciphertext δ submitted by $\mathcal{A}$ is valid. QH_2 and QH_4 separately denote events that $\mathcal{A}$ has ever asked corresponding h_2 and h_4 random oracles beforehand. Then we can express the error probability of any SRV query as

$$
\begin{aligned}
&\Pr[\text{AC-V} \mid \neg\mathrm{QH}_4 \vee \neg\mathrm{QH}_2] \\
&= \frac{\Pr[\text{AC-V} \wedge (\neg\mathrm{QH}_4 \vee \neg\mathrm{QH}_2)]}{\Pr[\neg\mathrm{QH}_4 \vee \neg\mathrm{QH}_2]} \\
&= \frac{\Pr[(\text{AC-V} \wedge \neg\mathrm{QH}_4) \vee (\text{AC-V} \wedge \neg\mathrm{QH}_2)]}{\Pr[\neg\mathrm{QH}_4 \vee \neg\mathrm{QH}_2]} \\
&\le \frac{\Pr[\text{AC-V} \wedge \neg\mathrm{QH}_4]}{\Pr[\neg\mathrm{QH}_4 \vee \neg\mathrm{QH}_2]} + \frac{\Pr[\text{AC-V} \wedge \neg\mathrm{QH}_2]}{\Pr[\neg\mathrm{QH}_4 \vee \neg\mathrm{QH}_2]} \\
&= \Big(\frac{\Pr[\text{AC-V} \wedge (\neg\mathrm{QH}_4 \wedge \neg\mathrm{QH}_2)]}{\Pr[\neg\mathrm{QH}_4 \vee \neg\mathrm{QH}_2]} + \frac{\Pr[\text{AC-V} \wedge (\neg\mathrm{QH}_4 \wedge \mathrm{QH}_2)]}{\Pr[\neg\mathrm{QH}_4 \vee \neg\mathrm{QH}_2]}\Big) \\
&\quad + \Big(\frac{\Pr[\text{AC-V} \wedge (\neg\mathrm{QH}_2 \wedge \neg\mathrm{QH}_4)]}{\Pr[\neg\mathrm{QH}_4 \vee \neg\mathrm{QH}_2]} + \frac{\Pr[\text{AC-V} \wedge (\neg\mathrm{QH}_2 \wedge \mathrm{QH}_4)]}{\Pr[\neg\mathrm{QH}_4 \vee \neg\mathrm{QH}_2]}\Big) \\
&= \frac{\Pr[\text{AC-V} \wedge (\neg\mathrm{QH}_4 \wedge \mathrm{QH}_2)]}{\Pr[\neg\mathrm{QH}_4 \vee \neg\mathrm{QH}_2]} + \frac{2\Pr[\text{AC-V} \wedge (\neg\mathrm{QH}_2 \wedge \neg\mathrm{QH}_4)]}{\Pr[\neg\mathrm{QH}_4 \vee \neg\mathrm{QH}_2]} \\
&\quad + \frac{\Pr[\text{AC-V} \wedge (\neg\mathrm{QH}_2 \wedge \mathrm{QH}_4)]}{\Pr[\neg\mathrm{QH}_4 \vee \neg\mathrm{QH}_2]} \\
&\le \frac{\Pr[(\text{AC-V} \wedge \mathrm{QH}_2) \wedge \neg\mathrm{QH}_4)]}{\Pr[\neg\mathrm{QH}_4]} + \frac{2\Pr[\text{AC-V} \wedge (\neg\mathrm{QH}_2 \wedge \neg\mathrm{QH}_4)]}{\Pr[\neg\mathrm{QH}_2 \wedge \neg\mathrm{QH}_4]} \\
&\quad + \frac{\Pr[(\text{AC-V} \wedge \mathrm{QH}_4) \wedge \neg\mathrm{QH}_2]}{\Pr[\neg\mathrm{QH}_2]} \\
&= \Pr[(\text{AC-V} \wedge \mathrm{QH}_2) \mid \neg\mathrm{QH}_4] + 2\Pr[\text{AC-V} \mid (\neg\mathrm{QH}_2 \wedge \neg\mathrm{QH}_4)] \\
&\quad + \Pr[(\text{AC-V} \wedge \mathrm{QH}_4) \mid \neg\mathrm{QH}_2] \\
&\le \Pr[\mathrm{QH}_2 \mid \neg\mathrm{QH}_4] + 2\Pr[\text{AC-V} \mid (\neg\mathrm{QH}_2 \wedge \neg\mathrm{QH}_4)] + \Pr[\mathrm{QH}_4 \mid \neg\mathrm{QH}_2]
\end{aligned}
$$

$$\le \frac{q_{h_2}}{2^k} + \frac{1}{2^{2k-1}} + \frac{q_{h_4}}{2^k}$$

$$\le \frac{q_{h_2}}{2^k} + \frac{1}{2^k} + \frac{q_{h_4}}{2^k}$$

$$= \frac{q_{h_2} + q_{h_4} + 1}{2^k}.$$

Since $\mathcal{A}$ can make at most q_{SRV} SRV queries, we can further express the probability of SRV_ERR as

$$\Pr[\text{SRV\_ERR}] \le \frac{q_{SRV}(q_{h_2} + q_{h_4} + 1)}{2^k}. \qquad (5.3.15)$$

Additionally, in the challenge phase, $\mathcal{B}$ has returned a simulated authenticated ciphertext $\delta^* = (r_1^*, r_2^*, R^*, T^*)$ where $T^* = (g^{\alpha_2})y_o{}^{v_1} \bmod p$, which implies the shared secret K^* is implicitly defined as $K^* = g^{\alpha_1\alpha_2} \bmod p$. Let GP be an event that the entire simulation game does not abort. Obviously, if the adversary $\mathcal{A}$ never makes an $h_4(K^*)$ query in Phase 2, the entire simulation game could be normally terminated. We denote the event that $\mathcal{A}$ does ask such an query in Phase 2 by $\text{QH}_4{}^*$. When the entire simulation game does not abort, it can be seen $\mathcal{A}$ gains no advantage in guessing λ due to the randomness of output of random oracles, i.e.,

$$\Pr[\lambda' = \lambda \mid \text{GP}] = 1/2. \qquad (5.3.16)$$

Rewriting the expression of $\Pr[\lambda' = \lambda]$, we have

$$\begin{aligned} \Pr[\lambda' = \lambda] &= \Pr[\lambda' = \lambda \mid \text{GP}]\Pr[\text{GP}] + \Pr[\lambda' = \lambda \mid \neg\text{GP}]\Pr[\neg\text{GP}] \\ &\le (1/2)\Pr[\text{GP}] + \Pr[\neg\text{GP}] && \text{(by Eq. (5.3.16))} \\ &= (1/2)(1 - \Pr[\neg\text{GP}]) + \Pr[\neg\text{GP}] \\ &= (1/2) + (1/2)\Pr[\neg\text{GP}]. && (5.3.17) \end{aligned}$$

On the other hand, we can also derive that

$$
\begin{aligned}
\Pr[\lambda' = \lambda] &\geq \Pr[\lambda' = \lambda \mid \mathrm{GP}] \Pr[\mathrm{GP}] \\
&= (1/2)(1 - \Pr[\neg\mathrm{GP}]) \\
&= (1/2) - (1/2)\Pr[\neg\mathrm{GP}]. \qquad (5.3.18)
\end{aligned}
$$

With inequalities (5.3.17) and (5.3.18), we know that

$$| \Pr[\lambda' = \lambda] - 1/2 | \leq (1/2)\Pr[\neg\mathrm{GP}]. \qquad (5.3.19)$$

Recall that in Definition 3.3.3, $\mathcal{A}$'s advantage is defined as $Adv(\mathcal{A}) = | \Pr[\lambda' = \lambda] - 1/2 |$. By assumption, $\mathcal{A}$ has non-negligible probability ε to break the proposed scheme. We therefore have

$$
\begin{aligned}
\varepsilon &= | \Pr[\lambda' = \lambda] - 1/2 | \\
&\leq (1/2)\Pr[\neg\mathrm{GP}] \qquad \text{(by Eq. (5.3.19))} \\
&= (1/2)(\Pr[\mathrm{QH}_4^* \vee \mathrm{SRV\_ERR}]) \\
&\leq (1/2)(\Pr[\mathrm{QH}_4^*] + \Pr[\mathrm{SRV\_ERR}])
\end{aligned}
$$

Combining Eq. (5.3.15) and rewriting above inequality, we get

$$
\begin{aligned}
\Pr[\mathrm{QH}_4^*] &\geq 2\varepsilon - \Pr[\mathrm{SRV\_ERR}] \\
&\geq 2\varepsilon - \frac{q_{SRV}(q_{h_2} + q_{h_4} + 1)}{2^k}.
\end{aligned}
$$

If the event QH_4^* happens, we claim that a correct answer $K^* = g^{\alpha_1\alpha_2}$ to the CDHP will be stored in some entry of the Q_h_4 array. Consequently, $\mathcal{B}$ has non-negligible probability

$$\varepsilon' \geq ({q_{h_4}}^{-1})(2\varepsilon - \frac{q_{SRV}(q_{h_2} + q_{h_4} + 1)}{2^k})$$

to solve the CDHP. The computation time required for $\mathcal{B}$ is $t' \approx t + t_\lambda(q_{h_3} + 2q_{PCG} + 4q_{ACG} + 3q_{SRV} + 3)$.

Q.E.D.

In 2000, Pointcheval and Stern introduced the Forking lemma [PS00] to prove EF-CMA security for generic digital signature schemes in the random oracle model. If we apply their techniques to prove our scheme, we can also obtain the generic result as follows.

(The Forking Lemma) *In the random oracle mode, let* $(\mathcal{G}, \Sigma, \mathcal{V})$ *be a generic signature scheme and* $\mathcal{A}$ *a probabilistic polynomial-time Turing machine whose input only consists of public data. We denote respectively by* N_1 *and* N_2 *the number of queries that* $\mathcal{A}$ *can ask to the random oracle and the number of queries that* $\mathcal{A}$ *can ask to the signer. Assume that, within a time bound* T*,* $\mathcal{A}$ *produces, with probability*$\varepsilon \geq 10(N_2 + 1)(N_2 + N_1)/2^k$*, a valid signature* $(m, \sigma_1, h, \sigma_2)$ *where* $\sigma_1 = (m_w, R, T)$*,* $h = (h_2(m, C, R), h_3(m))$ *and* $\sigma_2 = s$*. If the triples* (σ_1, h, σ_2) *can be simulated without knowing the private key with an indistinguishable distribution probability, then there is another machine which has control over the machine obtained from* $\mathcal{A}$ *replacing interaction with the signer by simulation and produces two valid signatures* $(m, \sigma_1, h, \sigma_2)$ *and* $(m, \sigma_1, h', \sigma_2')$ *such that* $h_2(m, C, R) \neq h'_2(m, C, R)$ *in the expected time* $T' \leq 120686T/\varepsilon$*.*

More concretely, in our scheme, we can first obtain two equations below:

$$R = g^{s} y_p^{h_2(m,\, C,\, R)} h_3(m) \bmod p,$$

$$R = g^{s'} y_p^{h'_2(m,\, C,\, R)} h_3(m) \bmod p.$$

By combining above two equalities, we can further derive the private key x_p as

$$x_p = (s - s')/(h'_2(m, C, R) - h_2(m, C, R)).$$

Yet, to give a tight reduction from the hardness of DLP to our proposed scheme, we present another more detailed security proof and the advantage analysis as Theorem 5.3.2.

Theorem 5.3.2. (Proof of Unforgeability) *The proposed scheme is* $(t, q_{h_1}, q_{h_2}, q_{h_3}, q_{PCG}, q_{ACG}, \varepsilon)$*-secure against existential forgery under adaptive chosen-message attacks (EF-CMA) in the random oracle model if there is no probabilistic polynomial-time adversary that can*

(t', ε')-break the DLP, where

$$\varepsilon' \geq (2^{-1})(\varepsilon - 2^{-2k})(1 + 4^{-1}(\varepsilon - 2^{-2k})^2(2^{-1} + q_{h_2}{}^{-1})),$$

$$t' \approx t + t_\lambda(q_{h_3} + 2q_{PCG} + 4q_{ACG}).$$

Here t_λ is the time for performing a modular exponentiation over a finite field.

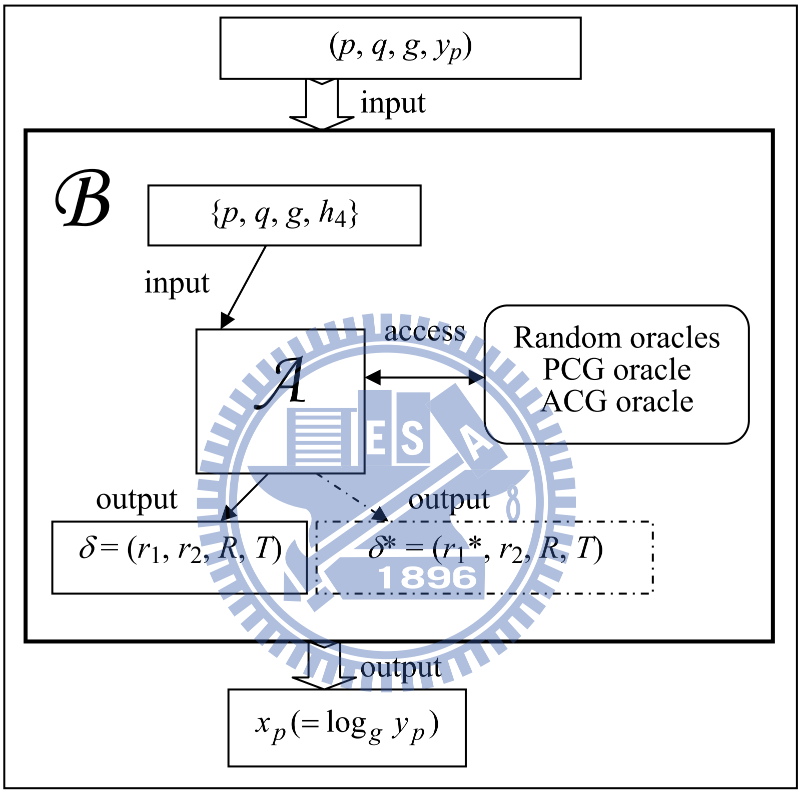


Fig. 5.3.10. The proof structure of unforgeability in Theorem 5.3.2

Proof: Fig. 5.3.10 depicts the proof structure of this Theorem. Suppose that a probabilistic polynomial-time adversary $\mathcal{A}$ can $(t, q_{h_1}, q_{h_2}, q_{h_3}, q_{PCG}, q_{ACG}, \varepsilon)$-break the proposed PCAE-(II) scheme with a non-negligible advantage ε under adaptive chosen-message attacks after running in time at most t and asking at most q_{h_i} h_i random oracle (for i = 1 to 3), q_{PCG} PCG and q_{ACG} ACG queries. Then we can construct another algorithm $\mathcal{B}$ that (t', ε')-breaks the DLP by taking $\mathcal{A}$ as a subroutine. Let all involved parties and notations be defined the same as those in Section 5.1, h_4 a collision resistant hash function and (h_1, h_2, h_3)

random oracles. The objective of $\mathcal{B}$ is to obtain $x_p (= \log_g y_p)$ by taking (p, q, g, y_p) as inputs. In this proof, $\mathcal{B}$ simulates a challenger to $\mathcal{A}$ in the following game.

Setup: The challenger $\mathcal{B}$ runs the Setup(1^k) algorithm to obtain system's public parameters *params* = $\{p, q, g, h_4\}$ and comes up with a random tape composed of a long sequence of random bits. Then $\mathcal{B}$ simulates one or two runs of the proposed scheme to the adversary $\mathcal{A}$ on input *params*, y_o, y_p, $y_v = g^{\alpha} \bmod p$ where $\alpha \in_R Z_q$, and the random tape.

Phase 1: $\mathcal{A}$ adaptively asks h_1, h_2 and h_3 random oracles, PCG and ACG queries as those defined in Theorem 5.3.1.

Analysis of the game: According to analyses of Theorem 5.3.1, simulations of PCG and ACG queries are perfect. Namely, the adversary $\mathcal{A}$ can not distinguish whether he is playing in either a simulation or a real scheme. Let AC-V be an event that $\mathcal{A}$ forges a valid authenticated ciphertext $\delta = (r_1, r_2, R, T)$ for his arbitrarily chosen message m. Since $\mathcal{A}$ has non-negligible probability ε to break the proposed scheme under adaptive chosen-message attacks by the initial assumption, we know that

$$\Pr[\text{AC-V}] = \varepsilon.$$

Now we further consider a situation where $\mathcal{A}$ is able to output a valid δ without asking h_2 and h_3 random oracles in advance. Let NH be an event that $\mathcal{A}$ guesses correct output values of $h_2(m, C, R)$ and $h_3(m)$ without asking random oracles, i.e., $\Pr[\text{NH}] \le 2^{-2k}$. Then, we can express the probability that $\mathcal{A}$ outputs a valid forgery $\delta = (r_1, r_2, R, T)$ after asking $h_2(m, C, R)$ and $h_3(m)$ random oracles as

$$\Pr[\text{AC-V} \mid \neg\text{NH}] \ge (\varepsilon - 2^{-2k}).$$

$$= \varepsilon^*$$

With the initially selected private key α, $\mathcal{B}$ first recovers

$$m = r_2 \oplus h_4((T(y_o^{h_1(m_w, T)})^{-1})^{\alpha} \bmod p),$$

$$s = (((T(y_o^{h_1(m_w, T)})^{-1})^{\alpha} \bmod p) \bmod q)^{-1} r_1 \bmod q.$$

Then $\mathcal{B}$ checks the entry of A_h_3 array with respect to the $h_3(m)$ random oracle query. By the algorithm of $\mathcal{O}$-**Sim(II)_*h*₃**(m) in Fig. 5.3.4, it can be seen that the simulated result of $h_3(m)$ random oracle would be in the form of either Rg^{v_3} or g^{v_3}. If $h_3(m) = Rg^{v_3}$, we know that

$$R = g^s y_p^{h_2(m,\, C,\, R)} h_3(m) \bmod p$$

$$= g^s y_p^{h_2(m,\, C,\, R)} Rg^{v_3} \bmod p$$

$$\Rightarrow \quad g^{-v_3} = g^s y_p^{h_2(m,\, C,\, R)} \bmod p$$

Consequently, $\mathcal{B}$ can derive the private key x_p by computing

$$x_p = (-v_3 - s)h_2(m, C, R)^{-1} \bmod q.$$

When $\Pr[h_3(m) = Rg^{v_3}] = 1$, we obtain a tight security reduction. On the contrary, when $\Pr[h_3(m) = Rg^{v_3}] = 0$, we get a loose security reduction just like the Forking lemma. As the provided random tape is composed of a long sequence of random bits which are statistically random and unpredictable, we can let $\Pr[h_3(m) = Rg^{v_3}] = 2^{-1}$ eclectically. Specially, if the event (AC-V $\wedge$ $\neg$NH) happens, $\mathcal{B}$ would have the probability of 2^{-1} to solve the DLP in the first simulation.

In case that $h_3(m) = g^{v_3}$, $\mathcal{B}$ has to launch the second simulation and we know such a situation happens with the probability of $(1 - 2^{-1}) = 2^{-1}$. $\mathcal{B}$ again runs $\mathcal{A}$ on input *params*, y_o, y_p, $y_v = g^{\alpha} \bmod p$ where $\alpha \in_R Z_q$, and the same random tape. Since $\mathcal{A}$ is given the same sequence of random bits, we can anticipate that the *i*-th random query $\mathcal{A}$ asks will always be the same as the one during the first simulation. In this simulation, $\mathcal{B}$ returns identical results as those he responds in the first time until $\mathcal{A}$ makes the $h_2(m, C, R)$ query. At this time, $\mathcal{B}$ directly gives another new answer $v_2^* \in_R Z_q$ rather than original v_2. From the statement of “Forking lemma”, we can learn that when $\mathcal{A}$ finally makes another valid forgery $\delta^* = (r_1^*, r_2^*, R, T^*)$ where $h_2(m, C, R) \neq h_2^*(m, C, R)$ or that $h_3(m) = Rg^{v_3}$ this time, $\mathcal{B}$ could solve the DLP with non-negligible probability. To analyze $\mathcal{B}$’s success probability, we use the “Splitting lemma” [PS00] described below:

Let X and Y be the sets of possible sequences of random bits and random function values provided to $\mathcal{A}$ before and after the $h_2(m, C, R)$ query is issued, respectively. It follows that on inputting a random value $(x \,||\, y)$ for any $x \in X$ and $y \in Y$, $\mathcal{A}$ returns a valid forgery with non-negligible probability ε, i.e.,

$$\Pr_{x \in X,\, y \in Y}[\text{AC-V}] = \varepsilon.$$

By the "Splitting lemma", there is a subset $D \in X$ such that

(a). $\Pr[x \in D] = |D| \cdot |X|^{-1} \geq 2^{-1}\varepsilon.$

(b). $\forall x \in D$, $\Pr_{y \in Y}[\text{AC-V}] \geq 2^{-1}\varepsilon.$

If we let $\rho \in D$ and $y' \in Y$ separately be the supplied sequences of random bits and random function values before and after $\mathcal{A}$ makes the $h_2(m, C, R)$ query, $\mathcal{A}$ is able to make a valid forgery in the second simulation with the probability of at least $(2^{-1}\varepsilon)^2 = 4^{-1}\varepsilon^2$, i.e.,

$$\Pr_{\rho \in D,\, y' \in Y}[\text{AC-V}] \geq 4^{-1}\varepsilon^2.$$

Since we have let $\Pr[h_3(m) = Rg^{v_3}] = 2^{-1}$ and the probability that $\mathcal{A}$ eventually returns another valid $\delta^* = (r_1^*, r_2^*, R, T^*)$ with $h_2(m, C, R) \neq h_2^*(m, C, R)$ is $q_{h_2}{}^{-1}$, the probability of $\mathcal{B}$ to solve the DLP in the second simulation can be represented as

$$(\varepsilon^*)(4^{-1}\varepsilon^{*2})(2^{-1} + q_{h_2}{}^{-1})$$

$$=(\varepsilon - 2^{-2k})(4^{-1}(\varepsilon - 2^{-2k})^2)(2^{-1} + q_{h_2}{}^{-1})$$

$$= 4^{-1}(\varepsilon - 2^{-2k})^3(2^{-1} + q_{h_2}{}^{-1}).$$

Considering the results of two rounds of simulation, we can obtain that after the second simulation, $\mathcal{B}$ could solve the DLP with non-negligible probability

$$\varepsilon' \geq (2^{-1})(\varepsilon^*) + (1 - 2^{-1})((\varepsilon^*)(4^{-1}\varepsilon^{*2})(2^{-1} + q_{h_2}{}^{-1}))$$

$$= (2^{-1})(\varepsilon - 2^{-2k}) + (1 - 2^{-1})(4^{-1}(\varepsilon - 2^{-2k})^3(2^{-1} + q_{h_2}{}^{-1}))$$

$$= (2^{-1})(\varepsilon - 2^{-2k})(1 + 4^{-1}(\varepsilon - 2^{-2k})^2(2^{-1} + {q_{h_2}}^{-1})).$$

Moreover, the computation time required for $\mathcal{B}$ in one simulation is

$$t + t_\lambda(q_{h_3} + 2q_{PCG} + 4q_{ACG}).$$

We therefore can compute the total computation time for $\mathcal{B}$ as

$$t' \approx (2^{-1})(t + t_\lambda(q_{h_3} + 2q_{PCG} + 4q_{ACG}))$$

$$+ (1 - 2^{-1})(t + t_\lambda(q_{h_3} + 2q_{PCG} + 4q_{ACG}))$$

$$= t + t_\lambda(q_{h_3} + 2q_{PCG} + 4q_{ACG}).$$

Q.E.D.

According to Theorem 5.3.2, the proposed PCAE-(II) scheme is secure against existential forgery attacks. That is, the delegated proxy signer can not repudiate having generated his authenticated ciphertext. Hence, we obtain the following corollary.

Corollary 5.3.1. *The proposed PCAE-(II) scheme satisfies the security requirement of non-repudiation.*

5.4 Group-Oriented Variant

In this subsection, we modify the proposed PCAE-(II) scheme to present a group-oriented variant. This variant allows a proxy signer to generate an authenticated ciphertext on behalf of an original signing group composed of n signers while only a designated recipient can decrypt the ciphertext and verify its corresponding multi-signature. The detailed construction is described as follows:

– **Setup:** Taking as input 1^k, the system authority (SA) selects two large primes p and q satisfying $q \mid (p - 1)$, and a generator g of order q, where $|q| = k$. Let h_1: $\{0, 1\}^k \times Z_p^* \rightarrow Z_q$,

h_2: $\{0, 1\}^k \times Z_p^* \times Z_p^* \times Z_p^* \rightarrow Z_q$ and h_3: $Z_p^* \rightarrow \{0, 1\}^k$ be collision resistant hash functions. The system's public parameters *params* = $\{p, q, g, h_1, h_2, h_3\}$. Each user U_i chooses his private key $x_i \in Z_q$ and computes the public key as $y_i = g^{x_i} \bmod p$.

– **Proxy-Credential-Generation (PCG):** Let $O = \{U_1, U_2, \ldots, U_n\}$ be a group of n original users delegating their signing power to a proxy signer U_p. With the following steps, $U_i \in O$ distributes the proxy share to U_p:

Step 1 $U_i \in O$ first chooses $d_i \in_R Z_q$ to compute

$$T_i = g^{d_i} \bmod p, \tag{5.4.1}$$

and then sends T_i to U_p and $U_j \in O$, for $j \neq i$.

Step 2 Upon receiving all T_j's, U_i computes

$$T = \prod_{j=1}^{n} T_j \bmod p, \tag{5.4.2}$$

$$\sigma_i = d_i - x_i h_1(m_w, T) \bmod q, \tag{5.4.3}$$

where m_w is a warrant consisting of the identifier of original signers, proxy signer and designated recipient, the delegation duration and so on. (σ_i, m_w, T) is then sent to U_p.

Step 3 Upon receiving (σ_i, m_w, T), U_p computes

$$C_i = y_i^{h_1(m_w, T)} \bmod p, \tag{5.4.4}$$

and verifies whether

$$T_i = g^{\sigma_i} C_i \pmod{p}. \tag{5.4.5}$$

If it does not hold, (σ_i, m_w, T) is requested to be sent again.

We show that the verification of Eq. (5.4.5) works correctly. From the right-hand side of Eq. (5.4.5), we have

$$g^{\sigma_i} C_i$$

$$= g^{d_i - x_i h_1(x_w, T)} {y_i}^{h_1(m_w, T)} \qquad \text{(by Eqs. (5.4.3) and (5.4.4))}$$

$$= g^{d_i}$$

$$= T_i \pmod p \qquad \text{(by Eq. (5.4.1))}$$

which leads to the left-hand side of Eq. (5.4.5).

– **Authenticated-Ciphertext-Generation (ACG):** For signing a message $m \in_R \{0, 1\}^k$ on behalf of the original signing group O, U_p chooses $r \in_R Z_q$ to compute

$$R = g^r \bmod p, \tag{5.4.6}$$

$$\sigma = \sum_{i=1}^{n} \sigma_i, \tag{5.4.7}$$

$$C = \prod_{i=1}^{n} C_i \bmod p, \tag{5.4.8}$$

$$K = {y_v}^{\sigma} \bmod p, \tag{5.4.9}$$

$$s = r + (\sigma - x_p h_2(m, C, K, R)) \bmod q, \tag{5.4.10}$$

$$r_2 = h_3(K) \oplus m, \tag{5.4.11}$$

and then delivers the warrant m_w and the authenticated ciphertext $\delta = (s, r_2, R, T)$ to a designated recipient U_v.

– **Signature-Recovery-and-Verification (SRV):** Upon receiving (δ, m_w), U_v first computes C as Eq. (5.4.8) and derives K as

$$K = (TC^{-1})^{x_v} \bmod p. \tag{5.4.12}$$

He then recovers the message as

$$m = r_2 \oplus h_3(K), \tag{5.4.13}$$

and checks the redundancy embedded in m. U_v can further verify the proxy multi-signature by checking if

$$RT = g^s y_p{}^{h_2(m,\, C\,,\, K,\, R)} C \bmod p. \tag{5.4.14}$$

The correctness of Eqs. (5.4.13) and (5.4.14) can be easily confirmed. From the right-hand side of Eq. (5.4.13), we have

$$\begin{aligned}
& r_2 \oplus h_3(K) \\
&= r_2 \oplus h_3((TC^{-1})^{x_v} \bmod p) && \text{(by Eq. (5.4.12))} \\
&= r_2 \oplus h_3((g^{\sigma})^{x_v} \bmod p) && \text{(by Eqs. (5.4.5), (5.4.7) and (5.4.8))} \\
&= m && \text{(by Eq. (5.4.9) and (5.4.11))}
\end{aligned}$$

which leads to the left-hand side of Eq. (5.4.13).

If the authenticated ciphertext (s, r_2, R, T) is correctly generated, it will pass the test of Eq. (5.4.14). From the right-hand side of Eq. (5.4.14), we have

$$\begin{aligned}
& g^s y_p{}^{h_2(m,\, C\,,\, K,\, R)} C \\
&= g^{r+\sigma - x_p h_2(m,\, C,\, K,\, R)} y_p{}^{h_2(m,\, C,\, K,\, R)} C && \text{(by Eq. (5.4.10))} \\
&= Rg^{\sigma} C && \text{(by Eq. (5.4.6))} \\
&= RT \pmod p && \text{(by Eqs. (5.4.2), (5.4.5) and (5.4.8))}
\end{aligned}$$

which leads to the left-hand side of Eq. (5.4.14).

When the case of a later dispute over repudiation occurs, U_v can reveal the converted proxy multi-signature $\Omega = (s, R, T, K)$, the warrant m_W and the original message m to prove proxy signer's dishonesty without any additional computation effort or communication overhead. Thus, anyone can verify the converted proxy multi-signature with the assistance of Eqs. (5.4.4), (5.4.8) and (5.4.14).

Since the group-oriented variant is modified from our proposed PCAE-(II) scheme, we can adopt similar approaches to prove its security in random oracle models.

5.5 Variant with Message Linkages

Consider the practical implementation that an original message may be large. It therefore will cause the difficulty in encryption. In this subsection, we propose a variant with message linkages to benefit the encryption of a large message by dividing it into lots of small message blocks. The phases of Setup and PCG are defined the same as those in Section 5.1. We describe the other two phases as follows:

– **Authenticated-Ciphertext-Generation (ACG):** For signing a large message m on behalf of an original signer U_o, U_p first divides the message m into n pieces, i.e., $m = m_1 \parallel m_2 \parallel \ldots \parallel m_n$, m_i's $\in$ GF(p), and then chooses $r \in_R Z_q$ and $w_0 = 0$ to compute

$$R = g^r h_3(m) \bmod p, \tag{5.5.1}$$

$$K = y_v^{\sigma} \bmod p, \tag{5.5.2}$$

$$s = r - x_p h_2(m, C, R) \bmod q, \tag{5.5.3}$$

$$r_1 = s(K \bmod q) \bmod q, \tag{5.5.4}$$

$$w_i = m_i \cdot h_4(w_{i-1} \oplus h_4(K)) \bmod p, \text{ for } i = 1, 2, \ldots, n, \tag{5.5.5}$$

and then delivers the warrant m_w and the authenticated ciphertext $\delta = (r_1, R, T, w_1, w_2, \ldots, w_n)$ to a designated recipient U_v.

– Signature-Recovery-and-Verification (SRV): Upon receiving *δ*, U_v first computes

$$C = {y_o}^{h_1(m_w,\, T)} \bmod p, \tag{5.5.6}$$

$$K = (TC^{-1})^{x_v} \bmod p, \tag{5.5.7}$$

$$s = (K \bmod q)^{-1} r_1 \bmod q, \tag{5.5.8}$$

$$m_i = w_i \cdot h_4(w_{i-1} \oplus h_4(K))^{-1} \bmod p, \text{ for } i = 1, 2, \ldots, n, \tag{5.5.9}$$

and recovers the original message m as $m_1 \,||\, m_2 \,||\, \ldots \,||\, m_n$. U_v can further verify the proxy signature by checking if

$$R = g^s {y_p}^{h_2(m,\, C\,,\, R)} h_3(m) \bmod p. \tag{5.5.10}$$

When the case of a later dispute over repudiation occurs, U_v can reveal the converted proxy signature $\Omega = (R, s, T)$, the warrant m_w and the original message m to prove proxy signer's dishonesty without any additional computation effort or communication overhead. Thus, anyone can verify the converted proxy signature with the assistance of Eqs. (5.5.6) and (5.5.10).

We show that with the authenticated ciphertext $(r_1, R, T, w_1, w_2, \ldots, w_n)$ and the warrant m_w, a designated recipient U_v can recover the message m and check its validity with Eq. (5.5.9). From the right-hand side of Eq. (5.5.9), we have

$$\begin{aligned} & w_i \cdot h_4(r_{i-1} \oplus h_4(K))^{-1} \\ &= m_i \cdot h_4(r_{i-1} \oplus h_4(K)) \cdot h_4(r_{i-1} \oplus h_4(K))^{-1} && \text{(by Eq. (5.5.5))} \\ &= m_i \pmod p \end{aligned}$$

which leads to the left-hand side of Eq. (5.5.9).

Since the variant with message linkages is based on our proposed PCAE-(II) scheme, we can adopt the similar approaches to prove its security in random oracle models.

6. PCAE-(III) Scheme

Since Wu and Hsu [WH02] proposed the first convertible authenticated encryption (CAE) scheme in 2002, lots of researchers have devoted themselves to the enhancement of CAE schemes. Recently, a so-called bilinear pairings cryptosystem from elliptic curves [Kob87, Men93, Mil85] has been found various applications [BKL$^+$02, BF01, BLS01, GS02, Sma02, ZK02] in cryptography. In this section, we demonstrate the proposed third proxy CAE (abbreviated to PCAE-(III)) scheme based on BDHP.

6.1 Construction

– **Setup:** Taking as input 1^k, the system authority (SA) selects two groups $(\boldsymbol{G}_1, +)$ and $(\boldsymbol{G}_2, \times)$ of the same prime order q, where $|q| = k$. Let P be a generator of order q over $\boldsymbol{G}_1$, e: $\boldsymbol{G}_1 \times \boldsymbol{G}_1 \rightarrow \boldsymbol{G}_2$ a bilinear pairing and h_0: $\{0, 1\}^* \rightarrow \boldsymbol{G}_1$, h_1: $\boldsymbol{G}_2 \times \boldsymbol{G}_2 \rightarrow \{0, 1\}^k$, h_2: $\boldsymbol{G}_1 \rightarrow \boldsymbol{G}_1$ and h_3: $\{0, 1\}^k \times \boldsymbol{G}_2 \times \boldsymbol{G}_1 \rightarrow Z_q$ collision resistant hash functions. The system publishes public parameters $params = \{\boldsymbol{G}_1, \boldsymbol{G}_2, q, P, e, h_0, h_1, h_2, h_3\}$. Each user U_i chooses his private key $x_i \in Z_q$ and computes the corresponding public key as $Y_i = x_iP$.

– **Proxy-Credential-Generation (PCG):** Let U_o be an original signer delegating his signing power to a proxy signer U_p. U_o computes

$$D = x_o \cdot h_0(m_w), \tag{6.1.1}$$

where m_w is a warrant consisting of the identifiers of original signer, proxy signer and designated recipient, the delegation duration and so on. (D, m_w) is then sent to U_p. Upon receiving (D, m_w), U_p checks its validity by verifying whether

$$e(Y_o, h_0(m_w)) = e(D, P). \tag{6.1.2}$$

If it does not hold, (D, m_w) is requested to be sent again.

– **Authenticated-Ciphertext-Generation (ACG):** For signing a message $m \in_R \{0, 1\}^k$ on behalf of an original signer U_o, U_p chooses $r \in_R Z_q$ to compute

$$R = rP + D, \tag{6.1.3}$$

$$T = e(D, Y_v), \tag{6.1.4}$$

$$V = e(h_2(R), Y_v)^{x_p}, \tag{6.1.5}$$

$$S = r(h_3(m, T, R) + x_p)^{-1}P, \tag{6.1.6}$$

$$X = h_1(T, V) \oplus m, \tag{6.1.7}$$

and then delivers the warrant m_w and the authenticated ciphertext $\delta = (R, S, X)$ to a designated recipient U_v.

– **Signature-Recovery-and-Verification (SRV):** Upon receiving it, U_v first computes

$$T = e(Y_o, h_0(m_w))^{x_v}, \tag{6.1.8}$$

$$V = e(h_2(R), Y_p)^{x_v}, \tag{6.1.9}$$

to recover the message m as

$$m = h_1(T, V) \oplus X \tag{6.1.10}$$

and checks the redundancy embedded in it. U_v further verifies the proxy signature by checking whether

$$e(Y_o, h_0(m_w))e(S, h_3(m, T, R)P + Y_p) = e(R, P). \tag{6.1.11}$$

Since the converted proxy signature $\Omega = (R, S, T)$ is derived during the verification process, a designated recipient U_v can easily announce it together with (m, m_w) in case of a later dispute over repudiation. Accordingly, anyone can check Eq. (6.1.11) to realize proxy signer's dishonesty.

6.2 Correctness

We first show that the verification of Eq. (6.1.2) works correctly. From the left-hand side of Eq. (6.1.2), we have

$$e(Y_o, h_0(m_w))$$

$$= e(x_oP, h_0(m_w))$$

$$= e(x_oh_0(m_w), P)$$

$$= e(D, P) \qquad \text{(by Eq. (6.1.1))}$$

which leads to the right-hand side of Eq. (6.1.2).

Upon receiving $\delta = (R, S, X)$ with the warrant m_w, a designated recipient can correctly recover the message m and check its validity with Eq. (6.1.10). From the right-hand side of Eq. (6.1.10), we have

$$h_1(T, V) \oplus X$$

$$= h_1(e(Y_o, h_0(m_w))^{x_v}, e(h_2(R), Y_p)^{x_v}) \oplus X \qquad \text{(by Eqs. (6.1.8) and (6.1.9))}$$

$$= h_1(e(D, P)^{x_v}, e(h_2(R), Y_v)^{x_p}) \oplus X \qquad \text{(by Eq. (6.1.2))}$$

$$= m \qquad \text{(by Eqs. (6.1.4), (6.1.5) and (6.1.7))}$$

which leads to the left-hand side of Eq. (6.1.10).

If an authenticated ciphertext (R, S, X) is correctly generated, it will pass the test of Eq. (6.1.11). From the left-hand side of Eq. (6.1.11), we have

$$e(Y_o, h_0(m_w))e(S, h_3(m, T, R)P + Y_p)$$

$$= e(Y_o, h_0(m_w))e(r(h_3(m, T, R) + x_p)^{-1}P, h_3(m, T, R)P + Y_p) \qquad \text{(by Eq. (6.1.6))}$$

$$= e(D, P)e(rP, P) \qquad \text{(by Eq. (6.1.2))}$$

$$= e(D + rP, P)$$

$$= e(R, P) \qquad \text{(by Eq. (6.1.3))}$$

which leads to the right-hand side of Eq. (6.1.11).

6.3 Security Proofs

We prove that the proposed PCAE-(III) scheme achieves the IND-CCA2 and the EF-CMA security in random oracle models as Theorems 6.3.1 and 6.3.2, respectively.

Theorem 6.3.1. (Proof of Confidentiality) *The proposed PCAE-(III) scheme is* (t, q_{h_0}, q_{h_1}, q_{h_2}, q_{h_3}, q_{PCG}, q_{ACG}, q_{SRV}, ε)*-secure against indistinguishability under adaptive chosen-ciphertext attacks* (*IND-CCA2*) *in the random oracle model if there is no probabilistic polynomial-time adversary that can* (t', ε')*-break the BDHP, where*

$$\varepsilon' \geq (q_{h_1}{}^{-1})\left(2\varepsilon - \frac{q_{SRV}(q_{h_1} + q_{h_3} + 1)}{2^k}\right),$$

$$t' \approx t + t_\lambda(2q_{ACG} + 3q_{SRV} + 1).$$

Here t_λ *is the time for performing one bilinear pairing computation.*

Proof: Fig. 6.3.1 depicts the proof structure of this Theorem. Suppose that a probabilistic polynomial-time adversary $\mathcal{A}$ can (t, q_{h_0}, q_{h_1}, q_{h_2}, q_{h_3}, q_{PCG}, q_{ACG}, q_{SRV}, ε)-break the proposed PCAE-(III) scheme with a non-negligible advantage ε under adaptive chosen-ciphertext attacks after running in time at most t and asking at most q_{h_i} h_i random oracle (for i = 0 to 3), q_{PCG} PCG, q_{ACG} ACG and q_{SRV} SRV queries. Then we can construct another algorithm $\mathcal{B}$ that (t', ε')-breaks the BDHP by taking $\mathcal{A}$ as a subroutine. Let all involved parties and parameters be defined the same as those in Section 6.1. The objective of $\mathcal{B}$ is to obtain $e(P, P)^{x_o x_p x_v}$ by taking (P, q, e, $Y_o = x_oP$, $Y_p = x_pP$, $Y_v = x_vP$) as inputs. In this proof, $\mathcal{B}$ simulates a challenger to $\mathcal{A}$ in the following game.

Setup: The challenger $\mathcal{B}$ runs the Setup(1^k) algorithm and sends system's public parameters *params* = {$\mathbf{G}_1$, $\mathbf{G}_2$, q, P, e, Y_o, Y_p, Y_v} to the adversary $\mathcal{A}$.

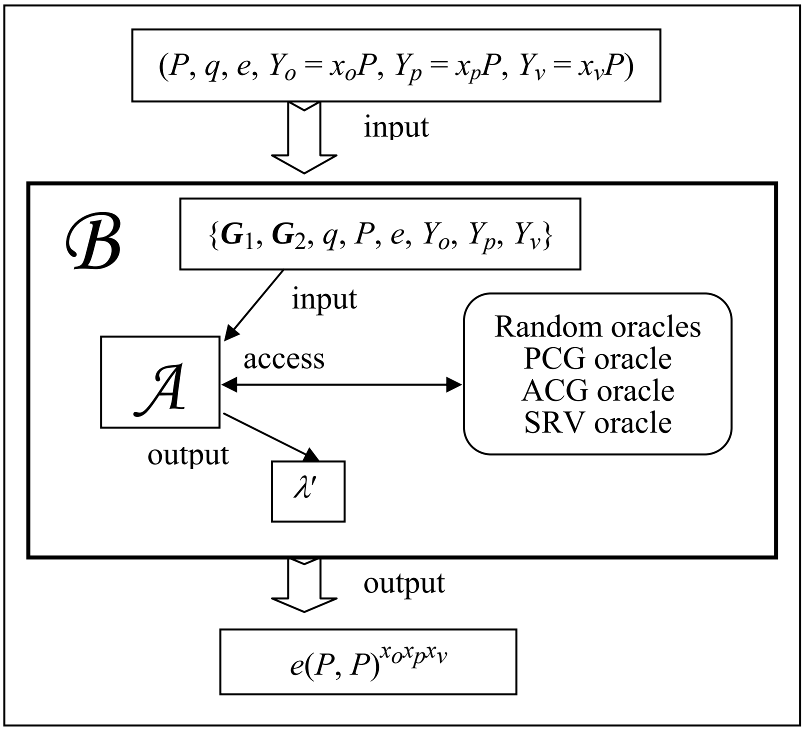


Fig. 6.3.1. The proof structure of confidentiality in Theorem 6.3.1

Phase 1: $\mathcal{A}$ issues the following queries adaptively:

– h_0 *oracle:* When $\mathcal{A}$ asks an h_0 oracle of $h_0(m_w)$, $\mathcal{B}$ returns $\mathcal{O}$**-Sim(III)_*h*₀**(m_w). The simulated random oracle $\mathcal{O}$**-Sim(III)_*h*₀** operates as Fig. 6.3.2.

```
oracle $\mathcal{O}$-Sim(III)_$h_0(m_w)$    // Let Q_$h_0[q_{h_0}]$ and A_$h_0[q_{h_0}][2]$ be two arrays.
  for $i = 0$ to $q_{h_0} - 1$
        if (Q_$h_0[i] = m_w$) then    // It is an old query.
              exit for;
        else if (Q_$h_0[i]$ = null) then    // It is a new query.
              insert(Q_$h_0$, $m_w$); insert(A_$h_0$, ($\sigma \in_R Z_q$, $V_0 = \sigma P$)); exit for;
        end if
  next $i$
  return A_$h_0[i][1]$;
```

Fig. 6.3.2. Algorithm of the simulated random oracle $\mathcal{O}$**-Sim(III)_*h*₀**

– h_1 *oracle:* When $\mathcal{A}$ asks an h_1 oracle of $h_1(T, V)$, $\mathcal{B}$ returns $\mathcal{O}$**-Sim_*h*₁**(T, V). The simulated random oracle $\mathcal{O}$**-Sim_*h*₁** operates as Fig. 6.3.3.

```
oracle O-Sim(III)_h1(T, V)   // Let Q_h1[$q_{h_1}$][2] and A_h1[$q_{h_1}$] be two arrays.
for $i = 0$ to $q_{h_1} - 1$
    if (Q_$h_1$[$i$][0] = $T$ and Q_$h_1$[$i$][1] = $V$) then   // It is an old query.
        exit for;
    else if (Q_$h_1$[$i$][0] = null) then   // It is a new query.
        insert(Q_$h_1$, ($T$, $V$)); insert(A_$h_1$, $v_1 \in_R \{0, 1\}^k$); exit for;
    end if
next $i$
return A_$h_1$[$i$];
```

Fig. 6.3.3. Algorithm of the simulated random oracle $\mathcal{O}$**-Sim(III)_*h*₁**

– h_2 *oracle:* When $\mathcal{A}$ asks an h_2 oracle of $h_2(R)$, $\mathcal{B}$ returns $\mathcal{O}$**-Sim(III)_*h*₂**(R). The simulated random oracle $\mathcal{O}$**-Sim(III)_*h*₂** operates as Fig. 6.3.4.

```
oracle O-Sim(III)_h2(R)   // Let Q_h2[$q_{h_2}$] and A_h2[$q_{h_2}$][2] be two arrays.
for $i = 0$ to $q_{h_2} - 1$
    if (Q_$h_2$[$i$] = $R$) then   // It is an old query.
        exit for;
    else if (Q_$h_2$[$i$] = null) then   // It is a new query.
        insert(Q_$h_2$, $R$); insert(A_$h_2$, ($v_2 \in_R Z_q$, $V_2 = v_2P$));
        exit for;
    end if
next $i$
return A_$h_2$[$i$][1];
```

Fig. 6.3.4. Algorithm of the simulated random oracle $\mathcal{O}$**-Sim(III)_*h*₂**

– h_3 *oracle:* When $\mathcal{A}$ asks an h_3 oracle of $h_3(m, T, R)$, $\mathcal{B}$ returns $\mathcal{O}$**-Sim(III)_*h*₃**(m, T, R). The simulated random oracle $\mathcal{O}$**-Sim(III)_*h*₃** operates as Fig. 6.3.5.

```
oracle 𝒪-Sim(III)_h3($m$, $T$, $R$)   // Let $Q\_h_3[q_{h_3}][3]$ and $A\_h_3[q_{h_3}]$ be two arrays.
 for $i = 0$ to $q_{h_3} - 1$
     if ($Q\_h_3[i][0] = m$) and ($Q\_h_3[i][1] = T$) and ($Q\_h_3[i][2] = R$) then
         exit for;   // It is an old query.
     else if ($Q\_h_3[i][0] =$ null) then   // It is a new query.
         insert($Q\_h_3$, ($m$, $T$, $R$));
         insert($A\_h_3$, $v_3 \in_R Z_q$);
         exit for;
     end if
 next $i$
return $A\_h_3[i]$;
```

Fig. 6.3.5. Algorithm of the simulated random oracle **𝒪-Sim(III)_*h*₃**

– *PCG queries:* When 𝒜 makes a PCG query, ℬ chooses a proper m_w and then returns (m_w, **𝒪-Sim(III)_*PCG***(m_w)) as the result. The simulated PCG oracle **𝒪-Sim(III)_*PCG*** operates as Fig. 6.3.6.

```
oracle 𝒪-Sim(III)_PCG($m_w$)
 $V_0 \leftarrow$ 𝒪-Sim_h0($m_w$);
 for $i = 0$ to $q_{h_0} - 1$
     if ($A\_h_0[i][1] = V_0$) then
         $\sigma \leftarrow A\_h_0[i][0]$;
         Compute $D = \sigma Y_o$;
         return $D$;
     end if
 next $i$
```

Fig. 6.3.6. Algorithm of the simulated PCG oracle **𝒪-Sim(III)_*PCG***

– *ACG queries:* When 𝒜 makes an ACG query for some message m, ℬ returns **𝒪-Sim(III)_*ACG***(m) as a result. The simulated ACG oracle **𝒪-Sim(III)_*ACG*** operates as Fig. 6.3.7.

```
oracle $\mathcal{O}$-Sim(III)_ACG($m$)
Choose a proper $m_w$; $D$ = $\mathcal{O}$-Sim(III)_PCG($m_w$);
Choose $s \in_R Z_q$ to compute $S = sP$ and $T = e(D, Y_v)$;
do
    Choose $v_3 \in_R Z_q$; Compute $R = sv_3P + sY_p + D$;
while (check(Q_$h_3$, ($m$, $T$, $R$)) = true);
insert(Q_$h_3$, ($m$, $T$, $R$)); insert(A_$h_3$, $v_3$);    // define $h_3(m, T, R) = v_3$
$V_2$ = $\mathcal{O}$-Sim_$h_2$($R$); $v_2$ = A_$h_2$[$i$][0];    // Assume that A_$h_2$[$i$][1] = $V_2$.
Compute $V = e(Y_p, Y_v)^{v_2}$; $X$ = $\mathcal{O}$-Sim_$h_1$($T$, $V$) $\oplus$ $m$;
return $\delta = (R, S, X)$ along with $m_w$;
```

Fig. 6.3.7. Algorithm of the simulated ACG oracle **$\mathcal{O}$-Sim(III)_*ACG***

– *SRV queries:* When $\mathcal{A}$ makes an SRV query for some authenticated ciphertext δ with a warrant m_w, $\mathcal{B}$ returns **$\mathcal{O}$-Sim(III)_*SRV***(δ, m_w) as the result. The simulated SRV oracle **$\mathcal{O}$-Sim(III)_*SRV*** operates as Fig. 6.3.8.

```
oracle $\mathcal{O}$-Sim(III)_SRV($\delta$, $m_w$)    // $\delta = (R, S, X)$
if (check(Q_$h_3$, (*, *, $R$)) = true) then    // $h_3$(*, *, $R$) has ever been queried.
    $m$ = Q_$h_2$[$i$][0]; $T$ = Q_$h_3$[$i$][1];    // Assume that Q_$h_3$[$i$][2] = $R$.
    if (check(Q_$h_1$, ($T$, *)) = true) then    // $h_1$($T$, *) has ever been queried.
        $V$ = Q_$h_1$[$i$][1]; $v_1$ = A_$h_1$[$i$]; // Assume that Q_$h_1$[$i$][0] = $T$.
        if ($X = v_1 \oplus m$) then
            if ($e(Y_o, h_0(m_w))e(S, h_3(m, T, R)P + Y_p) = e(R, P)$) then
                return ($m$, $R$, $S$, $T$, $m_w$);
            else    // Signature verification fails.
                return $\perp$;
            end if
        else    // Message recovery fails.
            return $\perp$;
        end if
    else    // $h_1$($T$, *) has never been queried.
        return $\perp$;
    end if
else    // $h_3$(*, *, $R$) has never been queried.
    return $\perp$;
end if
```

Fig. 6.3.8. Algorithm of the simulated SRV oracle **$\mathcal{O}$-Sim(III)_*SRV***

Challenge: $\mathcal{A}$ generates two messages, m_0 and m_1, of the same length. The challenger $\mathcal{B}$ flips a coin $\lambda \leftarrow \{0, 1\}$ and produces an authenticated ciphertext $\delta^* = (R^*, S^*, X^*)$ for m_λ by running the simulated **Sim(III)_Challenge**(m_λ). The algorithm of **Sim(III)_Challenge** operates as Fig. 6.3.9.

```
algorithm Sim(III)_Challenge($m_\lambda$)
  Choose a proper $m_w^*$; $D^* \leftarrow \mathcal{O}$-Sim(III)_PCG($m_w^*$);
  Choose $z, s \in_R Z_q$ to compute $S^* = sP$ and $T^* = e(D^*, Y_v)$;
  do
          Choose $v_3 \in_R Z_q$; Compute $R^* = sv_3P + sY_p + D^*$;
  while (check(Q_$h_3$, ($m_\lambda$, $T^*$, $R^*$)) = true);
  insert(Q_$h_3$, ($m_\lambda$, $T^*$, $R^*$)); insert(A_$h_3$, $v_3$);    // define $h_3(m_\lambda, T^*, R^*) = v_3$
  insert(Q_$h_2$, $R^*$); insert(A_$h_2$, ($\nabla$, $zY_o$)) where $\nabla$ denotes the null symbol;
    // define $h_2(R^*) = zY_o$
  Choose $v_1 \in_R \{0, 1\}^k$;
  Compute $X^* = v_1 \oplus m_\lambda$;    // Implicitly define $v_1 = h_1(T^*, V^*)$ where $V^* = e(zY_o, Y_v)^{xp}$.
  return $\delta^* = (R^*, S^*, X^*)$ and $m_w^*$;
```

Fig. 6.3.9. Algorithm of the simulated **Sim(III)_Challenge**

Phase 2: $\mathcal{A}$ makes new queries as those stated in Phase 1 except an SRV query for the target ciphertext δ^*.

Analysis of the game: Consider the simulations of PCG and ACG queries. It can be seen that the simulated proxy credential D and authenticated ciphertext δ are computationally indistinguishable from those generated by a real scheme. We refer the simulations of PCG and ACG queries to be perfect. Then we evaluate the simulation of SRV queries. From the algorithms of $\mathcal{O}$**-Sim(III)_*SRV***, one can observe that it is possible for an SRV query of some valid $\delta = (R, S, X)$ to return an error symbol $\perp$ on condition that $\mathcal{A}$ has the ability to produce δ without asking corresponding $h_3(m, T, R)$ or $h_1(T, V)$ random oracles in advance. Let SRV_ERR be an event that an SRV query returns an error symbol $\perp$ for some valid δ during the entire game, AC-V an event that an authenticated ciphertext δ submitted by $\mathcal{A}$ is valid. QH$_3$ and QH$_1$ separately denote the events that $\mathcal{A}$ has ever asked corresponding h_3 and h_1

random oracles beforehand. Then we can express the error probability of any SRV query as

$$\Pr[\text{AC-V} \mid \neg\text{QH}_3 \vee \neg\text{QH}_1]$$

$$= \frac{\Pr[\text{AC-V} \wedge (\neg\text{QH}_3 \vee \neg\text{QH}_1)]}{\Pr[\neg\text{QH}_3 \vee \neg\text{QH}_1]}$$

$$= \frac{\Pr[(\text{AC-V} \wedge \neg\text{QH}_3) \vee (\text{AC-V} \wedge \neg\text{QH}_1)]}{\Pr[\neg\text{QH}_3 \vee \neg\text{QH}_1]}$$

$$\leq \frac{\Pr[\text{AC-V} \wedge \neg\text{QH}_3]}{\Pr[\neg\text{QH}_3 \vee \neg\text{QH}_1]} + \frac{\Pr[\text{AC-V} \wedge \neg\text{QH}_1]}{\Pr[\neg\text{QH}_3 \vee \neg\text{QH}_1]}$$

$$= (\frac{\Pr[\text{AC-V} \wedge (\neg\text{QH}_3 \wedge \neg\text{QH}_1)]}{\Pr[\neg\text{QH}_3 \vee \neg\text{QH}_1]} + \frac{\Pr[\text{AC-V} \wedge (\neg\text{QH}_3 \wedge \text{QH}_1)]}{\Pr[\neg\text{QH}_3 \vee \neg\text{QH}_1]})$$

$$+ (\frac{\Pr[\text{AC-V} \wedge (\neg\text{QH}_1 \wedge \neg\text{QH}_3)]}{\Pr[\neg\text{QH}_3 \vee \neg\text{QH}_1]} + \frac{\Pr[\text{AC-V} \wedge (\neg\text{QH}_1 \wedge \text{QH}_3)]}{\Pr[\neg\text{QH}_3 \vee \neg\text{QH}_1]})$$

$$= \frac{\Pr[\text{AC-V} \wedge (\neg\text{QH}_3 \wedge \text{QH}_1)]}{\Pr[\neg\text{QH}_3 \vee \neg\text{QH}_1]} + \frac{2\Pr[\text{AC-V} \wedge (\neg\text{QH}_1 \wedge \neg\text{QH}_3)]}{\Pr[\neg\text{QH}_3 \vee \neg\text{QH}_1]}$$

$$+ \frac{\Pr[\text{AC-V} \wedge (\neg\text{QH}_1 \wedge \text{QH}_3)]}{\Pr[\neg\text{QH}_3 \vee \neg\text{QH}_1]}$$

$$\leq \frac{\Pr[(\text{AC-V} \wedge \text{QH}_1) \wedge \neg\text{QH}_3)]}{\Pr[\neg\text{QH}_3]} + \frac{2\Pr[\text{AC-V} \wedge (\neg\text{QH}_1 \wedge \neg\text{QH}_3)]}{\Pr[\neg\text{QH}_1 \wedge \neg\text{QH}_3]}$$

$$+ \frac{\Pr[(\text{AC-V} \wedge \text{QH}_3) \wedge \neg\text{QH}_1]}{\Pr[\neg\text{QH}_1]}$$

$$= \Pr[(\text{AC-V} \wedge \text{QH}_1) \mid \neg\text{QH}_3] + 2\Pr[\text{AC-V} \mid (\neg\text{QH}_1 \wedge \neg\text{QH}_3)]$$

$$+ \Pr[(\text{AC-V} \wedge \text{QH}_3) \mid \neg\text{QH}_1]$$

$$\leq \Pr[\text{QH}_1 \mid \neg\text{QH}_3] + 2\Pr[\text{AC-V} \mid (\neg\text{QH}_1 \wedge \neg\text{QH}_3)] + \Pr[\text{QH}_3 \mid \neg\text{QH}_1]$$

$$\leq \frac{q_{h_1}}{2^k} + \frac{1}{2^{2k-1}} + \frac{q_{h_3}}{2^k}$$

$$\leq \frac{q_{h_1}}{2^k} + \frac{1}{2^k} + \frac{q_{h_3}}{2^k}$$

$$= \frac{q_{h_1} + q_{h_3} + 1}{2^k}.$$

Since $\mathcal{A}$ can make at most q_{SRV} SRV queries, we can further express the probability of SRV_ERR as

$$\Pr[\text{SRV\_ERR}] \leq \frac{q_{SRV}(q_{h_1} + q_{h_3} + 1)}{2^k}. \tag{6.3.1}$$

Additionally, in the challenge phase, $\mathcal{B}$ has returned a simulated authenticated ciphertext $\delta^* = (R^*, S^*, X^*)$ where $h_2(R^*) = zY_o$, which implies the shared secret V^* is implicitly defined as $V^* = e(dY_o, Y_v)^{xp}$. Let GP be an event that the entire simulation game does not abort. Obviously, if the adversary $\mathcal{A}$ never makes an $h_1(T^*, V^*)$ query in Phase 2, the entire simulation game could be normally terminated. We denote an event that $\mathcal{A}$ does ask such a query in Phase 2 by QH$_1$*. When the entire simulation game does not abort, it can be seen $\mathcal{A}$ gains no advantage in guessing λ due to the randomness of output of random oracles, i.e.,

$$\Pr[\lambda' = \lambda \mid \text{GP}] = 1/2. \tag{6.3.2}$$

Rewriting the expression of $\Pr[\lambda' = \lambda]$, we have

$$\begin{aligned} \Pr[\lambda' = \lambda] &= \Pr[\lambda' = \lambda \mid \text{GP}]\Pr[\text{GP}] + \Pr[\lambda' = \lambda \mid \neg\text{GP}]\Pr[\neg\text{GP}] \\ &\leq (1/2)\Pr[\text{GP}] + \Pr[\neg\text{GP}] && \text{(by Eq. (6.3.2))} \\ &= (1/2)(1 - \Pr[\neg\text{GP}]) + \Pr[\neg\text{GP}] \\ &= (1/2) + (1/2)\Pr[\neg\text{GP}]. && (6.3.3) \end{aligned}$$

On the other hand, we can also derive that

$$\begin{aligned} \Pr[\lambda' = \lambda] &\geq \Pr[\lambda' = \lambda \mid \text{GP}]\Pr[\text{GP}] \\ &= (1/2)(1 - \Pr[\neg\text{GP}]) \\ &= (1/2) - (1/2)\Pr[\neg\text{GP}]. && (6.3.4) \end{aligned}$$

With inequalities (6.3.3) and (6.3.4), we know that

$$|\Pr[\lambda' = \lambda] - 1/2| \leq (1/2)\Pr[\neg\text{GP}]. \qquad (6.3.5)$$

Recall that in Definition 3.3.3, $\mathcal{A}$'*s* advantage is defined as $Adv(\mathcal{A}) = |\Pr[\lambda' = \lambda] - 1/2|$. By assumption, $\mathcal{A}$ has non-negligible probability ε to break the proposed scheme. We therefore have

$$\begin{aligned}\varepsilon &= |\Pr[\lambda' = \lambda] - 1/2| \\ &\leq (1/2)\Pr[\neg\text{GP}] && \text{(by Eq. (6.3.5))} \\ &= (1/2)(\Pr[\text{QH}_1{}^* \vee \text{SRV\_ERR}]) \\ &\leq (1/2)(\Pr[\text{QH}_1{}^*] + \Pr[\text{SRV\_ERR}])\end{aligned}$$

Combining Eq. (6.3.1) and rewriting the above inequality, we get

$$\begin{aligned}\Pr[\text{QH}_1{}^*] &\geq 2\varepsilon - \Pr[\text{SRV\_ERR}] \\ &\geq 2\varepsilon - \frac{q_{SRV}(q_{h_1} + q_{h_3} + 1)}{2^k}.\end{aligned}$$

If the event QH$_1$* happens, we claim that $V^* = e(zY_o, Y_v)^{xp}$ will be stored in some entry of Q_h_1 array. Consequently, $\mathcal{B}$ has non-negligible probability

$$\varepsilon' \geq (q_{h_1}{}^{-1})(2\varepsilon - \frac{q_{SRV}(q_{h_1} + q_{h_3} + 1)}{2^k})$$

to solve the BDHP by outputting $(V^*)^{z^{-1}}$. The computation time required for $\mathcal{B}$ is $t' \approx t + t_\lambda(2q_{ACG} + 3q_{SRV} + 1)$.

Q.E.D.

Theorem 6.3.2. (Proof of Unforgeability) *The proposed PCAE-(III) scheme is* (t, q_{h_0}, q_{h_1}, q_{h_2}, q_{h_3}, q_{PCG}, q_{ACG}, ε)*-secure against existential forgery under adaptive chosen-message attacks* (*EF-CMA*) *in the random oracle model if there is no probabilistic polynomial-time adversary* *that can* (t', ε')*-break the BDHP, where*

$$\varepsilon' \geq (\varepsilon - 2^{-(k + |\mathbf{G}_1|)})/(q_{h_1}q_{h_2}),$$

$$t' \approx t + t_\lambda(2q_{ACG}).$$

Here t_λ is the time for performing one bilinear pairing computation.

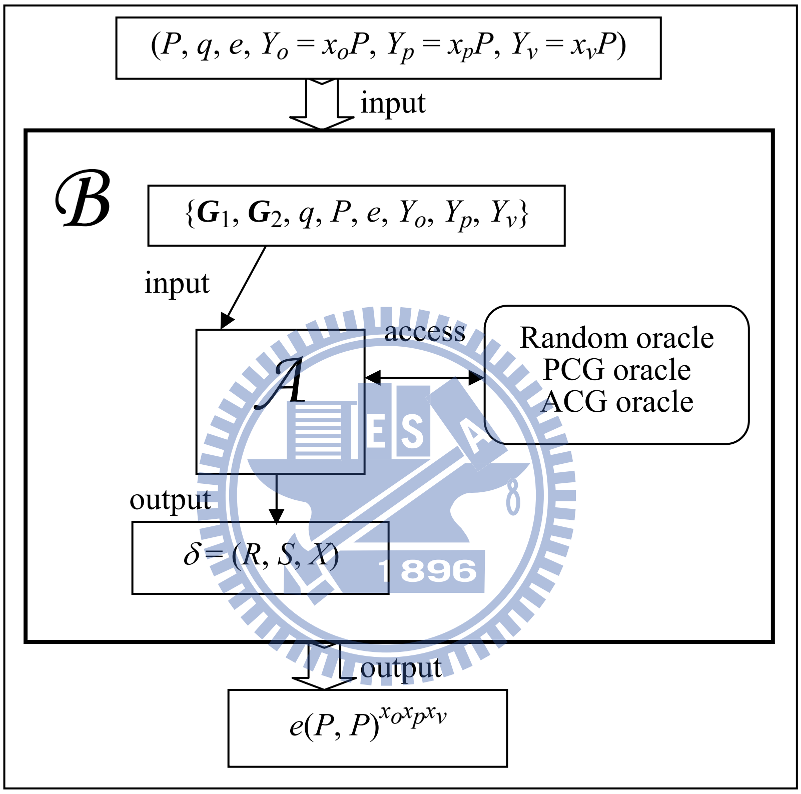


Fig. 6.3.10. The proof structure of unforgeability in Theorem 6.3.2

Proof: Fig. 6.3.10 depicts the proof structure of this Theorem. Suppose that a probabilistic polynomial-time adversary $\mathcal{A}$ can (t, q_{h_0}, q_{h_1}, q_{h_2}, q_{h_3}, q_{PCG}, q_{ACG}, ε)-break the proposed PCAE-(III) scheme with a non-negligible advantage ε under adaptive chosen-message attacks after running in time at most t and asking at most q_{h_i} h_i random oracle (for i = 0 to 3), q_{PCG} PCG and q_{ACG} ACG queries. Then we can construct another algorithm $\mathcal{B}$ that (t', ε')-breaks the BDHP by taking $\mathcal{A}$ as a subroutine. Let all involved parties and parameters be defined the same as those in Section 6.1. The objective of $\mathcal{B}$ is to obtain $e(P, P)^{x_ox_px_v}$ by taking (P, q, e, $Y_o = x_oP$, $Y_p = x_pP$, $Y_v = x_vP$) as inputs. In this proof, $\mathcal{B}$ simulates a challenger to $\mathcal{A}$ in the following game.

Setup: The challenger $\mathcal{B}$ runs the Setup(1^k) algorithm and sends system's public parameters *params* = {$\mathbf{G}_1$, $\mathbf{G}_2$, q, P, e, Y_o, Y_p, Y_v} to the adversary $\mathcal{A}$.

Phase 1: $\mathcal{A}$ adaptively asks h_i random oracle (for i = 0 to 3), PCG and ACG queries as those defined in Theorem 6.3.1. Note that in the j-th h_2 random oracle, where j is a random positive integer less than or equal to q_{h_2}, $\mathcal{B}$ directly returns zY_o for $z \in_R Z_q$.

Forgery: Finally, $\mathcal{A}$ outputs an authenticated ciphertext $\delta^* = (R^*, S^*, X^*)$ and m_w^* for his arbitrarily chosen message m^*. If the ciphertext is valid, $\mathcal{A}$ wins the game.

Analysis of the game: According to analyses of Theorem 6.3.1, we know that the simulation of each PCG or ACG query will be normally terminated. Besides, $\mathcal{B}$ answers each h_i random oracle with a computationally indistinguishable value without collision. Let AC-V and QH separately be the events that the outputted ciphertext $\delta^* = (R^*, S^*, X^*)$ is valid and $\mathcal{A}$ has ever asks corresponding $h_1(T^*, V^*)$ and $h_2(R^*)$ random oracles. The probability that $\mathcal{A}$ can guess correct random values without asking h_1 and h_2 random oracles is not greater than $2^{-(k + |\mathbf{G}_1|)}$. Since $\mathcal{A}$ has a non-negligible advantage ε to break the proposed scheme under adaptive chosen-message attacks, we have

$$\begin{aligned}\varepsilon &= \Pr[\text{AC-V}]\\ &\leq \Pr[\text{AC-V} \mid \text{QH}] + \Pr[\text{AC-V} \mid \neg\text{QH}]\\ &\leq \Pr[\text{AC-V} \mid \text{QH}] + 2^{-(k + |\mathbf{G}_1|)}.\end{aligned}$$

Further writing above inequality, we can also obtain

$$\Pr[\text{AC-V} \mid \text{QH}] \geq \varepsilon - 2^{-(k + |\mathbf{G}_1|)}.$$

Seeing that in the j-th h_2 random oracle, $\mathcal{B}$ directly returned zY_o as a result, i.e., $\Pr[R^* = R_j] = {q_{h_2}}^{-1}$, we claim that when the event (AC-V | QH) $\wedge$ ($R^* = R_j$) occurs, $\mathcal{B}$ would have the probability of (${q_{h_1}}^{-1}$) to output

$$(V^*)^{z^{-1}} = e(Y_o, Y_v)^{xp}$$

from some entry of Q_h_1 array. Therefore, we can express the probability of $\mathcal{B}$ to solve the BDHP problem as

$$\varepsilon' \geq (\varepsilon - 2^{-(k + |\mathbf{G}_1|)})/(q_{h_1} q_{h_2}).$$

The running time required for $\mathcal{B}$ is $t' \approx t + t_\lambda(2q_{ACG})$.

Q.E.D.

According to Theorem 6.3.2, the proposed PCAE-(III) scheme is secure against existential forgery attacks. That is, the delegated proxy signer cannot repudiate having generated his authenticated ciphertext. Hence, we obtain the following corollary.

Corollary 6.3.1. *The proposed PCAE-(III) scheme satisfies the security requirement of non-repudiation.*

7. Conclusions and Future Research

In this dissertation, the author proposed three PCAE schemes to solve the delegation problem for confidential transactions. The proposed schemes allow a proxy signer to produce an authenticated ciphertext on behalf of an original signer and only a designated recipient is capable of recovering the message and verifying its proxy signature for ensuring confidentiality.

It is unnecessary to establish a session key in advance between a proxy signer and a designated recipient. Without revealing the private key, a designated recipient can independently convert an authenticated ciphertext into an ordinary proxy signature for public arbitration in case of a later repudiation. Since a converted proxy signature is obtained during the message recovery and signature verification phase, the signature conversion process requires no extra computation efforts or communication overheads.

The author also presented a group-oriented PCAE variant allowing one proxy signer to generate a valid authenticated ciphertext on behalf of a signing group composed of *n* original signers. To benefit the encryption of large messages, the author addressed another variant with message linkages by dividing a large message into many small message blocks. Furthermore, the proposed schemes are proved to achieve the security requirement of confidentiality against indistinguishability under adaptive chosen-ciphertext attacks (IND-CCA2) and that of unforgeability against existential forgery under adaptive chosen-message attacks (EF-CMA) in random oracle models. To the best of our knowledge, the proposed PCAE-(I) scheme is the first provably secure PCAE scheme based on RSA assumption. As compared with previous works, the proposed schemes not only have lower computation costs, but also provide better functionalities.

With more and more complicated business applications, the signing policy and the way of proxy delegation might vary depending on different needs. For example, one original signer can delegate his signing power to two or more proxy signers, such that all proxy signers must cooperatively generate a valid authenticated ciphertext on behalf of the original one. In some circumstances, an authenticated ciphertext intended for a designated group can only be decrypted if *t*-out-of-*n* verifiers are willing to corporate, which is referred to as (*t*, *n*)-shared verification.

To mitigate the impact caused by the key exposure, a key-insulated cryptosystem is a better alternative for designing cryptographic protocols. In such a system, each user stores a long-term private key in a physically-secure but computation limited device (called base or helper). Another short-term private key is kept secret by the user and used to perform cryptographic protocols such as digital signature schemes. Integrating PCAE schemes with key-insulated systems will bring more benefits to realistic applicability. Therefore, in the future research, the author will devote himself to the study of more flexible PCAE schemes with provable security to fulfill all kinds of practical requirements.

Bibliography

[AUI99] S. Araki, S. Uehara and K. Imamura, “The limited verifier signature and its application,” *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, Vol. E82-A, No. 1, 1999, pp. 63-68.

[BD98] F. Bao and R. H. Deng, “A signcryption scheme with signature directly verifiable by public key,” *Workshop on Public Key Cryptography*, Springer-Verlag, 1998, pp. 55-59.

[BF01] D. Boneh and M. Franklin, “Identity-based encryption from the Weil pairing,” *Advances in Cryptology – CRYPTO 2001*, Springer-Verlag, 2001, pp. 213-229.

[BJY97] M. Bellare, M. Jakobsson and M. Yung, “Round-optimal zero-knowledge arguments based on any one-way hash function,” *Advances in Cryptology – EUROCRYPT’97*, Springer-Verlag, 1997, pp. 280-305.

[BKL$^+$02] P. S. L. M. Barreto, H. Y. Kim, B. Lynn and M. Scott, “Efficient algorithms for pairing-based cryptosystems,” *Advances in Cryptology – CRYPTO 2002*, Springer-Verlag, 2002, pp. 354-368.

[BLS01] D. Boneh, B. Lynn and H. Shacham, “Short signature from the Weil pairing,” *Advances in Cryptology – ASIACRYPT 2001*, Springer-Verlag, 2001, pp. 514-532.

[BLS03] P. S. L. M. Barreto, B. Lynn, and M. Scott, “On the selection of pairing-friendly groups,” *Selected Areas in Cryptography (SAC 2003)*, Springer-Verlag, 2003.

[Boy03] X. Boyen, “Multipurpose identity-based signcryption: A Swiss army knife for identity-based cryptography,” *Advances in Cryptology – CRYPTO’03*, Springer-Verlag, 2003, pp. 383-399.

[BSZ02] J. Baek, R. Steinfeld and Y. Zheng, “Formal proofs for the security of signcryption,” *Public Key Cryptography - PKC’02*, Springer-Verlag, 2002, pp. 80-98.

[CA90] D. Chaum and H. van Antwerpen, “Undeniable signature,” *Advances in Cryptology – CRYPTO’90*, Springer-Verlag, 1990, pp. 212-216.

[CC06] F. Cao and Z. Cao, "Cryptanalysis on a proxy multi-signature scheme," *Proceedings of the 1st International Multi-Symposiums on Computer and Computational Sciences, 2006 (IMSCCS'06)*, Vol. 2, IEEE Press, Piscataway, USA, 2006, pp. 117-120.

[Cha08] T. Y. Chang, "A convertible multi-authenticated encryption scheme for group communications," *Information Sciences*, Vol. 178, No. 17, 2008, pp. 3426-3434.

[Cha90] D. Chaum, "Zero-knowledge undeniable signatures," *Advances in Cryptology – EUROCRYPT'90*, Springer-Verlag, 1990, pp. 458-464.

[Chi08] H. Y. Chien, "Selectively convertible authenticated encryption in the random oracle model," *The Computer Journal*, Vol. 51, No. 4, 2008, pp. 419-434.

[DC06] S. Duan and Z. Cao, "Efficient and provably secure multi-receiver identity-based signcryption," *Information Security and Privacy*, Springer-Verlag, 2006, pp. 195-206.

[DCZ05] S. Duan, Z. Cao and Y. Zhou, "Secure delegation-by-warrant ID-based proxy signcryption scheme" *Proceedings of Computational Intelligence and Security Conference (CIS 2005)*, Springer-Verlag, 2005, pp. 445-450.

[DH76] W. Diffie and M. Hellman, "New directions in cryptography," *IEEE Transactions on Information Theory*, Vol. IT-22, No. 6, 1976, pp. 644-654.

[DK02] H. Delfs and H. Knebl, *Introduction to Cryptography: Principles and Applications*, 2nd Ed., Springer-Verlag, 2002.

[DYD03] J. Z. Dai, X. H. Yang and J. X. Dong, "Designated-receiver proxy signature scheme for electronic commerce," *Proceedings of IEEE International Conference on Systems, Man and Cybernetics*, Vol. 1, 2003, pp. 384-389.

[EA08] H. Elkamchouchi and Y. Abouelseoud, "A new proxy identity-based signcryption scheme for partial delegation of signing rights," *Cryptology ePrint Archive, Report 2008/041*, 2008. http://eprint.iacr.org/2008/041

[EAM06] D. H. Elkamshoushy, A. K. AbouAlsoud and M. Madkour, "New proxy signcryption scheme with DSA verifier," *Proceedings of the 23th National*

Radio Science Conference (NRSC 2006), 2006, pp. 1-8.

[ElG85] T. ElGamal, "A public key cryptosystem and a signature scheme based on discrete logarithms," *IEEE Transactions on Information Theory*, Vol. IT-31, No. 4, 1985, pp. 469-472.

[Gir91] M. Girault, "Self-certified public keys," *Advances in Cryptology – EUROCRYPT'91*, Springer-Verlag, 1991, pp. 491-497.

[GS02] C. Gentry and A. Silverberg, "Hierarchical id-based cryptography," *Advances in Cryptology – ASIACRYPT 2002*, Springer-Verlag, 2002, pp. 548-566.

[HC01] S. J. Hwang and C. C. Chen, "A new multi-proxy multisignature scheme," *Proceedings of 2001 National Computer Symposium*, 2001, pp. 19-26.

[HC03] H. F. Huang and C. C. Chang, "An efficient convertible authenticated encryption scheme and its variant," *Proceedings of the 5th International Conference on Information and Communications Security (ICICS2003)*, Springer-Verlag, 2003, pp. 382-392.

[Hen94] M. Hendry, *Smart Card Security and Applications*, Artech House, Inc., 1997.

[HLL00] M. S. Hwang, I. C. Lin, and J. L. Eric Lu, "A secure nonrepudiable threshold proxy signature scheme with known signers," *International Journal of Informatica*, Vol. 11, No. 2, 2000, pp. 1-8.

[HLL$^+$05] C. H. Huang, C. Y. Lee, C. H. Lin, C. C. Chang and K. L. Chen, "Authenticated encryption schemes with message linkage for threshold signatures," *Proceedings of the IEEE 19th International Conference on Advanced Information Networking and Applications*, Vol. 2, 2005, pp. 261-264.

[HLS05] R. J. Hwang, C. H. Lai and F. F. Su, "An efficient signcryption scheme with forward secrecy based on elliptic curve," *Applied Mathematics and Computation*, Vol. 167, No. 2, 2005, pp. 870-881.

[HMP94] P. Horster, M. Michel and H. Peterson, "Authenticated encryption schemes with low communication costs," *Electronics Letters*, Vol. 30, No. 15, 1994, pp. 1212-1213.

[HS00] S. J. Hwang and C. H. Shi, “A simple multi-proxy signature scheme,” *Proceedings of the 10th National Conference on Information Security*, 2000, pp. 134-138.

[HSM^{+}08] X. Huang, W. Susilo, Y. Mu and F. Zhang, “Short designated verifier signature scheme and its identity-based variant,” *International Journal of Network Security*, Vol. 6, No. 1, 2008, pp. 82-93.

[HW99] W. H. He and T. C. Wu, “Cryptanalysis and improvement of Petersen-Michels signcryption scheme,” *IEE Proceedings - Computers and Digital Techniques*, Vol. 146, No. 2, 1999, pp. 123-124.

[HWT^{+}04] F. Hou, Z. Wang, Y. Tang and Z. Liu, “Protecting integrity and confidentiality for data communication,” *Proceedings of the 9th International Symposium on Computers and Communications (ISCC)*, Vol. 1, No. 28, 2004, pp. 357-362.

[HWW01] C. L. Hsu, T. S. Wu and T. C. Wu, “New nonrepudiable threshold proxy signature scheme with known signers,” *The Journal of Systems and Software*, Vol. 58, No. 2, 2001, pp. 119-124.

[Jac91] J. Jacob, “A uniform presentation of confidentiality properties,” *IEEE Transactions on Software Engineering*, Vol. 17, No. 11, 1991, pp. 1186-1194.

[JS03] A. Juels and M. Szydlo, “A two-server, sealed-bid auction protocol,” *Financial Cryptography*, Vol. 2357, 2003, pp. 72-86.

[JSI96] M. Jakobsson, K. Sako and R. Impagliazzo, “Designated verifier proofs and their applications,” *Advances in Cryptology – EUROCRYPT’96*, Springer-Verlag, 1996, pp. 143-154.

[KBD09] B. Kang, C. Boyd and E. Dawson, “A novel identity-based strong designated verifier signature scheme,” *The Journal of Systems and Software*, Vol. 82, No. 2, 2009, pp. 270-273.

[Kob87] N. Koblitz, “Elliptic curve cryptosystems,” *Mathematics of Computation*, Vol. 48, No. 177, 1987, pp. 203-209.

[KPW97] S. Kim, S. Park and D. Won, “Proxy signatures, revisited,” *Proceedings of International Conference on Information and Communications Security*

(ICICS'97), Springer-Verlag, 1997, pp. 223-232.

[KSS06] K. Kumar, G. Shailaja and A. Saxena, "Identity based strong designated verifier signature scheme," *Cryptology ePrint Archive*, *Report 2006/134*, 2006. http://eprint.iacr.org/2006/134

[LC04] X. Li and K. Chen, "Identity based proxy-signcryption scheme from pairings," *Proceedings of the 2004 IEEE International Conference on Services Computing*, IEEE Computer Society, 2004, pp. 494-497.

[LC07] J. S. Lee and J. H. Chang, "Strong designated verifier signature scheme with message recovery," *Proceedings of the 9th International Conference on Advanced Communication Technology*, Vol. 1, 2007, pp. 801-803.

[LC09] J. S. Lee and J. H. Chang, "Comment on Saeednia *et al.*'s strong designated verifier signature scheme," *Computer Standards & Interfaces*, Vol. 31, No. 1, 2009, pp. 258-260.

[LHL$^+$01] Y. Li, B. Huang, W. Liu, H. Gou and C. Wu, "An electronic market architecture for virtual enterprises," *Proceedings of 2001 IEEE International Conference on Systems, Man, and Cybernetics*, Vol. 3, IEEE Press, Piscataway, USA, 2001, pp. 2028-2033.

[LHT09] C. C. Lee, M. S. Hwang and S. F. Tzeng, A new convertible authenticated encryption scheme based on the ElGamal cryptosystem, *International Journal of Foundations of Computer Science*, Vol. 20, No. 2, 2009, pp. 351-359.

[LHW07] R. Lu, D. He and C. Wang, "On the security of an identity-based threshold proxy signature scheme with known signers," *Proceedings of the 3rd International Conference on Natural Computation 2007 (ICNC'2007)*, Vol. 3, IEEE Press, Piscataway, USA, 2007, pp. 210-214.

[LHW98] N. Y. Lee, T. Hwang and C. H. Wang, "On Zhang's nonrepudiable proxy signature schemes," *Proceedings of the 3rd Australasian Conference on Information Security and Privacy (ACISP'98)*, Springer, Berlin, 1998, pp. 415-422.

[LW08] H. Y. Lin and T. S. Wu, "Bilinear pairings based convertible authenticated encryption scheme with provable recipient," *Proceedings of 2008 International Computer Symposium (ICS 2008)*, Taipei, Taiwan, November 2008.

[LW09] H. Y. Lin and T. S. Wu, "Pairings based designated verifier signature scheme for three-party communication environment," *Proceedings of 2009 International Conference on Computer Engineering and Technology (ICCET 2009)*, IEEE Computer Society, Singapore, January 2009, pp. 330-333.

[LWH02] C. Y. Lin, T. C. Wu and J. J. Hwang, "Multi-proxy signature schemes for partial delegation with cheater identification," *Proceedings of the 2nd International Workshop for Asia Public Key Infrastructure (IWAP 2002)*, IOS Press, Amsterdam, Netherlands, Technical Session E: Mobility & Certification, 2002.

[LWH$^+$07] H. Y. Lin, T. S. Wu, T. Y. Huang and T. C. Lin, "Group-oriented convertible authenticated encryption scheme with (t, n) shared verification," *Proceedings of TANET 2007*, Taipei, Taiwan, October 2007, pp. 502-506.

[LWH$^+$08] H. Y. Lin and T. S. Wu, T. Y. Huang and Y. S. Yeh, "Self-certified proxy convertible authenticated encryption scheme," *Proceedings of the 8th International Conference on Intelligent System Design and Applications (ISDA 2008)*, IEEE Press, Kaohsiung, Taiwan, November 2008.

[LWH10] H. Y. Lin and T. S. Wu, S. K. Huang and Y. S. Yeh, "Efficient proxy signcryption scheme with provable CCA and CMA security," *Computers and Mathematics with Applications*, Vol. 60, No. 7, 2010, pp. 1850-1858.

[LWK05] J. Lv, X. Wang and K. Kim, "Practical convertible authenticated encryption schemes using self-certified public keys," *Applied Mathematics and Computation*, Vol. 169, No. 2, 2005, pp. 1285-1297.

[LWY10] H. Y. Lin, T. S. Wu and Y. S. Yeh, "A DL based short strong designated verifier signature scheme with low computation," *Journal of Information Science and Engineering*, 2010. (to appear)

[LY08] H. Y. Lin and Y. S. Yeh, "A novel (t, n) threshold convertible authenticated encryption scheme," *Applied Mathematical Sciences*, Vol. 2, No. 5, 2008, pp. 249-254.

[Men93] A. Menezes, *Elliptic Curve Public Key Cryptosystems*, Kluwer Academic Publishers, 1993.

[Mil85] V. Miller, "Use of elliptic curves in cryptography," *Advances in Cryptology – CRYPTO'85*, Springer-Verlag, 1985, pp. 417-426.

[MOV97] A. Menezes, P. Oorschot and S. Vanstone, *Handbook of applied cryptography*, CRC Press, Inc, 1997.

[MUO96a] M. Mambo, K. Usuda and E. Okamoto, "Proxy signature for delegating signature operation," *Proceedings of the 3rd ACM Conference on Computer and Communications Security*, ACM press, 1996, pp. 48-57.

[MUO96b] M. Mambo, K. Usuda and E. Okamoto, "Proxy signatures: delegation of the power to sign messages," *IEICE Transactions on Fundamentals of Electronic Communications and Computer Science*, Vol. E79-A, No. 9, 1996, pp. 1338-1354.

[MWX02] B. Meng, S. Wang and Q. Xiong, "A fair non-repudiation protocol," *Proceedings of the 7th International Conference on Computer Supported Cooperative Work in Design (CSCW'02)*, Brazil, 2002, pp. 68-73.

[Neu93] B. C. Neuman, "Proxy-based authentication and accounting for distributed systems," *Proceedings of the 13th International Conference on Distributed Computing Systems*, 1993, pp. 283-291.

[NR93] K. Nyberg and R. A. Rueppel, "A new signature scheme based on the DSA given message recovery," *Proceedings of the 1st ACM Conference on Computer and Communications Security*, 1993, pp. 58-61.

[PM98] H. Petersen and M. Michels, "Cryptanalysis and improvement of signcryption schemes," *IEE Proceedings - Computers and Digital Techniques*, Vol. 145, No. 2, 1998, pp. 149-151.

[PS00] D. Pointcheval and J. Stern, "Security arguments for digital signatures and blind signatures," *Journal of Cryptology*, Vol. 13, 2000, pp. 361-369.

[RN01] I. Ray and N. Narasimhamurthi, "An anonymous electronic voting protocol for voting over the Internet," *Proceedings of the 3rd International Workshop on*

Advanced Issues of E-Commerce and Web-Based Information Systems (WECWIS'01), California, 2001, pp. 188-190.

[RRK^{+}04] S. Ravi, A. Raghunathan, P. Kocher and S. Hattangady, "Security in embedded systems: Design challenges," *ACM Transactions on Embedded Computing Systems (TECS)*, Vol. 3, No. 3, 2004, pp. 461-491.

[RSA78] R. Rivest, A. Shamir and L. Adleman, "A method for obtaining digital signatures and public-key cryptosystems," *Communications of the ACM*, Vol. 21, No. 2, 1978, pp. 120-126.

[Sch91] C. P. Schnorr, "Efficient signature generation by smart cards," *Journal of Cryptology*, Vol. 4, No. 3, 1991, pp. 161-174.

[Sch98] S. Schneider, "Formal analysis of a non-repudiation protocol," *Proceedings of 11th IEEE Computer Security Foundations Workshop*, IEEE Press, Piscataway, USA, 1998, p. 54-65.

[Sch99] B. Schoenmakers, "A simple publicly verifiable secret sharing scheme and its application to electronic voting," *Advances in Cryptology – CRYPTO'99*, Springer-Verlag, 1999, pp. 148-164.

[Sha84] A. Shamir, "Identity-based cryptosystems and signature schemes," *Advances in Cryptology – CRYPTO'84*, Springer-Verlag, 1984, pp. 47-53.

[SKM03] S. Saeednia, S. Kremer and O. Markowitch, "An efficient strong designated verifier signature scheme," *Proceedings of the 6th International Conference on Information Security and Cryptology (ICISC 2003)*, Berlin, 2003, pp. 40-54.

[Sma02] N. P. Smart, "Identity-based authenticated key agreement protocols based on Weil Pairings," *Electronics Letters*, Vol. 13, No. 38, 2002, pp. 630-632.

[SLH99] H. M. Sun, N. Y. Lee and T. Hwang, "Threshold proxy signatures," *IEE Proceedings of Computers & Digital Techniques*, Vol. 146, No, 5, 1999, pp. 259-263.

[SP02] K. M. Shelfer and J. D. Procaccino, "Smart card evolution," *Communications of the ACM*, Vol. 45, No. 7, 2002, pp. 83-88.

[Sta05] W. Stallings, *Cryptography and Network Security: Principles and Practice*, 4th. Ed., Pearson, 2005.

[Sta06] M. Stamp, *Information Security: Principles and Practice*, Wiley, 2006.

[SZM04] W. Susilo, F. Zhang and Y. Mu, “Identity-based strong designated verifier signature schemes,” *Information Security and Privacy*, Vol. 3108, Springer-Verlag, 2004, pp. 167-170.

[Tsa09] J. L. Tsai, “Convertible multi-authenticated encryption scheme with one-way hash function,” *Computer Communications*, Vol. 32, No. 5, 2009, pp. 783-786.

[TYH04] S. F. Tzeng, C. Y. Yang and M. S. Hwang, “A nonrepudiable threshold multi-proxy multisignature scheme with shared verification,” *Future Generation Computer Systems*, Vol. 20, No. 5, 2004, pp. 887-893.

[Var91] V. Varadharajan, P. Allen and S. Black, “An analysis of the proxy problem in distributed system,” *Proceedings of 1991 IEEE Computer Society Symposium on Research in Security and Privacy*, 1991, pp. 255-277.

[VM97] VISA and MasterCard Inc., *Secure Electronic Transaction (SET) Specification*, Version 1.0, 1997.

[Wan03] G. Wang, “An Attack on not-interactive designated verifier proofs for undeniable signatures,” *Cryptology ePrint Archive*, *Report 2003/243*, 2003. http://eprint.iacr.org/2003/243

[WC05] Q. Wang and Z. Cao, “Efficient ID-based proxy signature and proxy signcryption from bilinear pairings,” *Computational Intelligence and Security*, Vol. 3802, Springer-Verlag, 2005, pp. 167-172.

[WL05] M. Wang and Z. Liu, “Identity based threshold proxy signcryption scheme,” *Proceedings of the 15th International Conference on Computer and Information Technology (CIT 2005)*, 2005, pp. 695-699.

[WCL08] C. C. Wu, C. C. Chang and I. C. Lin, “New sealed-bid electronic auction with fairness, security and efficiency,” *Journal of Computer Science and Technology*, Vol. 23, No. 2, 2008, pp. 253-264.

[WCL^{+}07] L. Wang, Z. Cao, X. Li and H. Qian, “Simulatability and security of certificateless threshold signatures,” *Information Sciences*, Vol. 177, No. 6, 2007, pp. 1382-1394.

[WH02] T. S. Wu and C. L. Hsu, “Convertible authenticated encryption scheme,” *The Journal of Systems and Software*, Vol. 62, No. 3, 2002, pp. 205-209.

[WHL05] T. S. Wu, C. L. Hsu and H. Y. Lin, “Efficient convertible authenticated encryption schemes for smart card applications in network environments,” *Proceedings of the 9th World Multi-Conference on Systemics, Cybernetics and Informatics (WMSCI2005)*, Orlando, Florida, U.S.A., July 2005.

[WHL08] T. S. Wu, C. L. Hsu and H. Y. Lin, “Self-certified multi-proxy signature schemes with message recovery,” *Journal of Zhejiang University-SCIENCE A*, Vol. 10, No. 2, 2009, pp. 290-300.

[WHT^{+}08] T. S. Wu, C. L. Hsu, K. Y. Tsai, H. Y. Lin and T. C. Wu, “Convertible multi-authenticated encryption scheme,” *Information Sciences*, Vol. 178, No. 1, 2008, pp. 256-263.

[WL05] M. Wang and Z. Liu, “Identity based threshold proxy signcryption scheme,” *Proceedings of the 15th International Conference on Computer and Information Technology (CIT 2005)*, 2005, pp. 695-699.

[WL08a] T. S. Wu and H. Y. Lin, “A group-oriented proxy CMAE scheme with computational secrecy,” *International Journal of Innovative Computing, Information and Control*, Vol. 4, No. 11, 2008, pp. 3037-3047.

[WL08b] T. S. Wu and H. Y. Lin, “ECC based convertible authenticated encryption scheme using self-certified public key systems,” *International Journal of Algebra*, Vol. 2, No. 3, 2008, pp. 109-117.

[WL09] T. S. Wu and H. Y. Lin, “Secure convertible authenticated encryption scheme based on RSA,” *Informatica*, Vol. 33, No. 4, 2009, pp. 481-486.

[WLC06] T. S. Wu, H. Y. Lin and W. Y. Chang, “Improved threshold authenticated encryption scheme based on the factorization problem,” *Proceedings of 2006 International Conference of Digital Technology and Innovation Management,* Taipei, Taiwan, April 2006, pp. 1699-1709.

[WLH+07] T. S. Wu, H. Y. Lin, C. C. Hu and M. L. Lee, "ECC based self-certified convertible authenticated encryption scheme with computational secrecy," *Proceedings of 2007 National Computer Symposium (NCS 2007)*, Taichung, Taiwan, December 2007, pp. 789-795.

[XC04a] Q. Xue and Z. Cao, "A nonrepudiable multi-proxy multisignature scheme," *Proceedings of the 1st Joint Workshop on Mobile Future & Symposium on Trends in Communications (SympoTIC '04)*, IEEE Press, Piscataway, USA, 2004, pp. 102-105.

[XC04b] Q. Xue and Z. Cao, "Improvement of multi-proxy signature scheme," *Proceedings of the 4th International Conference on Computer and Information Technology (CIT'04)*, IEEE Press, Piscataway, USA, 2004, pp. 450-455.

[Yan05] F. Y. Yang, "A secure scheme for authenticated encryption," *Cryptology ePrint Archive*, *Report 2005/456*, 2005. http://eprint.iacr.org/2005/456

[YL10] F. Y. Yang and C. M. Liao, "A provably secure and efficient strong designated verifier signature scheme," *International Journal of Network Security*, Vol. 10, No. 3, 2010, pp. 223-227.

[YX00] L. B. Yi and G. Xiao, "Proxy multisignature scheme: a new type of proxy signature scheme," *Electronics Letters*, Vol. 36, No. 60, 2000, pp. 527-528.

[ZD04] Z. Zhang and Q. Dong, "A new publicly verifiable proxy signcryption scheme," *The International Series in Engineering and Computer Science – Progress on Cryptography*, Vol. 769, Springer-Verlag, 2004, pp. 53-57.

[Zhe97] Y. Zheng, "Digital signcryption or how to achieve cost(signature & encryption) << cost(signature) + cost(encryption)," *Advances in Cryptology – CRYPTO'97*, Springer-Verlag, 1997, pp. 165-179.

[ZK02] F. Zhang, and K. Kim, "ID-based blind signature and ring signature from pairings," *Advances in Cryptology – ASIACRYPT 2002*, Springer-Verlag, 2002, pp. 533-547.

[ZK03] F. Zhang and K. Kim, "A universal forgery on Araki *et al.*'s convertible limited verifier signature scheme," *IEICE Transactions on Fundamentals*, Vol. E86-A, No. 2, 2003, pp. 515-516.

[ZM08] J. Zhang and J. Mao, "A novel ID-based designated verifier signature scheme," *Information Sciences*, Vol. 178, No. 3, 2008, pp. 766-773.