Design and Analysis of Practical Digital Signature Schemes

by

Takeshi OKAMOTO

submitted to
Japan Advanced Institute of Science and Technology
in partial fulfillment of the requirements
for the degree of
Doctor of Philosophy

Supervisor: Associate Professor Atsuko Miyaji

School of Information Science
Japan Advanced Institute of Science and Technology

February 15, 2002

Abstract

The digital signature is a technology which guarantees the validity of the data together with the signer. In proportion as the spread of personal computers and open networks, the signature scheme becomes even more important than ever. More efficient and functional signature schemes are required.

In this thesis, we focus on two kinds of digital signatures, that is, fast on-line signatures and proxy signatures. "On-line" means the phase of real-time requirement for signing message. The dominant factor of on-line phase usually consists of some modular arithmetics such as multiplication and modular reduction. A signature scheme without modular reduction is called "on the fly" signature. On the other hand, a proxy signature means the scheme, which allows a designated person to sign on behalf of an original signer.

Our contributions are summarized as follows:

1. We research new mathematical problems, named self-powering RSA problem and extended finding order problem. Those problems are used as the underlying problem in our schemes;

2. We propose a new on the fly signature by improving the Poupard-Stern scheme (PS-scheme). Our scheme has the efficiency in terms of both amount of work and transmitted data size;

3. We design and analysis an new fast on-line signature, which has new feature in that on-line multiplication is not required;

4. We construct new proxy signature schemes, which are based on the original signer's message recovery. We also propose two practical schemes, whose security is based on Discrete logarithm and RSA problems, respectively.

We investigate the above themes in terms of both security and efficiency. Our schemes satisfy the provable security using well-known or our proposed assumptions. All our schemes are enough practical from both computational and transmitted-data point of view.

Acknowledgments

First of all, I am most grateful to my supervisor, Associate Professor Atsuko Miyaji, who gave me the opportunity to study in cryptography and supported me. I am grateful to thank Professor Mineo Kaneko, Professor Tetsuo Asano, Associate Professor Kunihiko Hiraishi and Dr. Tatsuaki Okamoto of NTT Laboratory for their much advice and encouragement. I would like to thank Professor Hiroakira Ono for helpful comments and nice discussions with him.

I wish to express my gratitude to Professor Eiji Okamoto of Toho University, Hiroki Shizuya of Tohoku University, Associate Professor Masahiro Mambo of Tohoku University and Professor Masahiro Kubota of Kyoto Institute of Technology for their precious advice and helpful support.

I greatly thank Associate Mitsuru Tada and Associate Masakazu Soshi for their eager suggestions and helpful comments. I deeply thank Dr.Shigeki Kitazawa of Mitsubishi Electric Corporation, Dr. Hisao Sakazaki of Hitachi Corporation, Dr. Kazumasa Omote, Ms. Yuko Tamura and Mr. Kei Kawauchi for many comments and helpful teaching. They guided me into the fascinating field of cryptography and the basis for this thesis lies in the joint work with them.

I really appreciate to take this opportunity and thank all the people who supported my research and gave me good suggestions. I would like to thank all the members in Miyaji Laboratory and former Okamoto Laboratory for their helpful comments, nice talks and much advice. In the laboratories, I always had a great and relaxing atmosphere.

Finally, I sincerely thank my parents, Hideo Okamoto, Tomiko Okamoto, my brothers, Tsuyoshi Okamoto, Satoshi Okamoto and my sister, Megumi Okamoto for their support, advice and encouragement.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Preface

Cryptography is a strategy of information protection. Through the ages, cryptography has protected communications through hostile environments, especially in the case of war or diplomacy. One can see that an innovative progress was made in such situations. For example, the world's first digital computers were built to crack the codes in World War II. In 1949, the publication by C. E. Shannon, "Communication Theory of Secret Systems" [49], opened up a new field of cryptography. He built on the *information theory* and showed for instance that "perfect secrecy" (theoretically secure ciphertext) requires a one-time secret key which is as long as the plain text. In 1977, Data Encryption Standard (DES) [31] was published by National Bureau of Standards. The idea of a "open algorithm" is a revolutionary event in cryptography. Before the publication of DES, there apparently were no publications containing a complete algorithm for practical cryptographic usage.

The remarkable breakthrough of the cryptography came with the publication in 1976 by W.Diffie and M.E.Hellman of their work "New Directions in Cryptography" [9]. In the paper, they proposed the concept of public key cryptography and showed that secret communication is possible without an exchange of secret key in advance, while usual symmetric cryptosystem was required for such preparations. Their splendid idea was to use two different keys, a public key for encryption and a private key for decryption. Based on this asymmetry, they further proposed the concept of digital signatures. Here, the private key is used to sign a message and the public key is used to verify a signature. However, they did not provide realizations of the new concepts, but they proposed a protocol that allows two entities to share a common secret key only by exchanging information in public.

The concept of public key cryptography inspired many researchers, and it soon became a fast-growing and fascinating research theme. In the following years, although many realization of public key encryption and digital signature schemes were proposed, most notable one was RSA scheme. This scheme was introduced by three inventors R.L.Rivest, A.Shamir and L.Adleman who published the paper "A method for obtaining digital sig-

natures and public key cryptosystems" [42] in 1978. This scheme was the first practical public-key encryption and digital signature schemes. Based on these primitives, more complex systems such as digital payment schemes or voting schemes were devised.

Nowadays, public-key cryptography and in particular digital signatures will play a key role in the emerging information society. Existing applications of such a cryptography include the authentication and encryption of bilateral communication (e.g., e-mail, internet banking, on-line shopping), electronic voting, and access control. Digital signatures can serve as a mechanism for making legally binding statements (e.g. contracts, testimonies.)

The goal of this thesis is propose now notions and techniques leading to digital signature systems that can be applied in practice. It is our belief that we provide *elegant* schemes with both security and efficiency. The security is an indispensable property for any puclic-key cryptosystem as well as the efficiency.

We now procced in three basic directions toward this ideal and consider how to efficiently extend digital signature with the best security. Those three themes, are described as follows.

1. Number theoretic problem

The security of public key cryptosystems relies on a number theoretic problem which is considered as a difficult problem. *Difficult problem* means the computational requirement in finding a solution. Those problems are called *hard problems*.

In a cryptographic setting, a number theoretic problem is regarded as a hard problem and suppose that such a problem is hard to solve. Informally, this means that any adversary cannot solve the problem in reasonable time. Concrete to say, a number theoretic problem is said to be easy if it can be solved for all possible inputs with a non-negligible fraction in (expected) polynomial time. In other words, if there is an algorithm which can solve a problem for all instances of a problem with non-negligible fraction in polynomial time, then any cryptosystem, whose security is based on that problem, must be considered insecure.

Nowadays, RSA problem [42], integer factoring problem, discrete logarithm problem etc., are well used in practical public-key cryptosystems. Among those problems, RSA problem is the most widely used in the real world. However, the true computational complexities of those hard problems has not been known. For example, the most efficient algorithms ever proposed to solve the integer factoring problem is the number field sieve [25] and it takes sub exponential time. They are widely believed to be intractable, although any proof is not known.

Although the only lower bounds to solve these problems are the trivial linear bounds generally, one can not provide any evidence of their intractability. Consequently, it is important to study their relative difficulties. For this reason, various techniques reducing one computational problem to another one have been proposed and studied very well. These reductions provide the techniques for converting algorithms that solves the second problem into an algorithm for solving the first problem.

2. Fast on-line signature

Up to the present, the computational performance of public-key cryptosystem is considerably inferior to that of symmetric-key cryptosystem. Therefore, one of the most important things for public-key cryptosystems is to study the computational efficiency. The most critical computation is the signature generations because the signature verification is done in rather high-performance machine. We now focus on the signer's computational work. It consists of two kinds of computation for the signer: *pre-computation* and *(actual) signature generation*. The pre-computation can be computed during idle hours because it is completely independent of the message to be signed. The computational amount in the pre-computation part is not critical, is called *off-line* processing.

On the other hand, the (actual) signature generation which does directly influence the processing time, is called *on-line* processing. To estimate the efficiency of a signature scheme, we separately consider the two types of computation. The fast on-line signature is required various applications. The pre-computation usually consists of hasing, addition, multiplication and modular reduction. Since the computational amount of hasing or addition is much smaller than that of multiplication or modular reduction, we usually discuss only multiplication and modular reduction. A signature scheme without on-line modular reduction, which is proposed by Poupard and Stern in [38], is called *on the fly signature*.

We now focus on the computational efficiency of both the algorithm for multiplication and the algorithm for modular reduction. A recursive algorithm due to Karatsuba and Ofman [20] reduce the complexity of the multiplication. On the other hand, Barrett reduction [3] reduce the complexity of the modular reduction and it is much advantageous if reduction is performed with a *single modulus*. As a result, the Barrett reduction is further advantageous than the Karatsuba-Ofman algorithm if the *single modulus* can be used in the case of the modular reduction. From this reason, we are longing for a signature scheme such that the actual on-line computation is only modular reduction and the signer can compute a value with a single modulus.

3. Proxy signature

Proxy signatures, introduced by Mambo, Usuda and Okamoto in [27], allow a designated person to sign on behalf of an original signer. Suppose that an employee in a company needs to go on a business trip without special tool available for access to the company. She has instructed her secretary to respond some important e-mail on behalf of her. Proxy signature gives a solution for such a situation, which has the following properties:

- **Unforgeability.** Only the original signer and the designated proxy signer can create a valid proxy signature;
- **Proxy signer's deviation.** A proxy signer cannot impersonate the original signature;

- **Verifiability.** From a proxy signature, a verifier can be convinced of the original signer's agreement on the signed message.
- **Distinguishability.** Proxy signatures are distinguishable from normal signatures by anyone;
- **Identifiability.** An original signer can determine from a proxy signer the identity of the corresponding proxy signer;
- **Undeniability.** A proxy signer cannot repudiate her generated signature.

In some cases, a stronger notion of identifiability in which anyone can determine the proxy signer's identity from the proxy, is required signature.

Our contribution

In this thesis, we study the following themes:

1. We research new mathematical problems named *self-powering RSA problem* and *extended finding order problem*. Those problems are used as underlying problem in our schemes. To discuss the former problem, we propose a modified Guillou-Quisquarter identity-based scheme whose security is based on the problem. We show that Modified GQ scheme is more efficient than previous one because it is not necessary to use a hash function in our scheme. We also show that our fast on-line schemes (described below) as secure as the latter problem, if we set a public key (modulus) n which n consists of three or more prime factors.

2. We propose a new on the fly signature by improving the PS-scheme. In PS-scheme, the size of secret key is fixed by modulus n, so that this feature lead to some drawbacks in terms of both the computational work and the communication load. The main idea of our scheme is to reduce the size of secret key in PS-scheme by using a public element g which has a specific structure. Consequently, our schemes are improved with respect to the computational work (which means the computational cost for "pre-computation", "(on-line) signature generation" and "verification") and the data size such as a secret-key and a signature.

3. We design and analysis an new fast on-line signature. One may consider that our scheme is a counterpart against on the fly signatures. Our scheme would require a modular exponentiation as preprocessing. However, no multiplication is used in the actual (i.e. on-line) signature generation. This means that the phase involves only a hasing operation, addition and a modular reduction. Hence, our approach to obtain the fast on-line signature is different from theirs. Our study is the first approach for the fast signature scheme without on-line modular multiplication.

4. We construct a new notion for proxy signature scheme. The remarkable property in our scheme is that in verification, the verifier accept the signature if and only if the verifier recovers the message created by the original signer. In our thesis, we propose new proxy signatures, DLP-PS and RSA-PS. Both schemes have the property that the original signer can control signing power of proxy signer to some extent. DLP-PS is obtained by improving the basic proxy signature and controls the power of proxy signer by adding *usage condition* to proxy signer's public key implicitly, which uses idea of message recovery. In verifying proxy signers signature, the usage condition can be checked independent of a message. RSA-PS controls the power of proxy signer by adding usage condition to proxy signer's public key explicitly, which uses the idea of ID-based cryptosystem. The efficiency of our proposed schemes are better than that of previous one with respect to both computational time and transmitted data size.

Thesis Outline

This thesis is organized as follows. Chapter 2 gives a general cryptographic background, which is used for the understanding of basic concepts used in later chapters. Chapter 3 presents the fundamental definitions and models with respect to three-pass identification scheme. The current schemes are also presented. Chapter 4 formally models the digital signature scheme and present current schemes. Chapter 5 gives a discussion of proposed mathematical problems. It also proposes a new identification scheme which is based on one of those problems. Chapter 6 deals with the fast on-line signature. In this chapter, two kinds of schemes are proposed. One of them is a on the fly signature which is obtained by improving PS-scheme. Another scheme is a signature without on-line multiplication. Chapter 7 propose a new system of the proxy signature, and show the two schemes which are based on the discrete logarithm and RSA problems, respectively. Chapter 8 concludes this thesis with remarks.

1.2 Preliminary Definition

We first define symbols, notations and definitions. They are used throughout our thesis.

- $\mathbb{Z}$ denotes a set of all integers.
- $\mathbb{Z}_p$ denotes a set of integers more than p.
- $\mathbb{Z}_p^*$ denotes a multiplicative group of $\mathbb{Z}_p$, that is $\mathbb{Z}_p^* = \{x \in \mathbb{Z}_p | \gcd(x, p) = 1\}$.
- $\mathbb{N}$ denotes a set of all natural numbers.
- $\mathbb{N}_{>i}$ denotes a set of natural integers more than i.

- $\mathbb{N}_{\text{prime}}$ denotes a set of all primes.
- $\text{Ord}_n(g)$ denotes the order of an element $g \in \mathbb{Z}_n^*$.
- $a|b$ means a divides b equivalently, that is, a is a divisor of b, or a is a factor of b.
- $|x|$ means the number of bits of x, that is, $|x|$ is the size of x.
- $\|x\|$ means the absolute value of x.
- $\gcd(a, b)$ means greatest common divisor of integer a and b.
- $\ln x$ is the natural logarithm of x, that is, the logarithm of x to the base base of natural logarithm, e.
- $\log x$ is the logarithm of x to the base 2.
- $\varphi(\cdot)$ denotes Euler totinent function, that is, $\varphi(n)$ is the greatest number among the possible orders of elements in $\mathbb{Z}_n^*$.
- $\lambda(\cdot)$ denotes so called Carmichael function, that is, $\lambda(n)$ is the greatest number among the possible orders of elements in $\mathbb{Z}_n^*$.

On-line computing	This is the (prover's/signer's) computational work which can be computed after (random challenge/message to be signed) is determined.
Off-line computing	This is the (prover's/signer's) computational work which can be computed without (random challenge/message to be signed).
Fast on-line signature	This is the signature which has the on-line computational efficiency.
Strong prime	We say that a prime p is a strong prime if $p = 2p' + 1$ and p' is also prime.
Generic (digital) signature	This means a signature scheme which can be derived from a three-pass identification scheme by using an appropriate hash function.
RSA modulus	We say that modulus n is RSA modulus if n is a product of two distinct prime, that is, $n = pq$ for primes $p \neq q$.
TA	This is the abbreviation for Trusted Authority.
PPT	This is the abbreviation for the probabilistic polynomial-time Turing.

Definition 1.2.1 ((Big-O/small-o) notation) Let $f(k) > 0$ and $g(k) > 0$ be two functions. If

- $(\exists c > 0, \exists u, \forall x > u)[f(x)/g(x) < c]$,
- $(\forall c > 0, \exists u, \forall x > u)[f(x)/g(x) < c]$,

hold, we write as $f(x) = O(g(x))$ and $f(x) = o(g(x))$, respectively. ■

Definition 1.2.2 ((Polynomial/sub exponential/exponential) time algorithm)

- **Polynomial time algorithm.** It is an algorithm whose running time is of the form $O(n^c)$, where n is the input size and c is a constant number.
- **Sub exponential time algorithm.** It is an algorithm worse running time is of the form $e^{o(n)}$, where e is the natural logarithm and n is the input size.
- **Exponential time algorithm.** It is an algorithm worse running time is of the form $O(c^n)$, where n is the input size and $c > 1$ is a constant number.

■

Definition 1.2.3 (Negligible/non-negligible/overwhelming) Let $f(k) > 0$ be a function. If

- $(\forall c > 0)[f(x) = o(1/x^c)]$,
- $1/x^c = O(f(x))$,
- $1 - f(x) = o(1/x^c)$,

hold, we say f is negligible, non-negligible and overwhelming, respectively. Additionally,

- $f(k) < \epsilon(k)$ means that a function $f(k)$ is negligible for k,
- $f(k) > \epsilon(k)$ means that a function $f(k)$ is not negligible for k,
- $f(k) > 1 - \epsilon(k)$ means that a function $f(k)$ is overwhelmingly for k.

■

In this thesis, we use the term "negligible", "non-negligible" or "overwhelming" for the purpose of security parameter. Hence we usually omit the description "for security parameter" except for the special case.

Chapter 2

Cryptographic Background

2.1 Introduction to Public Key Cryptosystems

Public-key cryptography provides a radical departure from all that has gone before. For one thing, public-key algorithms are based on mathematical functions rather than on substitution and permutation. But more important, public-key cryptography is asymmetric, involving the use of two separate keys, in contrast to the symmetric conventional encryption, which uses only one key.

The idea behind a "*public-key*" system is that it might be possible to find a cryptosystem which it is computationally infeasible to determine the decryption rule D_K even by using the encryption rule E_K. If so, then the E_K could be made public by publishing it in a directory (hence we call this "*public-key*" system). the advantage of a public-key system is that Alice (or anyone else) can send an encrypted message to Bob (without the prior communication of a secret key) by using the public encryption rule E_K. Bob will be the only person that can decrypt the ciphertext, using his secret decryption rule D_K.

2.1.1 Requirement

We show algorithms which public-key cryptography must fulfill:

1. It is computationally easy for Bob to generate a pair (PK_B, SK_B), where PK_B is a public key and SK_B is a secret key.

2. It is computationally easy for a sender Alice who know the public key PK_B and the message M which is encrypted by Alice to generate the corresponding ciphertext

$$C = E_{PK_B}(M).$$

3. It is computationally easy for the receiver Bob to decrypt the resulting ciphertext C using the private key to recover the original message

$$M = D_{SK_B}(C) = D_{SK_B}[E_{PK_B}(M)].$$

4. It is computationally infeasible for an opponent who know the public key PK_B to determinate the secret key SK_B.

5. It is computationally infeasible for an opponent who know the public key PK_B and a ciphertext C to recover the original message M.

We can add a sixth requirement that, although useful, is not necessary for all public-key applications:

6. The encryption and decryption functions can be applied in either order

$$M = E_{PK_B}[D_{SK_B}(M)].$$

There are formidable requirements, as evidenced by the fact that only one of such algorithm has received widespread acceptance in over a quarter-century since the concept of public-key cryptography was proposed.

Before elaborating on why the requirements are so formidable, let us first recast them. The requirements boil down to the need for a trapdoor one-way function. "*one-way function*" is one that maps a domain into a range such that every function value is easy whereas the calculation of the inverse is infeasible:

$$\begin{array}{ll} Y & = f(X) \quad \text{easy.} \\ X & = f^{-1}(Y) \quad \text{infeasible.} \end{array}$$

Generally, "*easy*" is defined to mean a problem that can be solved in polynomial time as a function of input size. Thus, if the length of the input is n bits, then the time cost to compute the function is proportional to n^a, where a is a fixed constant. Next, the term "*infeasible*" is a much fuzzier concept. In general, we can say a problem is infeasible if the effort to solve it grows faster than polynomial time as a function of input size. For example, if the length of the input is n bits and the time to compute the function is proposition to 2^n, it is considered infeasible. Unfortunately, it is difficult to determine if a particular algorithm exhibits this complexity. Furthermore, traditional notions of computational complexity focus on the worst-case of average-case complexity of an algorithm. These measures are worthless for cryptography, which requires that it be infeasible to invert a function for virtually all inputs, not for the worst case or even average case.

We now turn to the definition of a "*trapdoor one way function*", which, like the one-way function, is easy to calculate in one direction and infeasible to calculate in the other direction unless certain additional information is known. With the additional information, the inverse can be calculated in polynomial time. We can summarize as follows: A trapdoor one-way function is a family of invertible functions f_K, such that,

$$\begin{array}{ll} Y & = f_K(X) \quad \text{easy, if } K \text{ and } X \text{ are known.} \\ X & = f_K^{-1}(Y) \quad \text{easy, if } K \text{ and } Y \text{ are known.} \\ X & = f_K^{-1}(Y) \quad \text{infeasible, if } Y \text{ is known but } K \text{ is not known.} \end{array}$$

Thus, the development of a practical public-key scheme depends on discovery of a suitable trapdoor one-way function.

2.1.2 Application

In broad terms, we can classify the use of public-key cryptosystems into three categories:

1. *Encryption/Decryption* : The sender encrypts a message with the recipient's public key.

2. *Digital Signature* : The sender "*signs*" a message with its private key. Signing is achieved by a cryptographic algorithm applied to the message or to a small block of data that is bound in some way to the message.

3. *Key exchange* : Two sides cooperate to exchange a session key. Several quite different approaches are possible, involving the private key(s) of one or both parties.

Some algorithms are suitable for all three applications, whereas others can only be used for one or two of these applications.

2.1.3 Identity-Based Cryptosystem

An identity-based cryptographic system, which was proposed by A.Shamir [48], is an asymmetric system wherein an entity's public identification information (unique name) plays the role of its public key. This system is used as input by a trusted center T to compute the entity's corresponding private key.

In usual public-key cryptosystems, every user has a key-pair (s, P), where s is a secret key, only known to this user and P is a public key which anybody may know. By definition, public keys need not be protected for confidentiality; on the contrary, they have to be as made as public as possible. But this "publicity" becomes drawback toward active attacks, such as the substitution of a "false" public key to a "true" one in a directory. Therefore besides key-pair (s, P), we must include his identification string I and "guarantee" G that P is really the public key of user I, and not the one of an imposer I.

When we adopt the Identity-based systems, the public key is equivalent to the identity. i.e. $P = I$. And guarantee is equivalent to the secret key itself. i.e. $G = s$. Since there is no certificate to store and to check, this system has good properties.

After computing the entity's private key, T transfers the entity's private key to the entity over a secure (authentic and private) cannel. This private key is computed from not only the entity's identity information, but must also be a function of some privileged information known only to T (T's private key). This is necessary to prevent forgery and impersonation - it is essential that only T be able to create valid private keys corresponding to give identification information. Corresponding (authentic) publicly available system data must be incorporated in the cryptographic transformations of the ID-based system, analogous to the certification authority public key in certificate-based systems.

2.2 Complexity Theory

Nowadays, the security in the practical public key cryptosystem, depends on the hardness of solving problem for number theory. This section shows the known results of deterministic and probablistic polinomial-time reductions among the problems.

2.2.1 Complexity Issues in Cryptography

One very important observation is that a public-key cryptosystem can never provide unconditional security. Consequently, it is important to study the computational security of public-key systems.

Certainly a minimal expectation of a public-key cryptoystem is that this system cannot be cracked in polynomial time. If this condition is satisfied, then every polynomial time-bounded algorithm there exits infinitely many messages whose codes the algorithm, cannot crack. This leaves open possibility that there exists some algorithm, and infinitely many messages whose codes this algorithm can crack in polynomial time. For a public-key cryptoystem to be secure, it must be the case that ciphertexts cannot be cracked for most messages.

2.2.2 Complexity Class

We first define several complexity classes.

Definition 2.2.1 (Complexity Class P) The set of all decision problems that are solvable in polynomial time is called the *complexity class* **P**.

The computational tasks that need to get solved in practice are not all of the kind that take a "*yes*" or "*no*" answer. For example, we say need to find a satisfying truth assignment of a Boolean expression, not just to tell whether the expression is satisfiable; in the traveling salesman problem we want the optimal tour, not just whether a tour within a given budget exists; and so on. We call such problems requiring an answer more elaborate than "*yes*" or "*no*", "*function problems*". We can show the following definition which is associated with this problems.

Definition 2.2.2 (Complexity Class FP) The class of all function problems in **P** is called the *complexity class* **FP**.

2.2.3 Turing Reducibility

In this thesis, we introduced the (Turing) reducibility concept to evaluate the difficulty of solving the problem. These reduction are defined in the same manner of the reductions among languages over some finite alphabet. As for the concept, we obey the paper in [45].

We first define the function Q_1 to solve the program P_1, and the function q_2 to solve the program P_2, respectively. If one adopt P_1 as a subroutine, and can construct polynomial time computable program P_2, then we say *"Q_2 reduces to Q_1"*.

We show the instruction in graphical as follows:

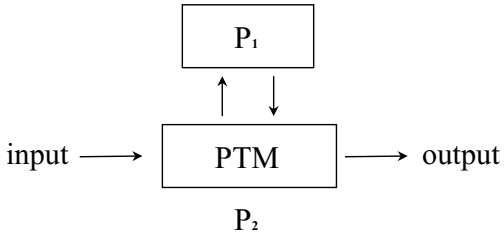

Figure 2.1: Querying an oracle.

In Figure 2.1, PTM means "*Polynomial-time Turing Machine*". With respect to reductions, there exist some classifications:

Definition 2.2.3 (Polynomial Turing reduction) For two functions f and g, the following reductions are used in this paper:

- **Many one reduction.** $f \leq_m^{fp} g$ denotes that f reduces to g with respect to (functionally) polynomial many-one reducibility. This reduction means that there exists a polynomial-time computable function h_1 and h_2 such that $f(x) = h_2(g(h_1(x)))$ for all input x.

- **Turing reduction.** $f \leq_T^{fp} g$ denotes that f reduces to g with respect to (functionally) polynomial-time Turing reducibility. This reduction means that a deterministic polynomial-time bounded Turing machine can access an oracle adaptively, where the number of queries to an oracle is bounded by the polynomial time.

- **Expected Turing reduction.** $f \leq_T^{efp} g$ denotes that f reduces to g with respect to (functionally) expected polynomial Turing reducibility. This reduction means that there exists a probablistic polynomial-time bounded Turing machine can access an oracle adaptively, where the number of queries to an oracle is bounded by expected polynomial time.

■

By definitions, it is obvious such that for function f and g,

$$f \leq_m^{fp} g \Rightarrow f \leq_T^{fp} g \Rightarrow f \leq_T^{efp} g.$$

Number theoretic problems

We first define the various number theoretic problems and also define the corresponding functions solving the problems. i.e. for a query to the function, the oracle (function) returns the answer if such an answer exists.

Definition 2.2.4 (Discrete logarithm problem over $\mathbb{Z}_n^*$) It is given $n \in N_{>1}, g \in Z_n^*, y \in Z_n^*$, to find $x \in Z_\varphi$ such that $y = g^x \bmod n$. We denote by $\mathtt{DL}_n$ the function enable to solve the problem. ■

Definition 2.2.5 (Factoring problem) It is given $n \in \mathbb{N}_{>1}$, to find a such that $1 < a < n$ and $a|n$. We denote by `FACT` the function enable to solve the problem. ■

Definition 2.2.6 (RSA problem [42]) It is given $n \in \mathbb{N}_{>1}, e \in \mathbb{Z}_{\varphi(n)}^*, y \in \mathbb{Z}_n^*$, to find x such that $y = x^e \bmod n$. We denote by `RSA` the function which enable to solve the problem. ■

Definition 2.2.7 (Strong RSA problem) It is given $n \in \mathbb{N}_{>1}, y \in \mathbb{Z}_n^*$, to find a pair (e, x) with $e > 1$ and $x \in \mathbb{Z}_n^*$ such that $y = x^e \bmod n$. We denote by `RSA` the function which enable to solve the problem. ■

Results

It is known that `st-RSA` $\leq_T^{efp}$ `RSA`, `RSA` $\leq_T^{fp}$ `FACT` and `FACT` $\leq_T^{efp}$ $\mathtt{DL}_n$ [37, 1, 52]. All the results including our results are shown in Chapter 5.

Chapter 3

Three-pass identification scheme

3.1 Introduction

Identification is a process of verifying the entity's identity. This scheme allows one entity, called *prover*, to identity herself to another entity, called *verifier*, in such a way that the adversary listening in the open network, cannot impersonate the prover.

The identification techniques, belonging to the information security, may be divided into two categories. One category is so-called *weak identification*, such as conventional password schemes. Another category is *strong identification*.

This chapter focus on a strong identification, called *challenge response* identification. One disadvantage of simple password protocol is that when a prover gives the password to the verifier, the verifier later can impersonate the prover. On the other hand, in challenge-response public-key techniques, the prover can prove its identity to the verifier without revealing the secret itself to the verifier. This might nonetheless reveal some partial information about the provers secret.

Zero-knowledge protocols, which is one of challenge response schemes are designed to avoid such a problem, by allowing a prover to demonstrate knowledge of a secret while revealing no information except for public information.

In this chapter, we introduce three-pass zero-knowledge based protocols. That is a practical scheme because the flow is only 3-pass. Moreover, there is another advantage, that is, such identification schemes can be transformed into digital signature schemes.

3.2 Model and Definition

We illustrate the general structure of 3-pass identification scheme.

- **Key generation.** it is a probabilistic algorithm which on input a string 1^k and a random tape ω_K, outputs a pair (pk, sk) with public key pk and secret key sk, where k is a security parameter.

- **Identification.** In this protocol, there exists two PPT parties, a prover P with random tape ω_P and a verifier V with random tape ω_V. We describe interactive protocol as follows. Here $A \to B$ denotes the parameter's flow from A to B for entities A and B.

  Move 1 ($P \to V$): P computes *commitment* x, and sends x to V.
  Move 2 ($V \to P$): V computes *challenge* e, and sends e to P.
  Move 3 ($P \to V$): P computes *answer* y, and sends y to V.

  On input the above parameters (x, e, y), V finally check whether the proof is valid or not.

First attempts to provide a definition to the concept of proofs of knowledge appear in Feige, Fiat and Shamir [11] and Tompa and Woll [51]. The issue of defining protocols of knowledge has been extensively investigated by Bellare and Goldreich. In this thesis, we follow the approach of Feige, Fiat and Shamir [11] and prove the security in our proposed identifications.

P and V means the honest prover and the honest verifier, respectively. We denote by $R = \{(x, s)\}$ a relation whose decision can be computed in polynomial time, where x is a common input tape and s is a knowledge of P such that $\{(x, s)\} \in R$. Let (P, V) an interactive protocol between two entities P and V. Then the interactive proof system of knowledge is defined as follows.

Definition 3.2.1 (Interactive proof system) We say that (P, V) is a interactive proof system of knowledge for the relation R, if the following conditions hold:

- **Completeness.**

$$\begin{aligned}&\exists P, \forall (x, s)\,[(x, s) \in R]\\ &\Pr\left[(P, V) \text{ accepts } x\right] > 1 - \varepsilon(|x|).\end{aligned}$$

- **Soundness.**

$$\begin{aligned}&\exists M, \forall P^*, \forall x, \forall s^*, \forall \omega_p^*\\ &\Pr\left[(P^*(s^*), V) \text{ accept } x\right] > \varepsilon(|x|)\\ &\Rightarrow \Pr\left[M(P^*(s^*, \omega_{p^*}), x) \text{ outputs } s \text{ such that } (x, s) \in R\right],\end{aligned}$$

  where P^* represents an honest/dishonest prover (PPT); ω_p^* is P^*'s random tape; and M is a PPT which on inputs p^* and x, outputs s.

■

Definition 3.2.2 (Indistinguishability) Let $\mathcal{D}_1$ and $\mathcal{D}_2$ be two distributions of probability. We denote by $\Delta_{(\mathcal{D}_1, \mathcal{D}_2)}$,the distance between $\mathcal{D}_1$ and $\mathcal{D}_2$:

$$\Delta_{(\mathcal{D}_1, \mathcal{D}_2)} = \sum_{\beta \in \{0,1\}^*} \left| \Pr_{\alpha \in \mathcal{D}_1}[\alpha = \beta] - \Pr_{\alpha \in \mathcal{D}_2}[\alpha = \beta] \right|$$

Then we classify into three indistinguishabilities.

- **Perfect indistinguishability.** We say that $\mathcal{D}_1$ and $\mathcal{D}_2$ are perfectly indistinguishable if $\Delta_{(\mathcal{D}_1,\mathcal{D}_2)} = 0$.
- **Statically indistinguishability.** We say that $\mathcal{D}_1$ and $\mathcal{D}_2$ are statically indistinguishable if $\Delta_{(\mathcal{D}_1,\mathcal{D}_2)} < \epsilon(|\alpha|)$.
- **Computational indistinguishability.** We say that $\mathcal{D}_1$ and $\mathcal{D}_2$ are computationally indistinguishable if there is not any distinguisher $\mathcal{D}$ with a non-negligible advantage.

■

It is obvious that if two distributions are perfectly (resp. statically) indistinguishable, they are statically (resp. computationally) indistinguishable.

Definition 3.2.3 (Honest verifier zero-knowledge identification protocol) We say that the protocol is honest verifier (perfectly/statically/computationally) zero-knowledge identification protocol if the distance between the real (i.e.$(P,\ V)$) and the simulated distributions is (perfectly/statically/computationally) indistinguishable. ■

3.3 Protocol

This section survey the conventional identification protocols based on or motivated by zero-knowledge techniques.

3.3.1 Schnorr scheme

Schnorr identification scheme [46, 47] allows pre-computation which reduce the real-time computation for the entity to one multiplication modulo a prime q. Consequently, one may say that it is suitable for claimants of limited computational ability.

Key generation: The following steps are executed:
1. Pick up two primes p and q satisfying $q|p-1$.
2. Find an element $g \in \mathbb{Z}_p^*$ such that $\mathrm{Ord}_p(g) = q$.
3. Pick up a random number s and compute $v = g^{-s} \bmod n$.

Secret-key/public-key: The public key is (p, q, g, v) and corresponding secret key is q.

Identification: A prover P and a verifier V execute the following steps:
1. P picks up a random number $r \in \mathbb{Z}_{2^a}$, computes the commitment $x = g^r \bmod n$ and sends x to the verifier.
2. V picks up a random challenge $e \in \mathbb{Z}_{2^k}$ and sends e to the prover.
3. P computes an answer $y = r + se \bmod q$, and sends y to the verifier.
4. V checks whether $x = g^y v^e \bmod p$ holds or not. If the equations holds, V accepts. Otherwise rejects.

3.3.2 Guillou-Quisquater scheme

Guillou and Quisquater proposed a variant of Fiat-Shamir identification protocol [12]. The security is base on the RSA problem. This protocol is specified in ISO/IEC 1488-2.

Key generation: The following steps are executed:

1. Pick up two primes p and q and compute $n = pq$.
2. Pick up $a \in \mathbb{Z}_n^*$ and compute b satisfying $ab = 1 \bmod \varphi(n)$.
3. Pick up a random number u and compute $s = u^{-b} \bmod n$.

Secret-key/public-key: The public key is (n, a, u) and corresponding secret key is s.

Identification: A prover P and a verifier V execute the following steps:

1. P picks up a random number $r \in \mathbb{Z}_{2^a}$, computes the commitment $x = r^a \bmod n$ and sends x to V.
2. V picks up a random challenge $e \in \mathbb{Z}_{2^k}$ and sends e to P.
3. P computes an answer $y = rs^e \bmod n$ and sends y to V.
4. V checks whether $x = u^e y^a \bmod n$ holds or not. If the equations holds, V accepts. Otherwise rejects.

3.3.3 Guillou-Quisquater identity-based scheme

The Guillou-Quisquater identification scheme in Section 3.3.2 can be transformed into what is known as an *identity based* identification scheme. In this case, certificates are not required. Instead, TA (Trusted Authority) computes the value u as a function of user's *ID string*, using a public hash function $\mathcal{H} : \{0,1\}^* \to \{0,1\}^n$. The value u is computed from the user's ID string via public hash function $\mathcal{H}$.

TA's parameter generation: The TA execute the following steps.

1. Pick up two distinct primes p and q and compute $n = pq$.
2. Pick up $a \in \mathbb{Z}_n^*$ and compute b satisfying $ab = 1 \bmod \varphi(n)$.

TA's parameter: The TA's public key is (n, a) and the corresponding secret key is two primes p and q (or b).

User's parameter generation: Suppose that TA issue the parameter for user P. Let *ID* be a P's ID information. Then TA executes the following steps.

1. Compute $u = \mathcal{H}(ID)$.
2. Compute $s = u^{-b} \bmod n$.

User's parameter: The P's ID string is u and the corresponding secret key is s.

Identification: Suppose P and verifier V construct an interactive proof system. Then they execute the following steps.

1. P picks up a random number $r \in \mathbb{Z}_n$ computes the commitment $x = r^a \bmod n$, and sends x to the verifier.
2. V picks up a random challenge $e \in \mathbb{Z}_{2^k}$ and sends e to P.
3. P computes the answer $y = rs^e \bmod n$ and sends y to V.
4. V computes $u = \mathcal{H}(ID)$ and checks whether $x = u^e y^a \bmod n$ holds or not. If the equations holds, V accepts. Otherwise rejects.

One can replace the redundancy function instead of $\mathcal{H}$. In this case, a example of the redundancy function is the redundancy mapping of the preprocessing stage of ISO/IEC 9796.

Chapter 4

Digital Signature Scheme

4.1 Introduction

A cryptographic primitive, which is fundamental in authentication, authorization, and non-repudiation is called the digital signature. The purpose of a digital signature is to provide a means for an entity to bind its identity to a piece of information. The process of signing entails transforming the message and some secret information held by the entity into a tag is called a signature.

The concept of a digital signature was recognized for several years, before any practical was available. The first scheme was the RSA signature scheme, which even now remains one of the most practical techniques available. Subsequent research has resulted in many alternative digital signature techniques. Some schemes offer significant advantages in terms of functionally and implementation.

This chapter describes some general models for digital signature schemes. Some concrete signature schemes, such as conventional, on the fly and proxy signatures, are also presented.

4.2 Model and Definition

A digital signature scheme consists of the following three algorithms:

- **Key generation.** As well as in the key generation of 3-pass identification scheme in Section 3.2, it is a probabilistic algorithm which on input 1^k and a random tape ω_K, outputs a pair (pk, sk) with public key pk and secret key sk, where k is a security parameter.

- **Signature generation.** It is a possibly probabilistic algorithm on input a random tape ω_S, sk and a message m to be signed, outputs a signature σ.

- **Verification.** It is a deterministic algorithm on input pk, m and a candidate signature σ for m, outputs a bit, i.e. "accept" or "reject".

The goal of attack in signature scheme is that an adversary (not actual signer) obtain forged signatures. To evaluate the security in signature schemes, we must cover two aspects, type of attack and that of forgery.

Definition 4.2.1 (Type of attack) There exist the following types of attack in signature scheme:

- **No message attack.** An adversary can only obtain a public key.
- **Known message attack.** An adversary can obtain a polynomial number of pairs of message and the corresponding signature. However she does not allow to choose those messages she like.
- **Chosen message attack.** In the same way as in known key attack, an adversary can obtain a polynomial number of pairs of message and the corresponding signature. Furthermore, she can allow to choose those messages she like.
- **Adaptively chosen message attack.** Under the above chosen message attack, an adversary can choose each message adaptively.

Definition 4.2.2 (Type of forgery) There exist the following types of forgery in signature scheme:

- **Total forgery.** An adversary obtain the secret key.
- **General forgery.** An adversary obtain the signature for a given message which she does not chooses herself.
- **Existential forgery.** An adversary obtain a pair of a message and the corresponding signature.

Definition 4.2.3 (Secure signature) We say that an signature scheme is secure if the probability, which any PPT adversary with random tape ω_A can obtain a existential forgery under the adaptive chosen message attack, is negligible:

$$\Pr_{\omega_K,\omega_S,\omega_A}[\text{Verifer accept the adversary}] < \epsilon(k)$$

Provable security. Informally, it is said that a signature scheme is *provably secure* if its security can be tied close to the security of cryptosystem and proved mathematically. In 1984, Goldwasser et al. proposed the first signature scheme [16] satisfying the above condition. In the scheme, they used the assumption of claw-free permutations pairs, that is, given two permutations f_1 and f_2, find a triple (x, y, z) such that $f_1(x) = f_2(y) = z$. Unfortunately, those provable secure schemes including [4, 29, 43] are not practical in terms of complexity or data size (or both).

Random oracle model. If the hash function in the signature scheme are modeled as being truly random functions, such a model is called *random oracle model* [12, 5]. In this model, we assume that all parties including adversary, can access to a *public random oracle*, which on input a string x, returns a (truly) random string $\mathcal{H}(x)$ with some appropriate length. In a *real world*, such an ideal hash function does not exist. So, in case of implementation, we replace the the random oracle by a "well defined" cryptographic hashing function, such as MD5 or SHA1. The signature schemes which are provably secure in random oracle model, such as PSS [6] or Guillou-Quisquater scheme [17], give the the benefit of good security with good performance. Therefore these schemes are much practical.

Converting form identification to signature scheme. As for the generic 3-pass identification protocol involving commitment-challenge-response sequence, such as Schnorr [47], the following technique proposed by Fiat and Shamir [11] may be used to convert the identification scheme to a signature scheme: One replaces random challenge e of the verifier by the appropriate one-way hashed value: $e = \mathcal{H}(x, m)$, where x is the commitment and m is the message to be signed.

4.3 Conventional Signature

4.3.1 RSA signature

The RSA signature scheme [42] was the first practical signature scheme based on public-key techniques. Let $\mathcal{H} : \{0,1\}^* \to \{0,1\}^n$ be a hash function.

Key Generation: The following steps are executed.

1. Pick up two distinct primes p and q and compute $n = pq$.
2. Pick up $e \in \mathbb{Z}_n^*$ and compute d satisfying $ed = 1 \bmod \varphi(n)$.

Public key/Secret key: The public key is (n, e) and the corresponding secret key is d (or two primes p and q).

Signature Generation: A signer executes the following step.

1. Compute $h = \mathcal{H}(m)$
2. Compute $\sigma = h^d \bmod n$

Signature: The signature of m is σ.

Verification: A verifier executes the following steps.

1. Compute $h = \mathcal{H}(m)$
2. Check $h \stackrel{?}{=} \sigma^e \bmod n$

The ISO/IEC 9796 standard lists RSA as a compatible cryptographic algorithm. PSS (Probabilistic Signature Scheme) is a provably secure way of creating signatures with RSA. The proof of security for PSS takes place in the random oracle model. The method for creating digital signatures with RSA is described in PKCS #1 [44].

4.3.2 Schnorr signature

The Schnorr signature scheme [47] is derived from an identification protocol given in Section 3.3.1. Signature generation in this scheme requires one exponentiation modulo p and one multiplication modulo p. Then the exponentiation modulo p could be done off-line. The method requires hash function $\mathcal{H} : \{0,1\}^* \rightarrow \{0,1\}^q$.

Key generation: The following steps are executed:

1. Pick up two primes p and q satisfying $q|p-1$.
2. Find an element $g \in \mathbb{Z}_p^*$ such that $\mathrm{Ord}_p(g) = q$.
3. Pick up a random number s and compute $v = g^{-s} \bmod n$.

Public key/Secret key: The public key is (p, q, g, v) and corresponding secret key is q.

Signature Generation: A signer executes the following step.

1. Pick up a random number $r \in \mathbb{Z}_q$ and compute $x = g^r \bmod n$.
2. Compute a hash value $e = \mathcal{H}(x, m)$
3. Compute $y = r + se \bmod q$.

Signature: The signature of m is (e, y).

Verification: A verifier executes the following steps.

1. Compute $x' = g^y v^e \bmod p$.
2. Compute $e' = \mathcal{H}(x, m)$.
3. Check whether $x' = g^y v^{e'}$ hold or not. If the equation holds, the verifier accept. Otherwise reject.

4.4 On the Fly Signature

4.4.1 Related Work

In 1992, Giraut [15] modified Schnorr's signature scheme [47] in which an RSA modulus is used instead of a prime modulus. This modification leads to no modulo reduction in the signature generation. Therefore, in Girault's scheme, faster processing of the signature generation is possible than in Schnorr's one. In 1998, Poupard and Stern [38] investigated and gave provable security for Girault's scheme, and named that scheme GPS-scheme.

In 1999, Poupard and Stern [39] proposed a generic signature scheme (PS-scheme), whose security relies on the difficulty of integer factoring. In this scheme, the size of the public-key is smaller than that in GPS-scheme. Consequently, compared with GPS-scheme, the computational cost and the data size can be decreased, and PS-scheme is seemed more secure under the *one-key attack* scenario [39]. However, PS-scheme has some drawbacks. For instance, the size of secret key is only dependent on modulus n, and considerably large (about $|n|/2$). This drawback leads to inefficient results in both communication work and data size. Moreover, computational cost in the verification is very high.

4.4.2 Giraut-Poupard-Stern scheme

Giraut-Poupard-Stern scheme [38] (GPS-scheme) is a variant of the Schnorr scheme. This scheme uses an RSA modulus in stead of a prime modulus in Schnorr scheme. We use a hash function $\mathcal{H} : \{0,1\}^* \to \{0,1\}^{|B|}$.

Key generation: The following steps are executed:

1. Pick up two distinct primes p and q, and compute $n = pq$.
2. Find an element $g \in \mathbb{Z}_p^*$ such that the order of g is a high order, that is, $\mathrm{Ord}_n(g) = \lambda(n)$.
3. Pick up a random number $s \in_R \mathbb{Z}_S$ and compute $v = g^{-s} \bmod n$.

Public key/Secret key: The public key is (n, g, v) and corresponding secret key is s.

Signature Generation: A signer executes the following step.

1. Pick up a random number $r \in \mathbb{Z}_A$ and compute $x = g^r \bmod n$.
2. Compute a hash value $e = \mathcal{H}(x, m)$.
3. Compute $y = r + se$ in $\mathbb{Z}$.

Signature: The signature of m is (e, y).

Verification: A verifier executes the following steps.

1. Compute $x' = g^y v^e \bmod p$.
2. Compute $e' = \mathcal{H}(x, m)$.
3. Check whether both $y < A$ and $x' = g^y v^{e'}$ hold or not. If the equations holds, the verifier accept. Otherwise reject.

The above parameters A, B and s satisfy the condition $B \ll s \ll A$.

4.4.3 Poupard-Stern scheme

Poupard-Stern scheme [39] (PS-scheme) is the first practical on the fly signature. The security of PS-scheme is provably as secure as the integer factoring problem. We use a hash function $\mathcal{H} : \{0,1\}^* \to \{0,1\}^{|B|}$.

Key generation: The following steps are executed:

1. Pick up two (distinct) strong primes p and q, and compute $n = pq$.
2. Find an element $g \in \mathbb{Z}_n^*$ such that $\mathrm{Ord}_n(g) \in \{\lambda(n), \lambda(n)/2\}$.
3. Compute $s = n - \varphi(n)$ $(= p + q + 1)$.

Public key/Secret key: The public key is (n, g) and corresponding secret key is s.

Signature Generation: A signer executes the following step.

1. Pick up a random number $r \in \mathbb{Z}_A$ and compute $x = g^r \bmod n$.

2. Compute a hash value $e = \mathcal{H}(x, m)$.
3. Compute $y = r + se$ in $\mathbb{Z}$.

Signature: The signature of m is (e, y).

Verification: A verifier executes the following steps.

1. Compute $x' = g^{y-ne} \bmod p$.
2. Compute $e' = \mathcal{H}(x, m)$
3. Check whether both $y < A$ and $x' = g^{y-ne'}$ hold or not. If the equations holds, the verifier accept. Otherwise reject.

The above parameters A, B and s satisfy the condition $B \ll s \ll A$.

Hereafter, we describe the features and the drawbacks in PS-scheme. A secret-key in PS-scheme is $s = n - \varphi(n)$ which depends only upon (a part of) the public-key n. The two parameters n and s are congruent under the modulo $\varphi(n)$, and the size of s is about a half of that of n.

Moreover, the computation of y is executed on $\mathbb{Z}$, and the information on a secret-key is protected by computing $r + se$ with condition $r \gg se$. Therefore, we can see that the size of r also depend upon that of se.

In the verification step, the size of y has to be explicitly verified whether the condition $y < A$ holds or not. This kind of verification cannot be seen in the existing signature schemes [13, 42, 47], hence we can say that such a verification indeed characterizes PS-scheme.

Unfortunately, PS-scheme has the following drawbacks.

High computational cost for verifier: In the verification step, $y \ll ne$ holds actually. And the order of $g \in \mathbb{Z}_n^*$ is not open. Therefore, the computational cost for a verifier is considerably large as $|ne|$ increases. The verifier must compute full exponentiation ($|y - ne|$ bits) calculus such as $x = g^{y-ne} \bmod n$.

Inefficiency by the increase of a secret-key size: If the size of a secret-key s increase for the security reason, then this scheme shall get inefficient in view of (1) the computational cost for pre-computation, signature generation and verification, and (2) data size such as the size of signature.

Restriction for the structure of a public-key n: When we set up a public-key n to be the product of three or more primes, the size of a secret-key shall accordingly increase. For example, in case n is the product of three primes, that is, p, q and r, the secret-key s $(= n - \varphi(n))$ turns out to be $n - (p-1)(q-1)(r-1)$ $(= pq + qr + rp - (p+q+r) + 1)$, whose size is about $3/2$ times of that in case n is the product of two primes.

Our fast on-line schemes (Scheme I and II) solve the above problems. Scheme I is obtained by improving PS-scheme, and Scheme II is more efficient than Scheme I. Those our schemes are described in Section 6.

4.5 Proxy Signature

4.5.1 Related Work

In 1996, Mambo, Usuda, and Okamoto [27] formulated the general idea of *proxy signature* and gave a solution by proposing the primitive schemes. This scheme allows a designated person, called a *proxy signer* to sign on behalf of the *original signer*. In their protocols, an original signer Alice creates a *proxy secret key* σ from her *original secret key* x, and secretly gives it to a proxy signer Bob. In order to generate a signature instead of Alice, he chooses a message m_p, and generates a *proxy signature* $Sig_\sigma(m_p)$ by using his proxy secret key σ. Then a verifier e.g., Carol can know that the signature is generated by Bob.

In 1997, Kim, Park and Won [21] revised the above concept to propose the two new types of digital proxy signatures, which are called partial delegation with warrant and threshold delegation. Zhang [53] proposed two schemes for partial delegation with nonrepudiable property, which means it is possible to decide who the actual signer of a proxy signature is. Therefore, in these schemes only the proxy signer can create a valid signature for the original signer.

In 1998, Lee, Hwang and Wang [24] pointed out that Zhang's second scheme [21] does not have such a property and showed that a dishonest proxy signer can cheat to get the original signer's signature. Hence they modified the scheme to prevent such attacks.

4.5.2 Mambo-Usuda-Okamoto scheme

We summarize Mambo-Usuda-Okamoto proxy signature scheme [27] (MUO-scheme). S_o, S_p and V denote original signer, proxy signer and verifier, respectively. We first show some parameters.

- **Original signer's parameter:** The original public key is (p, g, y), and the corresponding original secret key is x.
- **Proxy signer's parameter:** The proxy public key is K and the corresponding proxy secret key is σ.
- **Proxy signature:** The proxy signature of m_p is σ.

The protocol of MUO-scheme is given as follows.

Original parameter generation: S_o executes the following steps.

1. (Original generation) S_o picks up an primes and finds an element $g \in Z_p^*$. She picks up a random number $x \in Z_{p-1}$ and computes $y = g^x \bmod p$.

Proxy parameter generation: S_o and S_p executes the following steps.

2. (Proxy generation) S_o picks up a random number $k \in Z_{p-1}$ and computes a pair (K, σ) with $K = g^k \bmod p$, and $\sigma = x + kK \bmod p - 1$

3. (Proxy delivery) S_o sends a pair (K, σ) to S_p through a secure channel.
4. (Proxy verification) S_p checks whether $g^\sigma = yK^K \bmod p$ hold or not. If it does, then S_p accepts it as a valid proxy parameter. Otherwise she rejects it.

Signature and verification phase: S_p and V execute the following steps.

5. (Signing by the proxy signer) In order to generate a signature on behalf of S_o. S_p chooses a message m_p, by using her proxy secret key σ, and generates a proxy signature $Sig_\sigma(m_p)$ by executing a ordinary (generic) signing operation (such as Schnorr scheme [47]). She sends $(m_p, Sig_\sigma(m_p))$ to V.
6. (Verification of the proxy signature) V uses the original and proxy public keys (n, g, y, K) and executes the checking operation as in the ordinary signature scheme.

Unfortunately, if someone directly implements this scheme, we will have the following problem.

Proxy Signer's Deviation. Proxy signer can freely sign *any* message which he want to do, even if the original signer does not intend to do so.

One primitive solution for them would be obtained by adding *usage condition* $\mathcal{M}_p^{uc}$ to the above scheme. This is the message which includes proxy signer's identity information, deadline for signing power, categories which means a kind of messages she can sign, and so on. Then Alice generates a signature $Sig_x(\mathcal{M}_p^{uc})$ for $\mathcal{M}_p^{uc}$ by using her original secret key x and sends $(\mathcal{M}_p^{uc}, Sig_x(\mathcal{M}_p^{uc}))$ to Bob. Then he deals it with a kind of *certificate*. To sign a message instead of Alice, Bob generates a signature and sends it with that certificate. In verification Carol checks the validity of both the signature and the certificate. Therefore one more verification should be required, once a usage condition is just added. Apparently it is rather troublesome since one verification need at least one modular exponentiation.

4.5.3 Kim-Park-Won scheme

Kim, Park and Won propose new types of proxy signatures [21] (KPW-scheme), called partial delegation with warrant. The proxy public key σ in this case is computed from original signers public key x and a usage condition $\mathcal{M}_p^{uc}$. Let S_o, S_p and V be original signer, proxy signer and verifier, respectively. We use a hash function $\mathcal{H} : \{0,1\}^* \to \{0,1\}^t$, where t is a security parameter.

In KPW-scheme, original signer's parameter, proxy signer's parameter, Proxy signature and original parameter generation are the same as MUO-scheme. The protocol of KPW-scheme is given as follows.

Proxy parameter generation: S_o and S_p executes the following steps.

2. (Proxy generation) S_o picks up a random number $k \in Z_{p-1}$ and computes a pair (K, σ), where $K = g^k \bmod p$, $\sigma = ex + k \bmod p - 1$ and $e = \mathcal{H}(\mathcal{M}_p^{uc}, K)$
3. (Proxy delivery) S_o sends a pair (K, σ) to S_p through a secure channel.

4. (Proxy verification) S_p computes $e = \mathcal{H}(\mathcal{M}_p^{uc}, K)$. She checks whether $g^\sigma = y^e K \bmod p$ hold or not. If it does, then S_p accepts it as a valid proxy parameter. Otherwise she rejects it.

Signature and verification phase: S_p and V execute the following steps.

5. (Signing by the proxy signer) In order to generate a signature on behalf of S_o. S_p chooses a message m_p, by using her proxy secret key σ, and generates a proxy signature $Sig_\sigma(m_p)$ by executing a ordinary (generic) signing operation (such as Schnorr scheme [47]). She sends $(m_p, Sig_\sigma(m_p))$ to V.
6. (Verification of the proxy signature) V uses the original and proxy public keys (n, g, y, K) and executes the checking operation as in the ordinary signature scheme.

Note that V must explicitly obtain $\mathcal{M}_p^{uc}$ from somewhere, and $\mathcal{M}_p^{uc}$ should be used in verification every time. This leads to the terrible inefficiency of the transmitted data size. It is desired that new proxy signature schemes controlled by the usage condition can be realized with a little additional computation time and additional memory.

Our proposed proxy signature schemes, DLP-PS and RSA-PS, solve the above drawbacks. We explain our schemes in Section 7.

Chapter 5

New Mathematical Problem and Its Application

5.1 Introduction

In this chapter, we propose two kinds of the mathematical problems, *self powering RSA problem* and *extended finding order problem*. Each problem is used as the underlying problem of our schemes.

To estimate the difficulties of the problems, we use the technique of Turing reducibility. The way of description is the same as that of Section 2.4.

All the results of the reducibility are illustrated in Figure 5.1. In the figure, it is shown that for functions f, g and $R \in \{\leq_m^{fp}, \leq_T^{fp}, \leq_T^{efp}\}$, $f \xrightarrow{R} g$ means $f\ R\ g$.

5.2 Modified RSA Problem

We describe our new problems which are variations of the RSA problem.

Definition 5.2.1 (Self-powering RSA problem) It is given an RSA modulus n, to find $r > 1$ and $s \in \mathbb{Z}_n^*$ such that $r = s^r \bmod n$. We denote by `inv-RSA` the function enable to solve the problem. ■

One more problem can be proposed by switching a parameter r from the find part to the given part.

Definition 5.2.2 (Extended self-powering RSA problem) It is given an RSA modulus n and $r \in \mathbb{Z}_{\varphi(n)}^*$, to find $s \in \mathbb{Z}_n^*$ such that $r = s^r \bmod n$. ■

One may recall another variation of RSA problem, called *strong RSA problem* [2, 14]. This is proposed by Fujisaki and Okamoto.

Strong RSA problem is a problem, given an RSA modulus n and $y \in \mathbb{Z}_n$, find a pair (x, e) such that $y = x^e \bmod n$. In the problem. we can see four parameter (n, e, x, y)

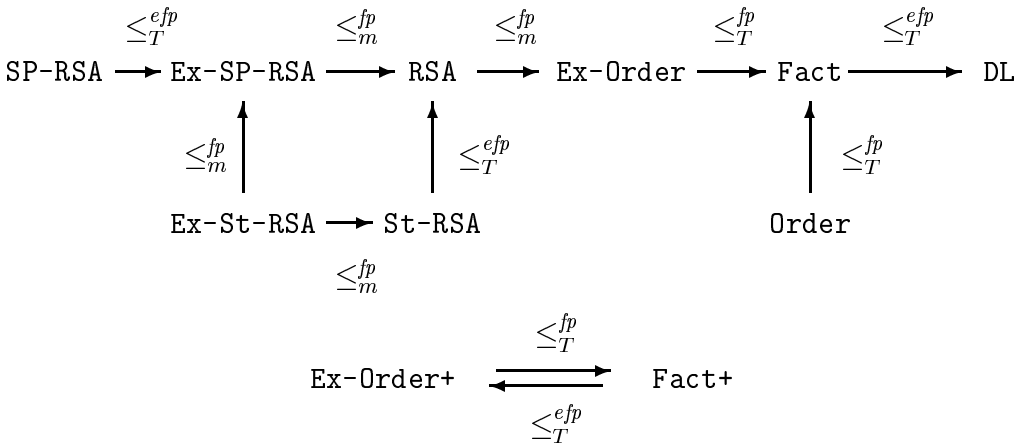

Figure 5.1: Reductions among functions.

On the other hand, our problem has only three parameter (n, r, s), and parameter r is appeared in two places. This means that our problem is the problem to find r th root of r for modulo n.

Table 5.2 lists the four possibilities of the variations.

Table 5.1: Variations of the RSA problem.

	Name	Given part	Find part	Equation
1	(original) RSA	n, e, y	x	$y = x^e \bmod n$
2	strong RSA	n, y	e, x	$y = x^e \bmod n$
3	self-powering RSA	n	r, s	$r = s^r \bmod n$
4	extended self-powering RSA	n, r	s	$r = s^r \bmod n$

We show the results of Turing reducibility.

Theorem 5.2.3 `inv-RSA` $\leq_T^{efp}$ `ex-inv-RSA`.

Proof. Assume that n is an instance of `inv-RSA`. Firstly $k \in \mathbb{Z}_n^*$ is picked up at random. We first pick up $k \in \mathbb{Z}_n$ at random. Then (n, k) be a query to `ex-inv-RSA`. If $\gcd(k, \varphi(n)) = 1$, the oracle returns s such that $k = s^k \bmod n$. In [40] it is shown that $\varphi(n) \geq \ln(2)/\ln(2n) \cdot n$ for positive number n. Hence the probability μ which the condition

is satisfied at one time is estimated as

$$\begin{aligned}\frac{\varphi(\varphi(n))}{n} &\geq \frac{\ln 2}{\ln(2\varphi(n))} \cdot \frac{\varphi(n)}{n} \\ &\geq \frac{\ln 2}{\ln(2\varphi(n))} \cdot \frac{\ln(2)}{\ln(2n)} \\ &\geq \left(\frac{\ln(2)}{\ln(2n)}\right)^2 \\ &= \frac{1}{\log^2(2n)}.\end{aligned}$$

Therefore the expected times of repetition to obtain k satisfying $\gcd(k, \varphi(n)) = 1$, is less than $\log^2(2n)$, which is bounded by a polynomial in expected polynomial time in $|n|$. This means that `inv-RSA` reduce to `ex-inv-RSA` in expected polynomial time. □

Theorem 5.2.4 `ex-inv-RSA` $\leq_m^{fp}$ `RSA`.

Proof. Assume that (n, r) is an instance of `ex-inv-RSA`. Let (n, y) be a query to `RSA`. Since $\gcd(r, \varphi(n)) = 1$, the oracle `RSA` always returns s satisfying $r = s^r \bmod n$. Hence s is regarded as the output of `ex-inv-RSA`(n, r). □

Theorem 5.2.5 `ex-inv-RSA` $\leq_m^{fp}$ `ex-st-RSA`.

Proof. Assume that (n, y) is an instance of `ex-inv-RSA`. Note that, in `ex-st-RSA` setting, $\gcd(y, \varphi(n)) = 1$. This means that for the query (n, y) to `ex-st-RSA`, the oracle always returns the answer x such that $y = x^y \bmod n$. Then (x, y) is regarded as the answer of `ex-inv-RSA`. □

Theorem 5.2.6 `ex-st-RSA` $\leq_m^{fp}$ `st-RSA`.

Proof. The difference between `ex-st-RSA` and `st-RSA` is that $\gcd(y, \varphi(n)) = 1$ in `ex-st-RSA` whereas $y \in \mathbb{Z}_n$ in `st-RSA`, and the other parameters are the same. Consequently this theorem is trivial. Assume that (n, y) is an instance of `ex-st-RSA`. For the query (n, y) to `st-RSA`, the answer s of the oracle `st-RSA` is directly regarded as the answer of `ex-st-RSA` since $\gcd(y, \varphi(n)) = 1$. □

5.3 Modified Finding Order Problem

In this section, we propose another problem, *extended finding order problem*.

Definition 5.3.1 (Extended finding order problem) It is given $n \in \mathbb{N}_{>1}$ and $g \in \mathbb{Z}_n^*$, to find L, where L is a multiple of $\mathrm{Ord}_n(g)$ and $|L|$ is bounded by a polynomial in $|n|$. We denote by `Order` the function enable to solve the problem. ■

Okamoto [33] and Brickell and McCurley [7] proposed variant of Schnorr's scheme [47], respectively. In their papers, they consider the following problem:

Definition 5.3.2 (Finding order problem) It is given $n \in \mathbb{N}_{>1}$ and $g \in \mathbb{Z}_n^*$, to find q such that $\mathrm{Ord}_n(g) = q$. We denote by `Order` the function enable to solve the problem.

We can say that our proposed problem is a variant of finding order problem. The main difference is that the output of finding order problem is just the order of n, that is $\mathrm{Ord}_n(g)$, whereas the variant one is a multiple of the order.

The results of Turing reducibility are stated as follows. Let `RSA` and `FACT` be two functions given in Section 2.2.3.

Theorem 5.3.3 `RSA` $\leq_m^{fp}$ `Ex-Order`.

Proof. Assume that (n, e, y) is an instance of `RSA`. Then (n, y) is a query for `Ex-Order`. The oracle `Ex-Order` returns L such that $y^L = 1 \bmod n$. Then we extract an integer b satisfying $L = e^a b$ for an integer a. Note that $\gcd(e, b) = 1$, we can compute d' such that $ed' = 1 \bmod b$. Finally we compute $x' = y^{d'} \bmod n$. Since $x'^e = (y^{d'})^e = y^{c\lambda(n)+1} \bmod n$ for an integer c, x' can be regarded as the answer of `RSA`. The above algorithm has the running time of $O(|L||n|^2)$, which is bounded by polynomial in $|n|$. □

Theorem 5.3.4 `Ex-Order` $\leq_T^{fp}$ `FACT`.

Proof. Assume that (n, g) is an instance of `Ex-Order`. Then we can get all the factors of n by using an oracle `FACT` repeatedly. The iteration is bounded by polynomial in $|n|$ because the number of the factors of n is less than $|n|$. Next we compute $\varphi(n)$. Since $\mathrm{Ord}_n(g) | \varphi(n)$, x' can be regarded as the answer of `Ex-Order`. □

Theorem 5.3.5 `Ex-Order` $\leq_m^{fp}$ `Order`.

Proof. Assume that (n, g) is an instance of `Ex-Order`. For a query (n, g) to `Order`, the oracle returns $q = \mathrm{Ord}_n(g)$. The answer of `Ex-Order` is q. □

In the public-key cryptosystem, such as RSA, the modulus n must satisfy the condition such that $\varphi(n)$ is not smooth for the security reason. We next consider the Turing reducibility of functions which have such an n.

We define two functions `Order+` and `FACT+`. `Order+` and `FACT+` are the same functions as `Order` and `FACT` respectively, except that the number t of factors of n satisfy $t = O(\log |n|)$ and all the prime factors of n are strong primes.

In the above setting, we obtain the results as follows.

Theorem 5.3.6 `Ex-Order+` $\leq_T^{fp}$ `FACT+`.

Assume that (n, g) is an instance of Ex-Order+. As well as in Theorem 5.3.4, we can get all the factors of n by using an oracle FACT+. Then the answer of Ex-Order+ is obtained by computing $\varphi(n)$. □

Theorem 5.3.7 FACT+ $\leq_T^{efp}$ Ex-Order+.

Proof. Assume that n is an instance of FACT+. We first pick up $k \in \mathbb{Z}_n$ at random. For a query (n, k) to Ex-Order+, the oracle returns L such that $L = c \cdot \mathrm{Ord}_n(k)$ for some integer c. If $\mathrm{Ord}_n(k) = \lambda(n)$, that is, L is a multiple of maximal order $\lambda(n)$, then such a L lead to the factorization of n by using the algorithm [28] and the running time is $O(|n||L|)$. The probability μ which satisfy the above condition at one time, is estimated as

$$\mu = \frac{\varphi(\lambda(n))}{\varphi(n)} = \frac{\varphi(\prod_i p_i)}{2^t \prod_i p_i} = \frac{1}{2^t} \prod_i \left(1 - \frac{1}{p_i}\right) > \frac{1}{2^{2t}}.$$

Consequently, the expected number of repetition to obtain such a L is less than 2^{2t}. Since t satisfies $t = O(\log |n|)$, such an expected time is bounded by a polynomial. This means that the reduction algorithm executes in expected polynomial time. □

5.4 Application

In this section, we describe the application of self-powering RSA problem.

Modified Guillou-Quisquater Identity-based Identification scheme

TA's parameter generation: The TA execute the following steps.

1. Pick up two distinct primes p and q and compute $n = pq$.

TA's parameter: The TA's public key is n and the corresponding secret key is two primes p and q.

User's parameter generation: Suppose that TA issue the parameter for a user P. Let u be a P's ID information. Then TA executes the following steps.

1. Compute u' such that $uu' = 1 \bmod \varphi(n)$.
2. Compute $s = u^{-u'} \bmod n$.

User's parameter: The P's ID string is u and the corresponding secret key is s.

Identification: Suppose P and verifier V construct an interactive proof system. Then they execute the following steps.

1. P picks up a random number $r \in \mathbb{Z}_n$, computes the commitment $x = r^u \bmod n$, and sends x to the verifier.
2. V picks up a random challenge $e \in \mathbb{Z}_{2^k}$ and sends e to P.

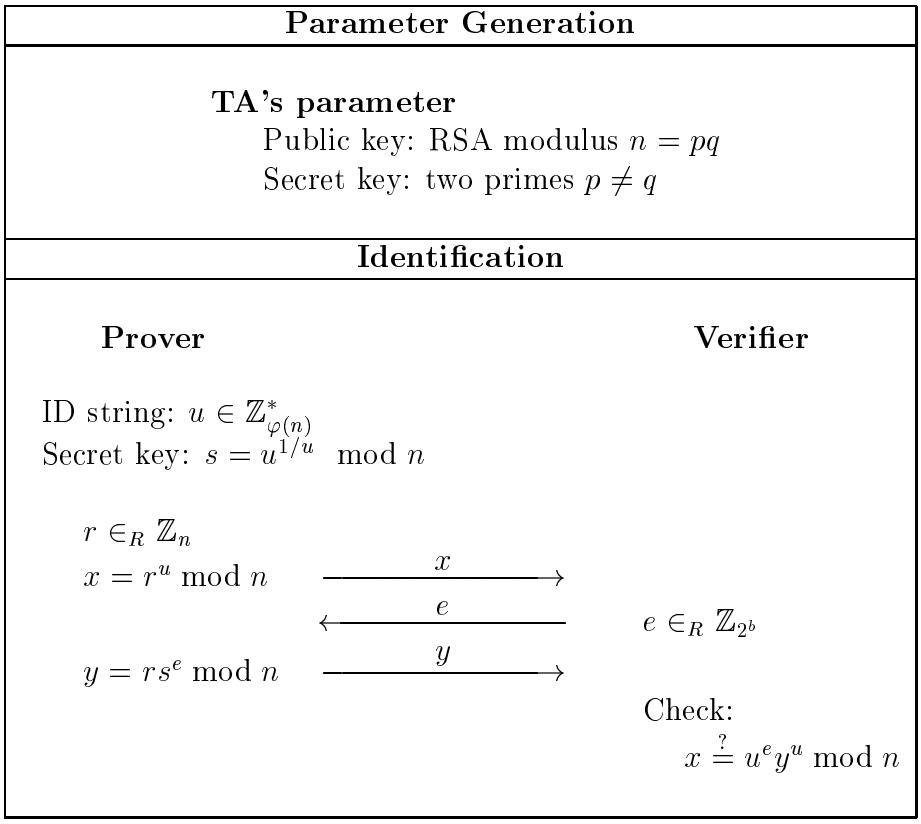

Figure 5.2: Modified Guillou-Quisquater identity-based identification scheme.

3. P computes the answer $y = rs^e \bmod n$ and sends y to V.
4. V computes $u = \mathcal{H}(ID)$ and checks whether $x = u^e y^u \bmod n$ holds or not. If the equations holds, V accepts. Otherwise rejects.

We list the difference between the proposed scheme and the original one.

Table 5.2: Comparison of original and proposed Guillou-Quisquater identity-based identification schemes.

	TA		Prover	
	Public key	Secret key	ID string	Secret key
Original scheme	n, e	p, q	$u = \mathcal{H}(ID)$	$s = u^{-1/e} \bmod n$
Proposed scheme	n	p, q	$u \in \mathbb{Z}^*_n$	$s = u^{-1/u} \bmod n$

5.5 Conclusion

We have constructed two mathematical problems and investigated those difficulties. A variants of Guillou-Quisquater identity-based identification scheme have also proposed. This scheme, based on our proposed problem, is efficient since there is no assumption of the hash function.

At present, the following questions are not known:

- Whether finding order problem reduce to extended fining order problem in the sense of polynomial time Turing reducibility;
- Whether RSA problem reduce to self-powering RSA problem in the sense of polynomial time Turing reducibility.

Those questions remain as open questions.

Chapter 6

Fast On-Line Signature

6.1 Introduction

In this chapter, we propose two signature schemes, named Scheme I and Scheme II, which have signer's on-line computational efficiency.

Our schemes are as secure as integer factoring problem for modulus n (in the random oracle model). To satisfy the security, we uses *asymmetric basis* g in $\mathbb{Z}_n^*$ which is a variant of [34]. This property leads to the good efficiency in terms of both size of data and amount of work.

Scheme I is a improved version of PS-scheme and realize a compactness. The performance in Scheme I is much superior to that in PS-scheme. Especially, the computation work in verification are much reduced by the changing n in $x = g^{y-ne} \bmod n$ (PS-scheme) into z in $x = g^{y-ze} \bmod n$ (our schemes).

Scheme II would require a modular exponentiation as preprocessing. However, no multiplication is used in the actual (i.e. on-line) signature generation. This means that the phase involves only a hashing operation, addition and a modular reduction. This paper is the first approach to obtain the fast signature scheme without on-line modular multiplication.

6.2 Preliminary Description

In this section, we describe security model. For our security analysis, we follow the approach of Feige, Fiat and Shamir [10] described in Section 3.

In [38], one can see the two types of attacks as follows.

- **One key attack.** An adversary try to forge valid signatures for fixed public key.
- **Possible key attack.** An adversary try to forge valid signatures for possible public keys, where "possible public key" means any public key satisfying the condition of the parameter.

The one key attack scenario seems to analyze the security more strictly than the possible key attack scenario. Therefore, all our schemes are estimated under the one key attack scenario.

Furthermore, we would like to establish theorems claiming that illegal actions such as impersonation are as difficult as mathematically well established problems. Therefore, Scheme I and II use slightly generalized asymmetric basis [34], which is defined as follows.

Definition 6.2.1 (Asymmetric basis) Let n be an RSA modulus such that $n = pq$. Then we say that g is an asymmetric basis in $\mathbb{Z}_n^*$ if the multiplicity of 2 in $\mathrm{Ord}_p(g)$ is not equal to that of 2 in $\mathrm{Ord}_q(g)$. ■

Lemma 6.2.2 Let n be an RSA modulus and g be an asymmetric basis in $\mathbb{Z}_n^*$. Assume that we find $L > 0$ such that $g^L = 1 \bmod n$. Then we can construct a Turing machine M which on input n, g and L outputs a factor of n in time $O(|L||n|^2)$

Proof. This lemma is basically due to [34]. Hereafter, we describe how to construct M.

At first, M extract the odd part b of L, such that $L = 2^a b$. Since g is an asymmetric basis in $\mathbb{Z}_n^*$, it holds $g^{2b} = 1 \bmod p$ and $g^{2b} = 1 \bmod q$, and also holds $g^b = 1 \bmod p$ and $g^b = -1 \bmod q$. Then we have the following results: $p \mid g^b - 1$ and $n \nmid g^b - 1$. Consequently, M can find a factor of n by computing $\gcd(g^b - 1 \bmod n, n)$.

Note that the modular exponentiation algorithm and the extended Euclidean algorithm have a running time of $O(|L||n|^2)$ and $O(|n|^2)$, respectively. Hence M can execute the above steps in time $O(|L||n|^2)$. □

In Section 6.5, we discuss the schemes in which the modulus n is built from more than three primes. In this case, those schemes cannot indicate the equivalence of integer factoring problem. Therefore, we show that one can solve the *extended finding order problem* if such schemes break. The extended finding order problem is a problem relative to the integer factoring and described in Section 5.3.

6.3 On the Fly Signature

6.3.1 Identification Scheme

Scheme I has the parameters k, κ, a, b and c with $2^{k-1} < s < 2^k \ll 2^a \ll 2^{bc}$. For more detailed conditions, we refer to Section 6.3.3.

Key generation step: The following steps are executed:

1. Pick up two same-size primes p and q, and compute $n = pq$.
2. Choose an element $g \in \mathbb{Z}_n^*$ which is an asymmetric basis in Z_n^*.
3. Pick up a random number $z \in \mathbb{Z}_{2^c}$ and computes $s = z \bmod q$, where $\mathrm{Ord}_n(g) = q$.

Secret-key/public-key: The secret-key is s and the corresponding public-key is (n, g, z).

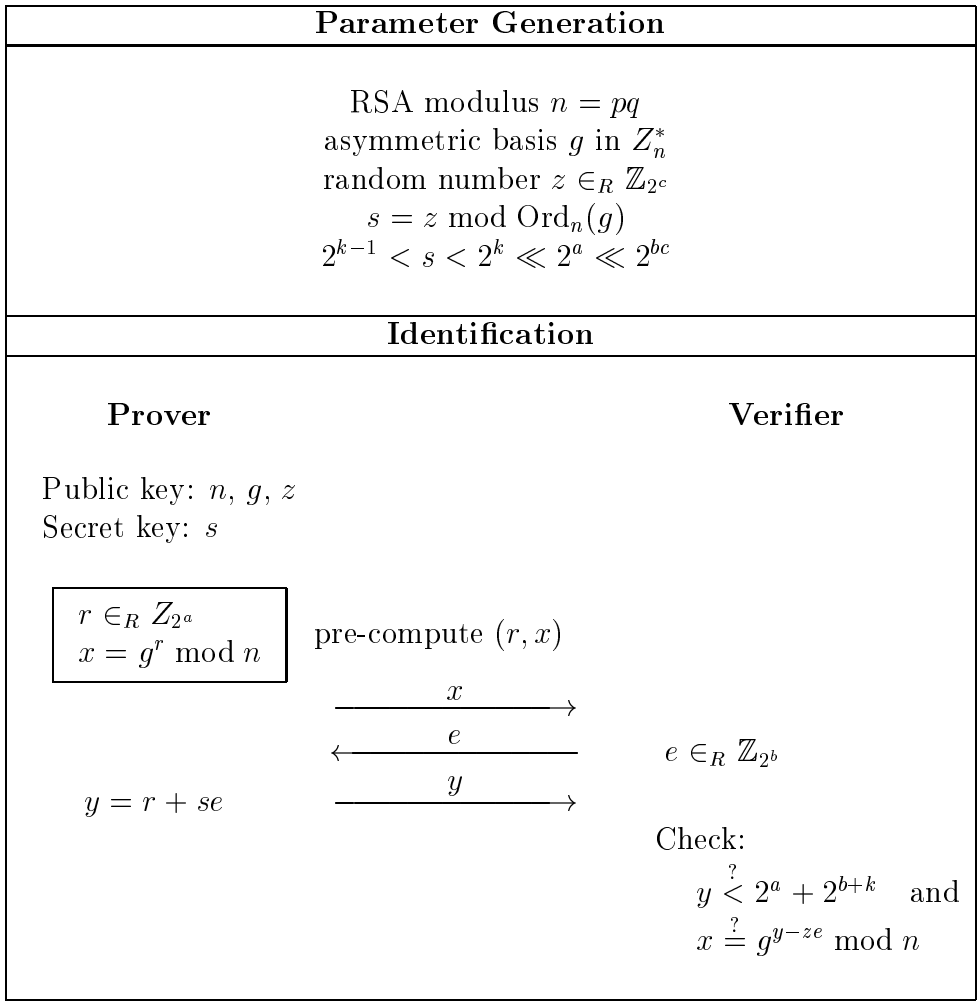

Figure 6.1: Proposed identification scheme: Scheme I

Identification step: A prover and a verifier execute the following steps:

1. The prover picks up a random number $r \in \mathbb{Z}_{2^a}$, computes the commitment $x = g^r \bmod n$ and sends x to the verifier.
2. The verifier picks up a random challenge $e \in \mathbb{Z}_{2^b}$ and sends e to the prover.
3. The prover computes an answer $y = r + se$ in $\mathbb{Z}$, and sends y to the verifier.
4. The verifier checks whether both $y < 2^a + 2^{b+k}$ and $x = g^{y-ze} \bmod n$ hold or not. If both equations hold, the verifier accepts. Otherwise she rejects.

The security is as secure as integer factoring problem for modulus n (in the random oracle model). To satisfy the security, Scheme I uses *asymmetric basis* g in $\mathbb{Z}_n^*$ which is a variant of [34].

Security Analysis

We show that our proposed scheme is a three-pass honest-verifier statically zero-knowledge identification protocol. As a strategy, we focus on that our scheme provides completeness, soundness and the honest-verifier statistical zero-knowledge property, respectively.

Theorem 6.3.1 (Completeness) In the proposed identification scheme, the prover who has a secret key, and who follows the scheme, is always successfully accepted.

Proof. The answer y correctly computed by the prover with the secret-key, is $r + se$ with $|se| < |r| \leq a$. Hence such a y satisfies $y < 2^a 2^{b+k}$. Furthermore, from the following equations:

$$g^{y-ze} = g^{(r+se)-ze} = g^{r+(z \bmod q)e-ze} = g^r = x \bmod n,$$

a correctly generated answer y always passes the verification. □

Theorem 6.3.2 Assume that a PPT adversary $\widetilde{P}$ is accepted by honest verifiers with probability $1/2^{b\ell} + \varepsilon$ with $\varepsilon > 0$. Then there exists a PPT machine M, which can figure out the secret-key s from the public-key, with overwhelming probability, in time $O(2^b \ell |n|^{O(1)}/\varepsilon)$.

Proof. Denote, by $\widetilde{P}(\omega)$, an adversary $\widetilde{P}$ with a random tape ω. Also denote, by $\wp(\omega, \boldsymbol{e})$, the predicate that $\widetilde{P}(\omega)$ is accepted for challenges $\boldsymbol{e} = (e_1, \ldots, e_\ell)$. Then with notation by [5], we have $\mathtt{Pr}\left[\omega \leftarrow \mathcal{T}; e_i \leftarrow \mathbb{Z}_{2^b}\ (1 \leq i \leq \ell)\ :\ \wp(\omega, \boldsymbol{e})\right] \geq 1/2^{b\ell} + \varepsilon$, where $\mathcal{T}$ is a set of possible random tapes. Let $\Omega \subset \mathcal{T}$ be the set of random tapes ω such that $\mathtt{Pr}\left[e_i \leftarrow \mathbb{Z}_{2^b}; (1 \leq i \leq \ell)\ :\ \wp(\omega, \boldsymbol{e})\right] \geq 1/2^{b\ell} + \varepsilon/2$. From *the Splitting Lemma* in [35], we have $\mathtt{Pr}\left[\omega \leftarrow \mathcal{T} : \omega \in \Omega\right] \geq \varepsilon/2$.

As follows, we construct a machine M which, on input of the public data, outputs two valid answers y_1, y_2 and the corresponding challenges e_1, e_2. First M picks up a random $\widetilde{g} \in \mathbb{Z}_n^*$. The probability for one trial that the order of $\widetilde{g}$ is $\lambda(n)$ is greater than $1/2^{2t}$ from the following:

$$\frac{\varphi(\lambda(n))}{\varphi(n)} = \frac{\varphi(\prod_i q_i)}{2^t \prod_i q_i} = \frac{1}{2^t} \prod_i \left(1 - \frac{1}{q_i}\right) > \frac{1}{2^{2t}}.$$

Then M chooses a random $\widetilde{z} \in [2^{k+\kappa}, 2^c)$, and gives $(n, \widetilde{g}, \widetilde{z})$ to $\widetilde{P}(\omega)$ as the public data. Let $1 \leq i \leq \ell$. For each i, $\widetilde{P}(\omega)$ gives the i-th commitment x_i. Then M repeats the following 2^b times: M returns a challenge and gets the answer from $\widetilde{P}(\omega)$, and M resets $\widetilde{P}(\omega)$ to be of the state just before M sends the challenge. Since $\mathtt{Pr}\left[\omega \leftarrow \mathcal{T}; e \leftarrow \mathbb{Z}_{2^b} : \omega \in \Omega\ \&\ \wp(\omega, \boldsymbol{e})\right] \geq \varepsilon/2$, with probability at least $\varepsilon/2$, $\omega \in \Omega$, and after that execution, M has two valid answer y_1, y_2 and the corresponding two challenges e_1, e_2. By repeating this procedure $2|n|/\varepsilon$ times, M can obtain such y_1, y_2, e_1, e_2 with overwhelming probability in time $O(2^b \ell |n| T/\varepsilon + |n|^{O(1)})$.

The two pairs (e_1, y_1) and (e_2, y_2) should pass the verification for the same commitment x. That means $x = \widetilde{g}^{y_1 - \widetilde{z}e_1} \bmod n = \widetilde{g}^{y_2 - \widetilde{z}e_2} \bmod n$. This implies $\widetilde{g}^{(y_1 - y_2) - \widetilde{z}(e_1 - e_2)} = 1 \bmod n$ and hence $L := (y_1 - y_2) - \widetilde{z}(e_1 - e_2)$ is a multiple of the order of $\widetilde{g}$.

Since the probability that an order-$\lambda(n)$ base is picked up is at least $1/2^{2t}$, by repeating the process given above $2^{2t}|n|$ times[1], M can meet the case that the order of $\widetilde{g}$ is $\lambda(n)$ and that such (e_1, y_1) and (e_2, y_2) are obtained, with overwhelming probability in time $O(2^b \ell T |n|^{O(1)}/\varepsilon)$.

By using technique of Lemma 6.2.2, M can factor n, and hence figure out the secret-key s from the public-key z. □

Theorem 6.3.3 (Soundness) Assume that a polynomial-time adversary $\widetilde{P}$ is accepted by honest verifiers with non-negligible probability, that $\log(|n|) = o(\ell)$, and that ℓ is bounded by a polynomial on $|n|$. Then there exists a probabilistic polynomial-time machine M, which can figure out the secret-key s from the public-key, with overwhelming probability.

Proof. This proposition can be shown as well as [38]. Let $\pi(|n|)$ be that non-negligible probability. Then there exists a d such that $\mu(|n|) \geq 1/|n|^d$ for infinitely many $|n|$'s. Since $\log(|n|) = o(\ell)$, $1/k^\ell \leq 1/2|n|^d$ holds for $|n|$ large enough. Letting $\varepsilon = \pi(|n|)/2$, we can obtain that the secret-key is figured out in time $O(2^b \ell |n| T/\varepsilon + |n|^{O(1)})$. Therefore since both ℓ and $1/\varepsilon$ are bounded by a polynomial on $|n|$, there is a polynomial-time machine M which can compute the secret-key from the public data, with overwhelming probability. □

Theorem 6.3.4 (Zero-knowledge property) The proposed identification scheme provides the statistical zero-knowledge property, if $qT/2^a$ is negligible, where $T(|n|)$ is the maximal number of repeating of the identification with an identical key.

Proof. We prove this proposition as well as [38]. To prove the proposition, we show that we can construct a polynomial-time machine (simulator) M which simulates the communication between the prover P and a dishonest verifier $\widetilde{V}$ trying to get information on s by choosing the challenges not randomly but by some computation. In the i-th round of the identification, $\widetilde{V}$ has already obtained some information, denoted by I_i, P sends a commitment x_i, and $\widetilde{V}$ chooses the challenge $e_i(I_i, x_i)$ possibly using I_i and t_i.

For the i-th round of the identification, the simulator M randomly picks up $e'_i \in \mathbb{Z}_{2^b}$ and $y'_i \in [\Phi, 2^a)$, where $\Phi := (2^b - 1)(2^k - 1)$. If $e_i(I_i, x'_i) \neq e'_i$, then M tries again with another pair (e'_i, y'_i). Otherwise M outputs (x'_i, e'_i, y'_i).

For a function $F : \mathbb{Z}_n \to \mathbb{Z}_{2^a}$, an integer A and a positive constant Δ, $\mathcal{N}(F, A, \Delta)$ is defined to be the number of pairs $(e, y) \in [0, 2^b) \times [A, A + \Delta)$ such that $F(g^{y - ze}) = e$. If $F(g^{y-ze}) = e$, then since $z = s \bmod q$ holds, we have $g^{(y - sd) - z(e+d)} = e \neq e + d$ for any integer $d \neq 0$. Hence the set $[0, 2^b) \times [A, A + \Delta)$ is separated into $(\Delta - s(2^b - 1))$ subsets each of which has exactly one pair (e, y) such that $F(g^{y-ze}) = e$, and $2s(2^b - 1)$

[1]Note that $2^{2t}|n| = |n|^{O(1)}$ follows from $t = O(\log |n|)$.

subsets each of which has at most one pair (e, y) with $F(g^{y-ze}) = e$. Therefore we can get $\Delta - \Phi \leq \mathcal{N}(F, A, \Delta) \leq \Delta + \Phi$ from $\Phi = (2^b - 1)(2^k - 1) \geq s(2^b - 1)$.

Denote, by $p(x, e, y)$, the probability to obtain the triplet (t, c, y) in the communication between P and $\widetilde{V}$, and denote, by $p'(x, e, y)$, the probability to obtain the triplet (t, c, y) by the simulator M. Let $e_i(I_i, x_i) = F(x_i)$. Then we have the following:

$$p(\alpha, \beta, \gamma) = \sum_{0 \leq r < 2^a} \mathtt{Pr} \left[\begin{array}{l} g^r \mathrm{mod} n = \alpha, \\ F(g^r) = \beta, \\ r + sF(g^r) = \gamma \end{array} \right] = \frac{1}{2^a} \chi \left(\begin{array}{l} 0 \leq \gamma - s\beta < 2^a, \\ F(\alpha) = \beta, \\ \alpha = g^{\gamma - z\beta} \mathrm{mod} n \end{array} \right),$$

where for a predicate $\wp$, $\chi(\wp)$ is the characteristic function of $\wp$, that is, $\chi(\wp) = 1$, if $\wp$ is true, and $\chi(\wp) = 0$, otherwise. If the triplet $(\alpha.\beta, \gamma)$ satisfies $\gamma \in [\Phi, 2^a)$, $F(\alpha) = \beta$ and $\alpha = g^{\gamma - z\beta} \mathrm{mod} n$, we have $p(\alpha, \beta, \gamma) = 1/2^a$. Hence the summation of the probability over $\alpha \in \mathbb{Z}_n^*, \beta \in [0, 2^b), \gamma \in [\Phi, 2^a)$ with $\alpha = g^{\gamma - z\beta} \mathrm{mod} n$, is equal to $\mathcal{N}(F, \Phi, 2^a - \Phi)/2^a$ by the definition of $\mathcal{N}$. On the other hand, as well as in [38], we have the following:

$$p'_i(\alpha, \beta, \gamma) = \chi \left(\begin{array}{l} \gamma \in [\Phi, 2^a), \\ F(\alpha) = \beta, \\ \alpha = g^{\gamma - z\beta} \mathrm{mod} n \end{array} \right) / \mathcal{N}(F, \Phi, 2^a - \Phi).$$

Therefore the summation of the differences, $\Sigma := \sum_{\alpha, \beta, \gamma} |p_i(\alpha, \beta, \gamma) - p'_i(\alpha, \beta, \gamma)|$, has a upper bound of $8q(2^b - 1)/2^a$, because $\Sigma = 2(1 - \mathcal{N}(F, \Phi, 2^a - \Phi)/2^a)$ holds similarly with [38], because $2^a - \Phi \leq \mathcal{N}(F, \Phi, 2^a - \Phi)$ holds, and because $\Phi := (2^b - 1)(2^k - 1) \leq (2^b - 1)2q$ follows from $2^{k-1} \leq q < 2^k$. If $q/2^a$ is negligible, then so is $8q(2^b - 1)/2^a$, and consequently, the output by $(P, \widetilde{V})$ and that by M are statistically indistinguishable. For a T-round identification by P and $\widetilde{V}$, we have the following:

$$\Sigma_T := \sum_{\alpha_i, \beta_i, \gamma_i, i \leq T} |\mathtt{Pr}\left[(x_i, e_i, y_i) = (\alpha_i, \beta_i, \gamma_i)\right] - \mathtt{Pr}\left[(x'_i, e'_i, y'_i) = (\alpha_i, \beta_i, \gamma_i)\right]| < 8(2^b - 1)\frac{qT}{2^a},$$

from Appendix A in [38], and if $qT/2^a$ is negligible, then so is Σ_T, and hence the proposed identification scheme has the statistical zero-knowledge property. □

6.3.2 Signature Scheme

As well as conventional identification schemes such as Schnorr [47] or Guillou-Quisquater [19], our identification scheme Scheme I can be turned into a signature scheme by using the technique in [12]. That is, the challenge in the identification is replaced with an appropriate hash function.

Key generation step: This is the same as our identification scheme.

Secret-key/public-key: This is the same as our identification scheme.

Signature generation step: Suppose a signer which has a public key and the corresponding secret key, generates the signature of her message $m \in \{0,1\}^*$. Then she executes the following steps:

1. Pick up a random number $r \in \mathbb{Z}_{2^a}$ and compute $x = g^r \bmod n$.
2. Compute $e = \mathcal{H}(x, m)$.
3. Compute $y = r + se$ in $\mathbb{Z}$.

Signature: The signature for a message m is (x, e, y).

Verification step: Suppose a verifier which has the signer's public-key and the corresponding message, checks the validity of the signature for m. Then she executes the following steps:

1. Check whether $y < 2^a + 2^{b+k}$ holds or not. If the equation does not hold, then reject the signature and stop this protocol.
2. Compute $e' = \mathcal{H}(x, m)$ and $x' = g^{y-ze} \bmod n$.
3. Check whether both $e = e'$ and $x = x'$ hold or not. If both equations hold, accept the signature. Otherwise reject it.

The performance in Scheme I is much superior to that in PS-scheme. Concrete to say, compared with PS-scheme, the size of a secret-key in Scheme I and a signature can be reduced by at least 69% and 47%, respectively. Furthermore, Scheme I has an advantage that the computational cost can also be smaller. Compared with PS-scheme, the computational cost in Scheme I for pre-computation, signature generation and verification can be reduced by at least 38%, 69%, and 64%, respectively.

Security Analysis

In Section 6.4.1, we have proved that our identification scheme is three-pass honest-verifier statistically zero-knowledge identification protocol. This result includes all the properties to apply the technique of *the forking lemma* in [35]. Therefore, we can say that, in the random oracle model [5, 35], our signature scheme is existentially unforgeable under the adaptive chosen message attack.

As a strategy, we actually show that if there exists a polynomial-time adversary which can existentially forge a signature in the proposed scheme, then we can construct a PPT machine which can compute the integer factoring problem.

Theorem 6.3.5 The signatures in the proposed scheme can be statistically simulated by a polynomial-time machine, if $2^b q / 2^a$ is negligible.

Proof. As follows, we can construct a polynomial-time simulator M. First M randomly picks up $e' \in [0, 2^b)$ and $y' \in [\Phi, 2^a)$. Then M computes $x' := g^{y'-ze'} \bmod n$, and outputs (x', e', y') as a valid signature. Denote, by $p(\alpha, \beta, \gamma)$ and $p'(\alpha, \beta, \gamma)$, the probabilities that (α, β, γ) is output by the signature algorithm and the simulator, respectively. Then

letting $\mathcal{R} : \{0,1\}^* \to \{0,1\}^b$ be an ideal hash function (random oracle), for a given message $m \in \{0,1\}^*$, we have the following:

$$p(\alpha,\beta,\gamma) = \frac{\chi\left(\begin{array}{l} g^{\gamma - z\beta} \bmod n = \alpha, \\ \mathcal{R}(\alpha, m) = \beta, \\ \gamma - s\beta \in [0, 2^a) \end{array}\right)}{2^a} \quad \text{and} \quad p'(\alpha,\beta,\gamma) = \frac{\chi\left(\begin{array}{l} g^{\gamma - z\beta} \bmod n = \alpha, \\ \mathcal{R}(\alpha, m) = \beta, \\ \gamma \in [\Phi, 2^a) \end{array}\right)}{\mathcal{N}(\mathcal{R}, \Phi, 2^a - \Phi)}.$$

Then like in Proposition 6.3.4, the summation $\Sigma := \sum_{\alpha,\beta,\gamma} |p(\alpha,\beta,\gamma) - p'(\alpha,\beta,\gamma)|$ has a lower bound of $8 \cdot 2^b q / 2^a$, which is negligible, if $2^b q / 2^a$ is negligible. □

Like [38, 39, 35], we first give the security proof against no-message adversaries. Then using Proposition 6.3.5, we can show that our signature scheme is *secure*.

Theorem 6.3.6 Assume that $1/2^b$ is negligible. Also assume that a polynomial-time no-message adversary $\mathcal{A}$ can ask $Q(k)$ queries to the random oracle, where $Q(k)$ is a polynomial on k. If $\mathcal{A}$ can forge a message m and one corresponding signature with probability $\varepsilon > 0$ within the time bound T, then we can construct a polynomial-time machine M which can factor n within $|n|^{O(1)} + (u(k) + v(k))T$, with probability $\varepsilon'/2^{2t}$, for two polynomials $u(k)$ and $v(k)$, where $\varepsilon' := \frac{1}{4}\left(1 - \mathsf{e}^{-\left(\varepsilon - \frac{1}{2^b}\right)u(k)}\right)\left(1 - \mathsf{e}^{-\left(\frac{\varepsilon}{4Q} - \left(\frac{1}{4Q}+1\right)\frac{1}{2^b}\right)v(k)}\right)$, where e is the base of natural logarithm. (Note that $\varepsilon'/2^{2t}$ is non-negligible, if and only if ε is non-negligible.)

Proof. Denote, by $\mathcal{Q}_i$ and ρ_i, the i-th query and the corresponding answer from the random oracle.

First M picks up a base $\widetilde{g} \in \mathbb{Z}_n^*$ and $\widetilde{z} \in [2^{k+\kappa}, 2^c)$. Then M gives $(n, \widetilde{g}, \widetilde{z})$ to $\mathcal{A}$, and makes $\mathcal{A}$ execute $u(k)$-time attacks with a random tape ω, where $u(k)$ is a polynomial on k. The probability that $\mathcal{A}$ succeeds to output m and (x, e, y), is at least $1 - \mathsf{e}^{-\left(\varepsilon - \frac{1}{2^b}\right)u(k)}$. Suppose that $\mathcal{A}$ succeeds for the i-th query. After that, M makes $\mathcal{A}$ execute more $v(k)$-time attacks with another random oracle which return the same answers, $\rho_1, \ldots, \rho_{i-1}$ for the first $(i-1)$ queries, $\mathcal{Q}_1, \ldots, \mathcal{Q}_{i-1}$. Then the probability that $\mathcal{A}$ succeeds for the i-th query to the second random oracle to output m and (x, e', y'), is at least $\varepsilon' := \frac{1}{4}\left(1 - \mathsf{e}^{-\left(\varepsilon - \frac{1}{2^b}\right)u(k)}\right)\left(1 - \mathsf{e}^{-\left(\frac{\varepsilon}{4Q} - \left(\frac{1}{4Q}+1\right)\frac{1}{2^b}\right)v(k)}\right)$.

If the order of $\widetilde{g}$ is $\lambda(n)$, then M can get a multiple of $\lambda(n)$ by computing $\widetilde{z} - (y - y')/(e - e')$. That means M can figure out a multiple of $\lambda(n)$ if M chooses an order-$\lambda(n)$ base $\widetilde{g}$, and if $\mathcal{A}$ outputs two signatures. Hence the probability that M can find a multiple of $\lambda(n)$ is at least $\varepsilon'/2^{2t}$, since denoting, by G and A, the events that the order of $\widetilde{g}$ is $\lambda(n)$ and that $\mathcal{A}$ outputs two valid signatures (x, e, y) and (x, e', y'), we have the following:

$$\mathtt{Pr}\left[G \,\&\, A\right] = \mathtt{Pr}\left[A|G\right] \cdot \mathtt{Pr}\left[G\right] = \mathtt{Pr}\left[G\right] \cdot \mathtt{Pr}\left[A\right] = \frac{1}{2^{2t}}\varepsilon'.$$

When the order of $\widetilde{g}$ is $\lambda(n)$, M can get a multiple of $\lambda(n)$ and hence the factorization of n by Lemma 6.2.2. □

Theorem 6.3.7 Assume that $2^b q/2^a$ and $1/2^b$ are negligible. Also assume that a polynomial-time adversary $\mathcal{A}$ can forge a signature with non-negligible probability by executing adaptive chosen-message attacks. Then we can construct a polynomial-time machine M which can factor n with non-negligible probability.

Proof. From the assumption that $2^b q/2^a$ is negligible, by Proposition 6.3.5, the signature oracle can be simulated by a polynomial-time machine. The proof is almost the same with the previous one except for that M takes not only a role of the the signing oracle but also a role of the signing oracle.

Unlike the previous proposition, we should consider the collision among (x, e)'s. For $\alpha \in \mathbb{Z}_n^*$, the probability that the commitment x is equal to α is at most $1/q + 1/2^a$. Suppose that $\mathcal{A}$ can ask $Q(k)$ queries to the random oracle and $R(k)$ queries to the signing oracle. Then such a collision occurs with probability at most $(Q(k)R(k) + R(k)^2)(1/q + 1/2^a)$, which is negligible, since we may assume $1/2^a$ is negligible, and since also $1/q$ is negligible from $2^{k-1} \leq q < 2^k$. Denote, by A and C, the events that $\mathcal{A}$ succeeds to output two valid signatures (x, e, y) and (x, e', y') for a message m and that a collision among the commitments occurs, respectively. Then since $\mathtt{Pr}[A \,\&\, \neg C] \geq \mathtt{Pr}[A] - \mathtt{Pr}[C]$, the probability that M can obtain such two valid signatures without occurring any collisions, is non-negligible, since $\mathtt{Pr}[A]$ is non-negligible, and since $\mathtt{Pr}[C]$ is negligible. Therefore M can factor n with non-negligible probability. □

6.3.3 Parameter Generation

In this section, we describe the conditions of the parameters to keep the security and the corresponding signature scheme. We also show that how to implement the parameters.

Parameter a and b: In case of signature $y = r + se$, with $|r| = a$, $|s| = k$ and $|e| = b$, the values of a, b, k, κ shall satisfy $a \geq b + k + \kappa$ for its security.

Parameter c: If $y > ze$ were allowed, then an adversary could impersonate the signer to easily compute y, along with the actual protocol, such that $x = g^{y-ze} \bmod n$ holds. To keep off such an attack, the condition of $c \geq k + 2\kappa$ shall be required from $c + b \geq a + \kappa \geq b + k + 2\kappa$. Furthermore, if $q > 2^c$ were satisfied, then $s = z$ would hold, that is, the secret-key would be disclosed. Hence also $q \leq 2^{c-\kappa}$ shall be required, and it is always held since $q < 2^k \leq 2^{c-2\kappa} \leq 2^{c-\kappa}$.

Parameter κ: The information leak parameter κ should be set up so that 2^κ-time computation should be intractable.

Parameter q: If an adversary could figure out $r \in \mathbb{Z}_q^*$ from x $(= g^r \bmod n)$ generated by the actual signer, then she could break the signature scheme. We can see the algorithms to extract r, such as *Pollard lambda method* in [36] and *the baby-step giant-step method* in [22]. One may say that the former is better than the latter since it has same computational complexity (exponential-time: $O(\sqrt{q})$) but does not need memory. The size of q shall be set up not so that r can be figured out with such an algorithm.

Choice of p, q and g: We describe how to find p, q and an asymmetric basis g in $\mathbb{Z}_n^*$.

1. Pick up two primes $p = 2p'p'' + 1$ and $q = 2q'q'' + 1$ such that p' and q' are also primes, and p'' and q'' are odd numbers.
2. Choose $\alpha_p \in \mathbb{Z}_p^*$ satisfying $g_p = \alpha_p^{(p-1)/p'} \neq 1 \bmod p$. In the same way, choose $\alpha_q \in \mathbb{Z}_q^*$ satisfying $\alpha_q \neq q - 1 \bmod q$, $\alpha_q^{(q-1)/2} \neq 1 \bmod q$ and $g_q = \alpha_q^{(q-1)/2q'} \neq 1 \bmod q$.
3. Compute $n = pq$ and $g = q(q^{-1} \bmod p)g_p + p(p^{-1} \bmod q)g_q \bmod n$.

In Step3, g is computed by using the technique of Chinese Reminder Theorem (CRT). Note that $\mathrm{Ord}_p(g) = p'$ and $\mathrm{Ord}_q(g) = 2q'$. Therefore $\mathrm{Ord}_n(g) = \mathrm{lcm}(p', 2q') = 2p'q'$.

Choice of $\mathcal{H}$: Next, we discuss secure hash algorithm which we should adopt. If $\mathcal{H}$ were an *ideal* hash function, then the proposed signature scheme would be *secure* under the meaning of the description in Section 6.4.2. Since such a random function does not exist in the real world, in implementation, we are recommended to adopt MD5 by [41] or SHA-1 by [30], each of which is designed so that the algorithm can be a collision intractable hash function [8].

6.3.4 Optimized scheme

Let us consider the optimized scheme of data size in Scheme II. We now focus on the following two parts:

Elimination of x: In the same way in Section 6.2.4, the parameter x can be eliminated for the efficiency. Consequently, the signature for m consists of (e, y).

Elimination of z: The size of public key is optimized as follows. We regard actual public-key as (n, g), and z is computed by $z = \mathcal{H}'(n, g)$, where $\mathcal{H}'$ is a hash function $\mathcal{H}' : \{0,1\}^* \to \{0,1\}^c$.

We give optimized scheme obtained by the above technique. Let $\mathcal{H} : \{0,1\}^* \to \{0,1\}^c$ and $\mathcal{H} : \{0,1\}^* \to \{0,1\}^b$ be two hash functions.

Key generation step: The following steps are executed:

1. Pick up two same-size primes p and q, and compute $n = pq$.
2. Choose an element $g \in \mathbb{Z}_n^*$ which is an asymmetric basis in Z_n^*.
3. Computes $z = \mathcal{F}(n, g)$ and $s = z \bmod q$, where $\mathrm{Ord}_n(g) = q$.

Secret-key/public-key: The secret-key is s and the corresponding public-key is (n, g).

Signature generation step: A singer executes the following steps:

1. Pick up a random number $r \in \mathbb{Z}_{2^a}$ and compute $x = g^r \bmod n$.
2. Compute $e = \mathcal{H}(x, m)$.
3. Compute $y = r + se$ in $\mathbb{Z}$.

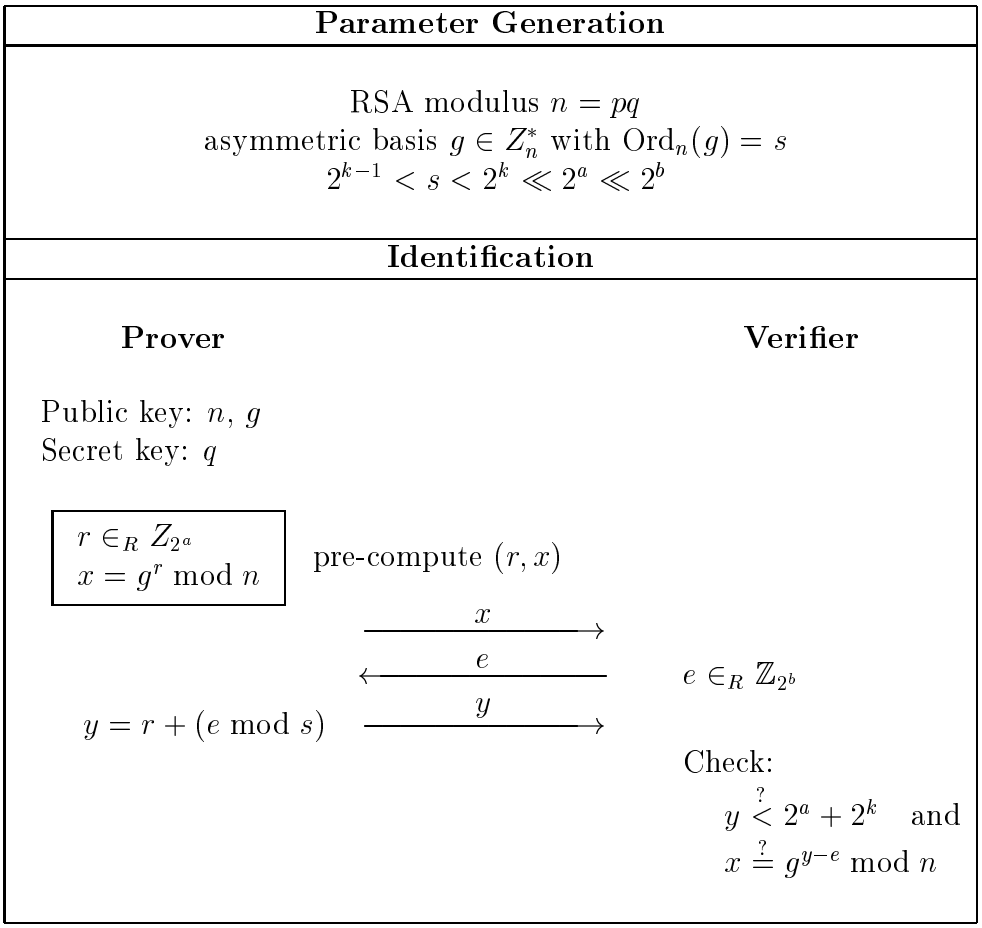

Figure 6.2: Proposed identification scheme: Scheme II

Signature: The signature for a message m is (e, y).

Verification step: A verifier executes the following steps:

1. Check whether $y < 2^a + 2^{b+k}$ holds or not. If the equation does not hold, then reject the signature and stop this protocol.
2. Compute $z' = \mathcal{G}(n, g)$ and $e' = \mathcal{F}(g^{y-z'e} \bmod n, m)$.
3. Check whether $e' = e$ holds or not. If the equation holds, accept the signature. Otherwise reject it.

6.4 Signature Without on-line Multiplication

6.4.1 Identification Scheme

Our identification scheme uses the parameters, k, s, a and b with $2^{k-1} \leq s < 2^k \ll 2^a \ll 2^b$. For more details, we refer to Section 6.4.3. We now give our identification protocol.

Key generation step: The following steps are executed:

1. Pick up two same-size primes p and q, and compute $n = pq$.
2. Choose an element $g \in \mathbb{Z}_n^*$ which is an asymmetric basis in Z_n^*, where $\mathrm{Ord}_n(g) = s$.

Secret-key/public-key: The secret-key is s and the corresponding public-key is (n, g).

Identification step: A prover and a verifier execute the following steps:

1. The prover picks up a random number $r \in \mathbb{Z}_{2^a}$, computes the commitment $x = g^r \bmod n$ and sends x to the verifier.
2. The verifier picks up a random challenge $e \in \mathbb{Z}_{2^b}$ and sends e to the prover.
3. The prover computes $z = e \bmod s$ and an answer $y = r + z$ in $\mathbb{Z}$, and sends y to the verifier.
4. The verifier checks whether both $y < 2^a + 2^k$ and $x = g^{y-e} \bmod n$ hold or not. If both equations hold, the verifier accepts. Otherwise she rejects.

In Step3, the on-line multiplication for the prover is eliminated. This is the main idea of our scheme.

Note that, in the conventional identification schemes such as Schnorr [47] or Guillou-Quisquater [19], the challenge e can be a fixed constant. In the following, such schemes have some rounds. However, in our identification, e has the condition such that $2^k \ll e$. Therefore, the round of our identification is constant and fixed one. We can say such a property indeed characterizes our schemes.

Security Analysis

We show that Scheme II is a three-pass honest-verifier statically zero-knowledge identification protocol.

As well as section 6.2.1, we show that Scheme II provides completeness, soundness and the honest verifier statistical zero-knowledge property.

Theorem 6.4.1 (Completeness) In the proposed identification scheme, the prover who has a secret-key, and who follows the scheme, is always successfully accepted.

Proof. The answer y correctly computed by the prover, is $r + z$, where $z = \bmod s$. Since $r < 2^a$ and $z < q < 2^k$, those conditions satisfy $y < 2^a + 2^k$. Furthermore, from the following equations:

$$g^{y-e} = g^{(r+z)-e} = g^{r+(z-e \bmod s)} = g^r = x \bmod n,$$

a correctly generated answer y always passes the verification. □

Theorem 6.4.2 (Soundness) If there exists a polynomial-time adversary $\tilde{P}$ which is accepted by honest verifiers with probability $> 1/2^b + \varepsilon, \varepsilon > 0$, where the average running time is T. Then, by using $\tilde{P}$, we can construct a polynomial-time machine M, which can figure out the factorization of public-key n, in expected time $O(T/\varepsilon + |n|^{O(1)})$.

Proof. We show to construct M on input a random tape RT and a public key (n, g) outputs L such that $g^L = 1 \bmod n$.

1. Pick up RT at random, input RT to $\tilde{P}$ and obtain a value x from $\tilde{P}$.
2. Pick up a random number $e \in \mathbb{Z}_{2^b}$, input e to $\tilde{P}$ and obtain a value y from $\tilde{P}$.
3. Check whether the above parameters (x, e, y) are, in verification, valid or not. If they are valid, fix the RT, store (x, e, y), and go to next step. Otherwise stop this protocol.
4. Check whether M obtains same commitment x and different parameters e, y such as (x, e, y) and (x, e', y'). If such parameters are obtained, output $L = (y - y') - (e - e') > 1$. Otherwise stop this protocol.

As for the above protocol, if we repeat kT/ε times, M can obtain L with non-negligible provability, by the results of [32, 35].

Here $L > 1$ is a multiple of $\mathrm{Ord}_n(g)$, that is, $g^L = 1 \bmod n$, and g is an asymmetric basis. Therefore, as in the consequence of Theorem 6.2.2, M can obtain the factorization of public in expected time $O(kT/\varepsilon + |n|^{O(1)})$. □

Theorem 6.4.3 (Zero-knowledge property) The proposed identification scheme provides honest-verifier statistical zero-knowledge property, if $2s/2^a$ is negligible.

Proof. We prove the above theorem along the line of Theorem 6 in [38]. To prove the theorem, we show that we can construct a polynomial-time machine (simulator) M which simulates the communication between the honest prover and a honest verifier.

In this case, M executes the following steps:

1. Pick up two random numbers: $e' \in \mathbb{Z}_{2^b}$ and $y' \in \mathbb{Z}_{2^a}$.
2. Compute $x' = g^{y'-e'} \bmod n$.
3. Output (x', e', y').

We denote by π, the probability in the communication between a honest prover and a honest verifier, that is, $\mathtt{Pr}\left[(x, e, y) = (\alpha, \beta, \gamma)\right]$. Then we have the following:

$$
\begin{aligned}
\pi &= \mathtt{Pr}\left[\begin{array}{c} \alpha = g^r \bmod n, \\ \beta = e, \\ \gamma = r + \Omega \end{array}\right] \\
&= \frac{1}{2^{a+b}} \cdot \sum_{\substack{0 \le r < 2^a \\ 0 \le e < 2^b}} \mathtt{Pr}\left[\begin{array}{c} \alpha = g^{\gamma - \Omega'} \bmod n, \\ \beta = e, \\ r = \gamma - \Omega' \end{array}\right] \\
&= \frac{1}{2^{a+b}} \cdot \mathtt{Pr}\left[\begin{array}{c} \alpha = g^{\gamma-\beta} \bmod n, \\ 0 \le \beta < 2^b, \\ 0 \le \gamma - \Omega' < 2^a \end{array}\right] \\
&= \frac{1}{2^{a+b}} \cdot \chi(\alpha = g^{\gamma-\beta} \bmod n) \\
&\cdot \chi(0 \le \beta < 2^b) \cdot \chi(\Omega' \le \gamma < 2^a + \Omega'),
\end{aligned}
$$

where $(e \bmod q)$ and $(\beta \bmod q)$ are denoted by Ω and Ω', respectively. For a predicate Q, $\chi(Q)$ means the characteristic function of Q, that is, if Q is true, then $\chi(Q) = 1$, otherwise $\chi(Q) = 0$.

In the same way as above, the probability by M, that is, $\mathtt{Pr}\left[(x', e', y') = (\alpha, \beta, \gamma)\right]$ is denoted by π'. In this case, we have the following:

$$\begin{aligned}
\pi' &= \mathtt{Pr}\left[\begin{array}{c} \alpha = g^{y'-e'} \bmod n, \\ \beta = e', \\ \gamma = y' \end{array}\right] \\
&= \frac{1}{2^{a+b}} \cdot \sum_{\substack{0 \le r < 2^a \\ 0 \le e < 2^b}} \mathtt{Pr}\left[\begin{array}{c} \alpha = g^{\gamma-\beta} \bmod n, \\ \beta = e', \\ \gamma = y' \end{array}\right] \\
&= \frac{1}{2^{a+b}} \cdot \chi(\alpha = g^{\gamma-\beta} \bmod n) \\
&\cdot \chi(0 \le \beta < 2^b) \cdot \chi(0 \le \gamma < 2^a),
\end{aligned}$$

Next, we will estimate the distance between actual system and M. We first define $\Delta = \sum_{\alpha,\ \beta,\ \gamma} \|\pi - \pi'\|$, where $\|x\|$ denotes the absolute value of x. From the results of π and π', we obtain

$$\Delta = \sum_{\substack{\alpha = g^{\gamma-\beta} \bmod n \\ 0 < \beta < 2^b \\ 0 \le \gamma < \Omega'}} \pi' + \sum_{\substack{\alpha \\ \beta \\ \Omega' \le \gamma < 2^a}} \|\pi - \pi'\| + \sum_{\substack{\alpha \\ \beta \\ 2^a \le \gamma < 2^a + \Omega'}} \pi.$$

Since

$$\left(\sum_{\substack{\alpha,\ \beta \\ 0 \le \gamma < \Omega'}} \pi'\right), \left(\sum_{\substack{\alpha,\ \beta \\ 2^a \le \gamma < 2^a + \Omega'}} \pi\right) \le \frac{\Omega'}{2^a} < \frac{q}{2^a}$$

and

$$\sum_{\substack{\alpha,\ \beta \\ \Omega' \le \gamma < 2^a}} \|\pi - \pi'\| = 0,$$

we can conclude $\Delta < 2q/2^a$. This proves the theorem. □

6.4.2 Signature Scheme

Let $\mathcal{H} : \{0,1\}^* \to \{0,1\}^b$ be a hash function.

Key generation step: This is the same as our identification scheme.

Secret-key/public-key: This is the same as our identification scheme.

Signature generation step: Suppose a signer which has a public-key and the corresponding secret-key, generates the signature of her message $m \in \{0,1\}^*$. Then she executes the following steps:

1. Pick up a random number $r \in \mathbb{Z}_{2^a}$ and compute $x = g^r \bmod n$.
2. Compute $e = \mathcal{H}(x, m)$.
3. Compute $z = e \bmod s$ and $y = r + z$ in $\mathbb{Z}$.

Signature: The signature for a message m is (x, e, y).

Verification step: Suppose a verifier which has the signer's public-key and the corresponding message, checks the validity of the signature for m. Then she executes the following steps:

1. Check whether $y < 2^a + 2^k$ holds or not. If the equation does not hold, then reject the signature and stop this protocol.
2. Compute $e' = \mathcal{H}(x, m)$ and $x' = g^{y-e} \bmod n$.
3. Check whether both $e = e'$ and $x = x'$ hold or not. If both equations hold, accept the signature. Otherwise reject it.

We now give the concrete evaluation measured by comparing the integer factoring based scheme, OTM-scheme (resp. PS-scheme) and our scheme. Compared with OTM-scheme, our scheme enables the computational cost to be reduced by 32% for verification. For the data size, the signature size is reduced by 21%. In the following, compared with PS-scheme, our scheme enables the computational cost to be reduced by 83% for pre-computation and by 78% for verification. For the data size, the secret-key size is reduced by 68% and the signature size is 58%.

Security Analysis

In Section 6.4.1, we have proved that our identification scheme is three-pass honest-verifier statistically zero-knowledge identification protocol. This result includes all the properties to apply the technique of *the forking lemma* in [35]. Therefore, we can say that, in the random oracle model [5, 35], our signature scheme is existentially unforgeable under the adaptive chosen message attack.

Theorem 6.4.4 Let Q be the number of queries which a polynomial-time adversary $\mathcal{A}$ executing the adaptive chosen-message attack, can ask to the random oracle, and let R be the number of queries which $\mathcal{A}$ can ask to the actual signer. Assume that $2s/2^a$ is negligible. Also assume that, $\mathcal{A}$ can forge a signature with non-negligible probability $\varepsilon \geq 10(R+1)(R+q)/2^b$, where the average running time of $\mathcal{A}$ is T. Then we can construct a polynomial-time machine M which can factor n with non-negligible probability in expected time $O(QT/\varepsilon + |n|^{O(1)})$.

Proof. By the result of Theorem 6.4.3, the signature oracle in our signature scheme can be statistically simulated by a probabilistic polynomial-time machine M which works according to the protocol like [35].

When M uses $\mathcal{A}$, she can obtain two distinct signatures (x, e, y) and (x', e', y') such that $x = x'$, but $e \neq e'$. Then, we can get a multiple of $\mathrm{Ord}_n(g)$ such that $L > 1$ and $g^L = 1 \bmod n$. Here g is an asymmetric basis in $\mathbb{Z}_n^*$, therefore by the result of Lemma 6.2.2 we can get a factor of n in expected time $O(QT/\varepsilon + |n|^{O(1)})$. □

6.4.3 Parameter Generation

We describe remarks on the parameters for the security of Scheme II.

Parameters a and b: For the security reason, the values of a and b shall satisfy: $a = k + \kappa_1$ and $b = a + \kappa_2$, where k is a security parameter satisfying $|s| = k$ with $\mathrm{Ord}_n(g) = s$, and where both κ_1 and κ_2 are information leak parameters.

Parameters κ_1 and κ_2: By Theorem 6.4.3, we must set such that $1/2^{\kappa_1}$ is intractable. As for κ_2, let us first consider the following problem: given (n, g, α, κ_2), find β, where $g^\alpha = g^\beta \bmod n$ and where $|\beta|$ is at least κ_2-bits smaller than $|\alpha|$. In our schemes, we must set κ_2 on the condition that the problem is hard to solve for the security parameter k. For implementation, we should take κ_1 and κ_2 greater than 64 and 24 bits, respectively.

Parameter s: Let us consider the attack which an adversary computes s only from the information of the public-key (n, g). We can see the algorithms to extract s, such as *Pollard lambda method* in [36] and *the baby-step giant-step method* in [23]. One may say that the former is better than the latter since it has same computational complexity (exponential-time: $O(\sqrt{s})$) but does not need large memory. The size of s shall be set up such that the above algorithms cannot apply for the security parameter k. For implementation, we should take $|s| = k$ greater than 160 bits for the security reason.

6.4.4 Optimized Scheme

As for the signature scheme in Scheme II, we can diminish the size of the signature. Consequently, communication lord is more efficient than before. We now focus on the following two parts:

Elimination of x: When we have two parameters e and y, the parameter x can be generated by computing $x = g^{y-e} \bmod n$. Therefore, the signature x is eliminated like conventional generic signature schemes such as Schnorr [47] or Guillou-Quisquater [19]. In this case, the signature for m consists of (e, y).

Using a short-size e: As for the signature scheme in Scheme II, signature e is, in a certain sense, verbose. In our signature scheme, large-size e such as $2^a \ll e$ is actually

needed in verification. Hence, we can use the following technique: we regard short-size e satisfying $2^k \ll e \ll 2^b$ as signature, and in verification, we extend the size of e by using appropriate hash function.

We use two hash functions $\mathcal{F} : \{0,1\}^* \to \{0,1\}^c$ and $\mathcal{G} : \{0,1\}^c \to \{0,1\}^b$, where c has the condition $2^c \ll 2^b$. The optimized signature scheme, which apply the above techniques, are as follows:

Key generation step: This is the same as our identification scheme.

Secret-key/public-key: This is the same as our identification scheme.

Signature generation step: A singer executes the following steps:

1. Pick up a random number $r \in \mathbb{Z}_{2^a}$ and compute $x = g^r \bmod n$.
2. Compute $e = \mathcal{F}(x, m)$.
3. Compute $\epsilon = \mathcal{G}(e)$, $z = \epsilon \bmod s$ and $y = r + z$ in $\mathbb{Z}$.

Signature: The signature for a message m is (e, y).

Verification step: A verifier executes the following steps:

1. Check whether $y < 2^a + 2^k$ holds or not. If the equation does not hold, then reject the signature and stop this protocol.
2. Compute $\epsilon' = \mathcal{G}(e)$ and $e' = \mathcal{F}(g^{y-\epsilon'} \bmod n, m)$.
3. Check whether $e' = e$ holds or not. If the equation holds, accept the signature. Otherwise reject it.

As for the hash function $\mathcal{G}$, we should take c greater than 160 bits for implementation.

6.5 Further Discussion

In this section, we analyze the security of our schemes (Scheme I and II) in the case that n is built from more than three primes. We have seen that our schemes are provably as secure as the integer factoring problem for RSA modulus n if (1) public element g is an asymmetric basis in $\mathbb{Z}_n^*$ and (2) public key n is a RSA modulus. We now consider the case such that the number of factors of n is more than three. Then there exists some practical reasons to use more than three prime factors of n.

- In response to the small primes, key generation takes less time.
- The pre-computation for the signer take less time if one uses the Chinese Remainder Theorem. For example, using three primes vs. two primes theoretically gives a speedup of 9/4.

The reducibility is not known at present such that breaking our schemes with multi-prime modulus reduce to integer factoring problem. Therefore, we use the extended finding order problem given in Section 5.3 for the security.

Let Q be the number of queries which a polynomial-time adversary $\mathcal{A}$ (adaptive chosen-message attacker) can ask to the random oracle. Let R be the number of queries which $\mathcal{A}$ can ask to the the actual signer.

Theorem 6.5.1 Assume that $2^b q/2^a$ and $1/2^b$, in Scheme I with multi-prime modulus, are negligible. Also assume that $\mathcal{A}$ can forge a signature with non-negligible probability $\varepsilon \geq 10(R+1)(R+q)/2^b$, and with the average running time T. Then we can construct a polynomial-time machine M which can solve the extended finding order problem for public key (n, g) with non-negligible probability in expected time $O(QT/\varepsilon)$.

Proof. (Sketch) We firstly show that the signatures in Scheme I can be statistically simulated by a polynomial-time machine. This machine is simulated according to the protocol like in [35].

We denote, by $p(\alpha, \beta, \gamma)$ and $p'(\alpha, \beta, \gamma)$, the probabilities that (α, β, γ) is output by the signature algorithm and the simulator, respectively. We set $\phi = (2^b - 1)(2^k - 1)$, and let $\mathcal{R} : \{0,1\}^* \to \{0,1\}^b$ be an ideal hash function (random oracle) for a given message $m \in \{0,1\}^*$. For an integer A and a positive constant Δ, $\mathcal{N}(\mathcal{R}, A, \Delta)$ is defined to be the number of pairs $(e, y) \in [0, 2^b) \times [A, A + \Delta)$ such that $\mathcal{R}(g^{y-ze}, m) = e$. Then we have the following:

$$p(\alpha, \beta, \gamma) = \frac{\chi\left(\begin{array}{l} g^{\gamma - z\beta} \bmod n = \alpha, \\ \mathcal{R}(\alpha, m) = \beta, \\ \gamma - s\beta \in [0, 2^a) \end{array}\right)}{2^a}$$

and

$$p'(\alpha, \beta, \gamma) = \frac{\chi\left(\begin{array}{l} g^{\gamma - z\beta} \bmod n = \alpha, \\ \mathcal{R}(\alpha, m) = \beta, \\ \gamma \in [\phi, 2^a) \end{array}\right)}{\mathcal{N}(\mathcal{R}, \phi, 2^a - \phi)},$$

where for a predicate $\wp$, $\chi(\wp)$ is the characteristic function of $\wp$, that is, $\chi(\wp) = 1$, if $\wp$ is true, and $\chi(\wp) = 0$, otherwise.

Therefore, the summation $\Sigma = \sum_{\alpha,\beta,\gamma} |p(\alpha, \beta, \gamma) - p'(\alpha, \beta, \gamma)|$, has a upper bound of $8q(2^b - 1)/2^a$, because $\Sigma = 2(1 - \mathcal{N}(\mathcal{R}, \phi, 2^a - \phi)/2^a)$ holds similarly with [38], because $2^a - \Phi \leq \mathcal{N}(\mathcal{R}, \phi, 2^a - \phi)$ holds, and because $\phi = (2^b - 1)(2^k - 1) \leq (2^b - 1)2q$ follows from $2^{k-1} \leq q < 2^k$. If $q/2^a$ is negligible, then so is $8q(2^b - 1)/2^a$, and consequently, the output by real signer and that by the simulator are statistically indistinguishable.

Next, by using the technique in [35], we can get a multiple of $\mathrm{Ord}_n(g)$, L, such that $g^L = 1 \bmod n$. This means that we obtain the extended finding order's answer L for

(n, e). Hence this proves the theorem. □

It is obvious that the same result can be obtained for Scheme II.

We now consider the secure size of n, and also discuss secure number of the prime factors for n in our schemes.

Of course, if the modulus n were factored, then the proposed signature schemes would be broken. In [25], we can see *the number field sieve method* for factorization, which is the most efficient algorithm ever proposed, and whose running time depends upon the size of n. On the other hand, in [26], we can see *the elliptic curve method*, which is also one of efficient algorithms for factorization, and whose running time depends upon the size of factors of n. Therefore, the faster one is determined according to the size of the input and upon the number of the factors of n.

Referring to [50] for computational cost of algorithms, in case that $|n| = 1024$ and that n has three prime factors, the number field sieve method is faster, whereas in case n has four prime factors, the other is faster. Hence we recommend $|n| = 1024$ and the number of n is three for the security reason.

6.6 Performance

We evaluate the efficiency of our signature scheme by comparing existing on the fly signatures. Table 6.1 gives the performance of various schemes, such as OTM-scheme, PS-scheme and GPS-scheme, including ours.

The parameters in the schemes are set up as follows.

- Scheme I has $a = 224$ and $b = 248$ by taking $k = 160$, $\kappa_1 = 64$ and $\kappa_2 = 24$. This scheme is also use the technique of Section 6.4.4.
- Scheme II has $a = 104$, $b = 80$ and $c = 288$ by taking $k = 160$. To set up the same condition as our scheme, in this scheme, n is a RSA modulus and g is an asymmetric basis in $\mathbb{Z}_n^*$. Furthermore, the size of public-key is optimized as follows.
- PS-scheme has $|A| = 656$ and $|B| = 80$ by taking $k = 513$.
- GPS-scheme has $|A| = 1184$ and $|B| = 80$ by taking $k = 1024$.

Additionally, we set $|n| = 1024$ for all schemes.

Next, we show the comparison between OTM-scheme (resp. PS-scheme and resp. GPS-scheme) and our signatures.

Scheme I: We consider the computational efficiency between the multiplication in Scheme I and the modular reduction in the other on the fly schemes. In the multiplication, a recursive algorithm due to [20] reduces the complexity of the multiplying. On the other hand, in the modular reduction, we can use the efficient technique such as [3, 36].

Those methods are further advantageous than [20] because a *single modulus* can be used in our schemes and many reductions are performed by using such a modulus. That is, in Scheme I the modulus s is fixed because s is a secret key. This property lead to a good computational efficiency for modular reduction. On the other hand, such an efficiency does not exist for the multiplication in on the fly signatures.

Consequently, the on-line modular reduction in Scheme I is faster than the on-line multiplication in on the fly signatures from implementation point of view.

Scheme II: One of the verifications in OTM-scheme is $x = g^{y-ze} \bmod n$, where in the index of g, we can see the multiplication of two parameters z and e. On the contrary, the verification in our scheme is $x = g^{y-e} \bmod n$, hence the multiplication in the index does not exist. The large-size index involved by the multiplication, lead to the inefficiency from both the amount of work and data size point of view. Nowadays, the existing on the fly signatures have the same drawbacks.

Consequently, `CVF` and `SSig` in our scheme is superior to those in OTM-scheme. Since the public key in our scheme and that in OTM-scheme are (n, g) and (n, g, z), respectively, the number of the parameters in our scheme is smaller than that in OTM-scheme.

PS-scheme: The size of secret key in PS-scheme is only dependent on the modulus n, and that is considerably large (about $|n|/2$). This drawback leads to inefficient results with respect to the computational work (`CPC`, `CSG` and `CVF`) and the data size (`SSK` and `SSig`).

On the other hand, since PS-scheme is intended to be used with a modulus product of two strong primes, $g = 2$ is a correct basis and do not have to be included in the public key. Consequently, we can set $\texttt{SPC} = 1024$ for PS-scheme. Therefore, one may say that PS-scheme is more efficient than our scheme in terms of size of public key.

GPS-scheme: Since the public key in GPS-scheme consists of three parameters such as (n, g, v), the size of the public key is $\texttt{SPK} = 3072$. Hence, GPS-scheme has the largest size for public-key of all the schemes in Table 6.1.

Note that, in the table, all the schemes are based on the one-key attack scenario. Consequently, GPS-scheme has provable security such that the scheme is as secure as the discrete problem for modulo n. However, the size of secret-key in GPS-scheme is considerably large: $\texttt{SSK} = 1024$. In the same reason as in PS-scheme, this result leads to the inefficiency mentioned above.

Table 6.1 show that our signature scheme is quite efficient from both the computational cost and the data size point of view.

6.7 Conclusion

We have proposed two fast on-line signature schemes, which are derived from a three-pass identification scheme.

Scheme I is a signature scheme without on-line modular reduction, which is called on the fly signature. whereas Scheme II is one without on-line multiplication. Therefore Scheme II is, in a sense, a counterpart for on the fly signature schemes.

Compared between two types of signatures, the on-line computation of Scheme II take less time than that of Scheme I, because modular reduction is faster than multiplication from implementation point of view.

We have obtained the results that under such an attack, if we set that n is an RSA modulus and g is an asymmetric basis, our schemes are as secure as integer factoring. On the other hand, if we set that n consists of three or more prime factors of n, our schemes are equivalent to the extended finding order problem.

It is not unknown at present whether our schemes is as secure as integer factoring problem in the case of multi-prime modulus. This remains open problem.

Table 6.1: Performance of signature schemes.

Scheme	UMP	CPC ($\times M$)	CSG	CVF ($\times M$)	SPK (bits)	SSK (bits)	SSig (bits)
Scheme I	Integer factoring with g	61	248 mod 160	372	2048	160	304
Scheme II	Integer factoring with g	61	80×160	552	2048	160	384
PS-scheme	Integer factoring	381	80×512	1656	1024	513	736
GPS-scheme	Discrete log. for modulo n	381	80×1024	1796	3072	1024	1264

Abbreviation:

- UMP means the underlying mathematical problem that the signature scheme relies on for its security.
- CPC, CSG and CVF mean the computational cost for pre-computation, signature generation and verification, respectively
- SPK, SSK and SSig means the size of a public-key, a secret-key and a signature, respectively.

- M represents the computational cost for one multiplication under a 1024-bit modulus.
- α mod β represents the computational cost for modular reduction of an α-bit number and a β-bit number modulus.
- $\gamma \times \delta$ represents the computational cost for multiplication of an γ-bit number and a δ-bit number on $\mathbb{Z}$.

Notes:

- For all schemes in the table, we set up the parameter under the line of the one-key attack scenario [39].
- For respective computational cost, a primitive arithmetic of binary methods [22] are used, e.g. amount of work for $g^\alpha \bmod n$ is $\frac{3}{2}|\alpha|M$ if $|n| = 1024$. Of course there exist more sophisticated techniques which reduce the amount of computational work. However we think they estimate the concrete performance without loss of generosity.
- In UMP, integer factoring with g means that it is a variant of integer factoring problem on input RSA modulus n and the asymmetric basis g, outputs the factor of n.
- In CPC, the signer uses the technique of CRT. In this case, the signer must secretly have the factors of n, p and q.

Chapter 7

Proxy signatures with message recovery

7.1 Introduction

In this chapter, we propose two new proxy signature schemes, DLP-PS and RSA-PS, in which the original signer can control signing power of a proxy signer to some extent. DLP-PS, which is given by improving MUO-scheme and KPW-scheme, controls the power of a proxy signer by adding usage condition to proxy public key implicitly, which uses idea of message recovery. In verifying proxy signer's signature, the usage condition can be checked independent of a message. RSA-PS controls the power of a proxy signer by adding usage condition to proxy public key explicitly, which uses idea of ID-based cryptosystem [48]. Up to the present, any proxy signature [21, 24, 53, 27] is based on Discrete logarithm problem. Therefore RSA-PS is a new type scheme, which is based on RSA.

7.2 Our Basic Idea

We describe the basic idea to solve the problem stated in Section 4.5.

- **Proposed Proxy Signature.** This is a variant system of the original proxy signature defined in Section 4.5. In our schemes there exists the following property: In verification, the verifier accepts it if and only if she recovers the usage condition created by the *original signer*. Otherwise she rejects it. Therefore, in our schemes, the original signer can control the signer's power. For keeping the security, it must be infeasible for the proxy signer to forge the signature such that she switches the original signer's usage condition to the proxy signer's one.

We state our system more concretely. In verification, the verifier can check two messages at once: One is a usage condition $\mathcal{M}_p^{uc}$ and the other is a message m_p which the proxy signer chooses on behalf of the original signer. Note that those messages are guaranteed by *distinct* entities, respectively: An original signer signs $\mathcal{M}_p^{uc}$ by using a manner of

Table 7.1: Comparison of the original and the proposed proxy signatures.

Scheme	Underlying Problem	Proxy signer's power	Type of usage condition	Type of proxy signature
MUO-scheme	Discrete log.	not restricted	none	any
KPW-scheme	Discrete log.	restricted	appendix	any
DLP-PS	Siscrete log.	restricted	recovery	message recovery
RSA-PS	RSA	restricted	recovery	message recovery

signature schemes with message recovery, and creates a proxy signer's parameter. In the following, a proxy signer signs m_p by using a manner of signature schemes with appendix. Therefore in the verification, the proxy signer does not need to sends $\mathcal{M}_p^{uc}$.

For further discussion, the performance evaluation will be given in Section 7.5.

7.3 Proposed Proxy signatures

7.3.1 DLP-basesd Scheme

We show the scheme (DLP-PS), which is based on discrete logarithm problem. Let S_O, S_P and V be original signer, proxy signer and verifier, respectively. We first introduce some parameters and the proxy signature in DLP-PS.

- **Original signer's parameter:** The original public key is (p, g, y), and the corresponding original secret key is x.
- **Proxy signer's parameter:** The proxy public key is the pair (y, ρ) and the corresponding proxy secret key is σ.
- **Proxy signature:** The proxy signature for m_p is (r, s).

The protocol is given as follows.

Original parameter generation: S_O executes the following step.

1. (Original generation) S_O picks up an prime p and finds an generator $g \in Z_p^*$. She picks up a random number $x \in Z_{p-1}$ and computes $y = g^x \bmod p$.

Proxy parameter generation: S_O and S_P execute the following steps.

2. (Proxy generation) S_O creates a usage condition $\mathcal{M}_p^{uc} \in Z_p$, generates $\kappa \in Z_{p-1}$ randomly, and computes:

$$\begin{aligned} \rho &= \mathcal{M}_p^{uc} g^{-\kappa} \mod p; \\ \sigma &= -x\rho + \kappa \mod p-1. \end{aligned}$$

3. (Proxy delivery) S_O sends a pair (ρ, σ) to S_P through a secure channel.
4. (Proxy verification) S_P recovers $\mathcal{M}_p^{uc}$ by computing:

$$\mathcal{M}_p^{uc} = g^{\sigma} y^{\rho} \rho \bmod p.$$

If both redundancy and contents in $\mathcal{M}_p^{uc}$ are valid, She accepts it as a correct proxy parameter. Otherwise she rejects it.

Signature and verification phase: S_P and V execute the following steps.

5. (Signing by the proxy signer) In order to generate a signature on behalf of S_O, S_P chooses a message $m_p \in \mathbb{Z}$, picks up a random number $k \in \mathbb{Z}_{p-1}$, computes:

$$\begin{aligned} r &= g^{-k} && \bmod p; \\ s &= \sigma + kh(r, m_p) && \bmod p-1, \end{aligned}$$

and sends $(m_p, (r, s), \rho)$ to V. Then the proxy signature for m_p is (r, s).

6. (Verification of the proxy signature) V verifies $\mathcal{M}_p^{uc}$ by using the original and proxy public keys (n, g, y, ρ). Then she computes:

$$\mathcal{M}_p^{uc} = g^s v r^{h(r,m_p)} \bmod p.$$

If both redundancy and contents in $\mathcal{M}_p^{uc}$ are valid, she accepts it as a correct proxy signature. Otherwise she rejects it.

In Step 6, if the proxy signature is valid, then the verification holds since:

$$\begin{aligned} g^s v r^{h(r,m_p)} \bmod p &= g^{\sigma + kh(r,m_p)} y^{\rho} \rho g^{-kh(r,m_p)} \\ &= g^{-x\rho + \kappa + kh(r,m_p)} g^{x\rho} \mathcal{M}_p^{uc} g^{-\kappa} g^{-kh(r,m_p)} \\ &= \mathcal{M}_p^{uc} \quad \bmod p. \end{aligned}$$

7.3.2 RSA-based Scheme

We show the protocol RSA-PS which is based on the RSA problem. Let S_O, S_P and V be original signer, proxy signer and verifier, respectively. We first introduce some parameters and the proxy signature in RSA-PS.

- **Original signer's parameter:** The original public key is (n, e, g) and the corresponding original secret key is x.
- **Proxy signer's parameter:** The proxy public key is $\mathcal{M}_p^{uc}$ and the corresponding proxy secret key is σ.
- **Proxy signature:** The proxy signature for m_p is (r, s).

The protocol is given as follows.

Original parameter generation: S_O executes the following step.

1. (Original generation) S_O picks up two primes p and q and computes n. She generates a element $g \in \mathbb{Z}_n^*$ which is a generator of both Z_p^* and Z_q^*. She also picks up $e \in Z_{\varphi(n)}^*$ and computes x such that $ed = 1 \bmod \varphi(n)$.

Proxy parameter generation: S_O and S_P execute the following steps.

2. (Proxy generation) S_O creates a usage condition $\mathcal{M}_p^{uc} \in Z_p$ and computes:

$$\sigma = (\mathcal{M}_p^{uc})^x \bmod n.$$

3. (Proxy delivery) S_O sends σ to S_P through a secure channel.
4. (Proxy verification) S_P recovers $\mathcal{M}_p^{uc}$ by computing:

$$\mathcal{M}_p^{uc} = \sigma^{\ e} \bmod n.$$

If both redundancy and contents in $\mathcal{M}_p^{uc}$ are valid, She accepts it as a correct proxy parameter. Otherwise she rejects it.

Signature and verification phase: S_P and V execute the following steps.

5. (Signing by the proxy signer) In order to generate a signature on behalf of S_O, S_P chooses a message $m_p \in \mathbb{Z}$, picks up a random number $k \in \mathbb{Z}_n$, computes:

$$\begin{aligned} r &= g^{kh(m_p)}\sigma \mod n; \\ s &= g^{-ek} \mod n, \end{aligned}$$

and sends $(m_p, (r, s))$ to V. Then the proxy signature for m_p is (r, s).

6. (Verification of the proxy signature) V verifies $\mathcal{M}_p^{uc}$ by using the original and proxy public keys $(n, e, g, \mathcal{M}_p^{uc})$. Then she computes:

$$\mathcal{M}_p^{uc} = r^e s^{h(m_p)} \bmod n.$$

If both redundancy and contents in $\mathcal{M}_p^{uc}$ are valid, she accepts it as a correct proxy signature. Otherwise she rejects it.

In Step 6, if the proxy signature is valid, then the verification holds since:

$$\begin{aligned} r^e s^{h(m_p)} \bmod n &= (g^{kh(m_p)}\sigma)^e (g^{-ek})^{h(m_p)} \\ &= g^{ekh(m_p)} (\mathcal{M}_p^{uc})^{ex} g^{-ekh(m_p)} \\ &= \mathcal{M}_p^{uc} \mod n. \end{aligned}$$

Note that this protocol includes the idea of *ID-based cryptosystem* [48], which is an asymmetric system employing user's identities instead of public keys. Therefore $\mathcal{M}_p^{uc}$ created by the original signer is employed as the proxy public key, and this value is recovered in the verification.

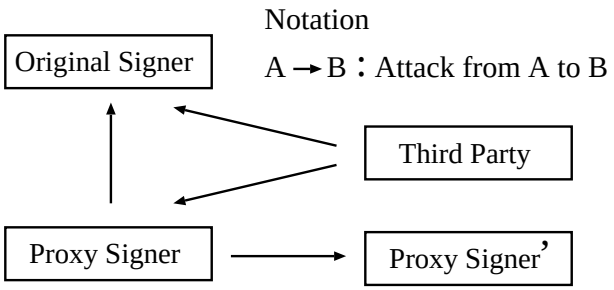

Figure 7.1: Adversary's target

7.4 Security Consideration

Figure 7.4 illustrates the entities appeared in proxy signatures and adversary's target from one entity and the other one. In the figure, we see the following entities:

- Original signer;
- Proxy signer;
- The other proxy signer different from the above one;
- Third party who is not a designated proxy signer.

We consider the above all entities except for the original signer may attack. On such attacks, the adversaries can use all information they have. Especially, the following attack model is considered in this thesis:

Insider attack: The adversary who is a designated proxy signer, try to obtain forged signature of the other's proxy signer. She can give a partial proxy signature of a proxy message m_p to the honest proxy signer and obtain a valid partial signature.

As a strategy, the reductions and mathematical function, given in Section 2.2.2, are used in our poofs.

Next, we explain hash function in detail to discuss the rigorous security of our protocols. During the discussion of security consideration, we may use following hash function: Let h be an *ideal* hash function as mentioned before. Then we define $\tilde{h}$ as follows.

$$\tilde{h}(x) := \begin{cases} h(x) & \text{if } x \neq 0; \\ 1 & \text{if } x = 0; \end{cases}$$

To prove the security for our systems, we use $\tilde{h}$ as our hash function instead of h. The system using h is trivially *more secure* than that using $\tilde{h}$. Hence the proofs of the security for the system using $\tilde{h}$ can guarantee that for the systems using h. Hereafter we denote $\tilde{h}$ by h for the simplicity.

For the proof of security, we use the functions `RSA` and `DLP` given in Section 2.2.3. Additionally, the following functions are also used.

Definition 7.4.1 (Function to break DLP-PS) DLP-PS $(p,\ g,\ y,\ \rho,\ m_p,\ r,\ s,\ \sigma')$ is a function that on input $p \in \mathbb{N}_{\text{prime}}$, $g \in \mathbb{Z}_p^*$, $y \in \mathbb{Z}_p^*$, $\rho \in \mathbb{Z}_p^*$, $m_p \in \{0,1\}^*$, $r \in \mathbb{Z}_p^*$, $s \in \mathbb{Z}_{p-1}$, $\sigma' \in \mathbb{Z}_{p-1}$, outputs $\sigma \in Z_{p-1}$ such that $y = g^x \bmod p$, $\rho = \mathcal{M}_p^{uc} g^{-\kappa} \bmod p$, $\sigma = -x\rho + \kappa \bmod p-1$, $r = g^{-k} \bmod p$, $s = \sigma + kh(r, m_p) \bmod p-1$ with a hash function h, $x \in \mathbb{Z}_{p-1}$, $\mathcal{M}_p^{uc} \in \mathbb{Z}_{p-1}$, $\kappa \in \mathbb{Z}_{p-1}$, $k \in \mathbb{Z}_{p-1}$, $\rho' = \mathcal{M}'^{uc}_p g^{-\kappa'} \bmod p$, $\sigma' = -x\rho' + \kappa' \bmod p-1$, $\mathcal{M}'^{uc}_p \in \mathbb{Z}_{p-1}$, $\kappa' \in \mathbb{Z}_{p-1}$, if such a σ exists. ■

Definition 7.4.2 (Function to break RSA-PS) RSA-PS $(n,\ e,\ g,\ m_p,\ r,\ s,\ \sigma')$ is a function that on input $n \in N_{>1}$, $e \in Z^*_{\lambda(n)}$, $g \in Z_n^*$, $m_p \in \{0,1\}^*$, $r \in Z_n^*$, $s \in Z_n^*$, $\sigma' \in Z_n^*$, outputs $\sigma \in Z_n^*$ such that $n = pq$, $\sigma = (\mathcal{M}_p^{uc})^x \bmod n$, $r = g^{kh(m_p)}\sigma \bmod n$ with a hash function h, $s = g^{-ek} \bmod\ n$, $p \in N_{prime}$, $q \in N_{prime}$, $\mathcal{M}_p^{uc} \in Z_n$, $k \in Z_n$, $\sigma' = (\mathcal{M}'^{uc}_p)^x \bmod n$, $\mathcal{M}'^{uc}_p \in Z_n$, if such a σ exists. ■

The results of the difficulties to forge a signature are given as follows.

Theorem 7.4.3 (Security of DLP-PS)

1. DLP $\leq_m^{fp}$ DLP-PS.
2. DLP-PS $\leq_T^{fp}$ DLP.

Proof.

1. $\texttt{DLP}(p, g, X) = \texttt{DLP-PS}\ (p,\ g,\ X^{-1},\ 1,\ 1,\ 1,\ 1,\ 1,\ 1,\ 1)$
2. DLP-PS $\leq_T^{fp}$ DLP:
   $\texttt{DLP-PS}\ (p,\ g,\ y,\ \rho,\ m_p,\ r,\ s,\ m'_p,\ r')$
   $= -\texttt{DLP}(p, g, y)\rho + \texttt{DLP}(p, g,\ (g^s v r^{h(r,m_p)})^{-1} \cdot \rho) + \texttt{DLP}(p, g, r')h(r', m_p)$

□

Theorem 7.4.4 (Security of RSA-PS)

1. RSA $\leq_m^{fp}$ RSA-PS.
2. RSA-PS $\leq_T^{fp}$ RSA.

Proof.

1. $\texttt{RSA}(n,\ e,\ C) = \texttt{RSA-PS}\ (n,\ e,\ 1,\ 1,\ 1,\ 1,\ 1,\ \mathcal{M}'^{uc}_p)$
2. $\texttt{RSA-PS}\ (n,\ e,\ g,\ m_p,\ r,\ s,\ \tilde{\sigma},\ m'_p,\ s')\ \texttt{RSA}(n,\ e,\ C)$
   $= \texttt{RSA}(n,\ e,\ (s')^{-h(m'_p)}) \cdot \texttt{RSA}(n,\ e,\ r^e s^{h(m_p)})$

□

Table 7.2: Performance of computation time.

	Original-PS	Original-PS (with $\mathcal{M}_p^{uc}$)	DLP-PS	RSA-PS
Proxy parameter generation	1280	2560	1280	1280
Proxy signature generation	$1280 + H$	$1280 + H$	$1280 + H$	$2560 + H$
Verification	$2560 + H$	$5120 + H$	$3840 + H$	$2048 + H$

7.5 Performance Evaluation

In this section, we evaluate the efficiency of our proposed schemes with respect to both computational time and transmitted data size.

Let us denote the computation time for modular multiplication and squaring by M and S, where the modulo size for Z_p and Z_n is 1024 bits, and where $|e| = 16$ bits for RSA-PS. Here we assume that $s = 0.75M$, and that we use a primitive method like binary method [23] to compute exponentiation.

From Table 7.2 and 7.3, we see that the efficiency of our schemes are better than that of original one. Here Original-PS denotes the scheme of both MUO-scheme and KPM-scheme.

In Table 7.2, compared DLP-PS with Original-PS(with $\mathcal{M}_p^{uc}$), we see that computation time for verification is reduced by 25%. This is the reason why Original-PS(with $\mathcal{M}_p^{uc}$) needs two verification, whereas DLP-PS needs only one. On the relationship between RSA-PS and Original-PS(with $\mathcal{M}_p^{uc}$), those types are different. Hence it is not so significant to compare the computation time. So is it on the transmitted data size.

In Table 7.3, we focus on the values in the first row at first. Usually, those original public keys will be preliminary delivered to each verifier and they store them. Therefore, most important values are those in the third row, because the proxy signer must send an above mentioned-size signature for verification. Compared DLP-PS with Original-PS(with $\mathcal{M}_p^{uc}$), the transmitted data size for verification is reduced by less than 40%.

Table 7.2 shows computation time, where H means amount of work to compute a hashed values, and Table 7.3 shows transmitted data size. To evaluate the same condition, both Original-PS and Original-PS(with $\mathcal{M}_p^{uc}$) are executed by the method of DLP-based message recovery.

7.6 Conclusion

We have proposed two new proxy signature schemes, called DLP-PS and RSA-PS, In both schemes, the original signer can control the signing power of proxy signer to some extent. DLP-PS, improved the basic proxy signature [27], controls power of a proxy

Table 7.3: Performance of transmitted data size.

	Original-PS	Original-PS (with $\mathcal{M}_p^{uc}$)	DLP-PS	RSA-PS
Org. pub. key	3072	3072	3072	2064
Pxy. par.	2048	4096	2048	1024
Verification	$3072 + \|m_p\|$	$5120 + \|m_p\| + \|\mathcal{M}_p^{uc}\|$	$3072 + \|m_p\|$	$2048 + \|m_p\|$

signer by adding usage condition to the proxy signer's public key implicitly, which uses idea of message recovery. In verifying proxy signer's signature, the usage condition can be checked independent of a message. RSA-PS controls the signing power of a proxy signer by adding usage condition to proxy signer's public key explicitly, which uses idea of ID-based cryptosystem [48].

Chapter 8

Conclusion

The main goal of this thesis is to design and analyze practical authentication cryptosystems. Throughout this thesis, we focus on both efficiency and provable security of the proposed systems.

In this thesis, we have studied the following themes:

1. We have defined two mathematical problems, named self-powering RSA problem and extended self-powering RSA problem. Those are used as underlying problem in our schemes. Using the problems, we have constructed efficient cryptosystems. We also have proposeed an identification scheme whose security is based on our problems. Our scheme is given by improving Guillou-Quisquater identity-based scheme [18]. In the *identity-based* [48] scheme, we can see the two types of attack as follows. *One key attack*: Adversary try to forge for the fixed ID-based public key. *Possible key attack*: Adversary try to forge for the possible ID-based public-keys.

   Under one key attack scenario, our scheme is as secure as extended self-powering RSA problem. On the other hand, our scheme is as secure as self-powering RSA problem under the self-powering RSA problem.

2. We have constructed on the fly signatures by improv PS-scheme. In our schemes, a public-key g has a specific structure. Consequently, in comparison with PS-scheme, the size of secret-key is small ($\ll |n|/2$). In the following, our schemes realize a compactness of signature. Especially, the computation work in verification are much reduced by the changing n in $x = g^{y-ne} \bmod n$ (PS-scheme) into z in $x = g^{y-ze} \bmod n$ (our schemes).

   Concrete to say, compared with PS-scheme, the size of a secret-key and a signature can be reduced by at least 69% and 47%, respectively. Furthermore, Our scheme has an advantage that the computational cost can also be smaller. Compared with PS-scheme, the computational cost for pre-computation, signature generation and verification can be reduced by at least 38%, 69%, and 64% (resp. 54%, 63%, and 61%), respectively.

3. We have proposed fast signature scheme without on-line multiplication. The proofs for our schemes are based on a formal security model and we have given the provable security. We aim to provide our schemes that can be applied in current smart cards.

   As for the security, our scheme is as secure as integer factoring problem based on a RSA modulus n (in the random oracle model). To satisfy the security, our schemes use a public key g with specific structure, called *asymmetric basis*, and which is a variant of [33]. This property leads to the good efficiency in terms of both size of data and amount of work.

   Concrete to say, compared with OTM-scheme, the size of a signature in our signature can be reduced by at least 21%, and computational cost in our scheme for verification can be reduced by 32%, respectively. In the same way, compared with PS-scheme, the size of a secret key and a signature can be reduced by at least 68% and 58%, respectively. Furthermore, the computational cost for pre-computation and verification can be reduced by at least 83% and 78%, respectively.

4. We have constructed a new notion for proxy signature and proposed two schemes, named DLP-PS and RSA-PS. The main idea is that our schemes are based on original signer's message recovery. This feature leads to the reduction of transmitted data size and controls the proxy signer's power efficiently.

   DLP-PS, improved the basic proxy signature [27], controls the power of a proxy signer by adding usage condition to proxy public key implicitly, which uses idea of message recovery. In verifying proxy signer's signature, the usage condition can be checked independent of a message. RSA-PS controls the power of a proxy signer by adding usage condition to proxy public key explicitly, which uses idea of ID-based cryptosystem. Up to the present, any proxy signature is based on DLP. Therefore RSA-PS is a new type scheme, which is based on RSA.

In the course, we make some technical and theoretical contributions which enhance our understanding of digital signatures including identifications.

Bibliography

[1] E. Bach. *Discrete logarithms and factoring.* Technical Report UCB/CSD 84/186. Computer Science Division (EECS), 1984.

[2] N. Baric and B. Pfitzmann. Collision-free accumulators and fail-stop signature schemes without trees. In Walter Fumy, editor, *Advances in Cryptology - EuroCrypt '97*, pages 480–494, Berlin, 1997. Springer-Verlag. Lecture Notes in Computer Science Volume 1233.

[3] Paul Barrett. Implementing the rivest shamir and adleman public key encryption algorithm on a standard digital signal processor. In A.M. Odlyzko, editor, *Proc. CRYPTO 86*, pages 311–323. Springer-Verlag, 1987. Lecture Notes in Computer Science No. 263.

[4] M. Bellare and S. Micali. How to sign given any trapdoor function. In *Crypto '88*, LNCS No. 403, pages 200–215. Springer-Verlag, 1990.

[5] M. Bellare and P. Rogaway. Random oracles are practical: A paradigm for designing efficient protocols. In *Proc. of 1st ACM Conference on Computer and Communications Security*, pages 62–73. Springer-Verlag, 1993.

[6] M. Bellare and P. Rogaway. The exact security of digital signatures – how to sign with RSA and Rabin. In *Eurocrypt '96*, LNCS No. 1070, pages 399–416. Springer-Verlag, 1996.

[7] E. F. Brickell and K. S. McCurley. An interactive identification scheme based on discrete logarithms and factoring. *Journal of Cryptology*, 5:29–39, 1992.

[8] Damgard. Collision free hash functions and public key signature schemes. In *Eurocrypt '87*, LNCS No. 304, pages 203–216. Springer-Verlag, 1988.

[9] W. Diffie and M. E. Hellman. New directions in cryptography. In *IEEE Trans. Information Theory*, volume IT-22, pages 644–654, 1976.

[10] U. Feige, A. Fiat, and A. Shamir. Zero-knowledge proofs of identity. *Journal of Cryptology*, 1:77–95, 1988.

[11] Uriel Feige, Amos Fiat, and Adi Shamir. Zero knowledge proofs of identity. In *Proc. 19th ACM Symp. on Theory of Computing*, pages 210–217, May 1987.

[12] A. Fiat and A. Shamir. How to prove yourself: practical solutions of identification and signature problems. In *Crypto '86*, LNCS No. 263, pages 186–194. Springer-Verlag, 1987.

[13] National Institute for Standards and Technology. Digital Signature Standard (DSS). *Federal Register*, 56(169), August 30 1991.

[14] E. Fujisaki and T. Okamoto. Statistical zero knowledge protocols to prove modular polynomial relations. In Burt Kaliski, editor, *Advances in Cryptology - Crypto '97*, pages 16–30, Berlin, 1997. Springer-Verlag. Lecture Notes in Computer Science Volume 1294.

[15] M. Giraut. Self-certified public keys. In *Eurocrypt '91*, LNCS No. 547, pages 490–497. Springer-Verlag, 1992.

[16] S. Goldwasser, S. Micali, and R. L. Rivest. A digital signature scheme secure against adaptive chosen-message attacks. *SIAM*, 17(2):281–308, April 1988.

[17] L. C. Guillou and J. J. Quisquater. A practical zero-knowledge protocol fitted to security microprocessor minimizing both transmission and memory. In Christof G. Günther, editor, *Advances in Cryptology - EuroCrypt '88*, pages 123–128, Berlin, 1988. Springer-Verlag. Lecture Notes in Computer Science Volume 330.

[18] L. C. Guillou and J. J. Quisquater. A "paradoxal" identity-based signature scheme resulting from zero-knowledge. In *Crypto '88*, LNCS No. 403, pages 216–231. Springer-Verlag, 1989.

[19] L. C. Guillou and J. J. Quisquater. A "paradoxical" identity-based signature scheme resulting from zero-knowledge. In Shafi Goldwasser, editor, *Advances in Cryptology - Crypto '88*, pages 216–231, Berlin, 1989. Springer-Verlag. Lecture Notes in Computer Science Volume 403.

[20] A. Karatsuba and Yu Ofman. Multiplication of multidigit numbers on automata. *Doklady Akademii Nauk SSSR*, 145(2):293–294, 1962.

[21] S. Kim, S. Park, and D. Won. Proxy signatures, revisited. In *ICICS '97*, LNCS No. 1334, pages 223–232. Springer-Verlag, 1997.

[22] D. E. Knuth. *Seminumerical Algorithms*, volume 2 of *The Art of Computer Programming*. Addison-Wesley, 1969. Second edition, 1981.

[23] D. E. Knuth. *Sorting and Searching*, volume 3 of *The Art of Computer Programming*. Addison-Wesley, 1998. Second edition.

[24] N. Y. Lee, T. Hwang, and C. H. Wang. On zhang's nonrepudiable proxy signature schemes. In *ACISP '98*, LNCS No. 1438, pages 415–422. Springer-Verlag, 1998.

[25] A. K. Lenstra, H. W. Jr. Lenstra, M. S. Manassse, and J. M. Pollard. The number field sieve. In *Proc. of ACM Annual Symposium on Theory of Computing*, pages 564–572, 1990.

[26] H. W. Jr. Lenstra. Factoring integers with elliptic curves. *Annals of Mathematics*, 126:649–673, 1987.

[27] K. Usuda M. Mambo and E. Okamoto. Proxy signatures: Delegation of the power to sign messages. *IEICE Trans. Fundamentals*, E79-A(9):1338–1354, 1996.

[28] G. Miller. Riemann's hypothesis and test for primality. *Journal of Computer and System Sciences*, 13:300–317, 1976.

[29] M. Naor and M. Yung. Universal one-way hash functions and their cryptographic applications. In *Proc. 21st ACM Symposium on the Theory of Computing*, pages 33–43, 1989.

[30] National Institute of Standards and Technology (NIST). Secure hash standard(SHS). In *Federal Information Processing Standards*, April 1995.

[31] National Bureau of Standards. Announcing the data encryption standard. Technical Report FIPS Publication 46, National Bureau of Standards, January 1977.

[32] K. Ohta and T. Okamoto. On concrete security treatment of signatures derived from identification. In *Crypto '98*, LNCS No. 1462, pages 354–369. Springer-Verlag, 1998.

[33] T. Okamoto. Provably secure and practical identification schemes and corresponding signature schemes. In *CRYPTO92*, pages 31–53. Springer-Verlag, 1992. LNCS No. 740.

[34] D. Poincheval. The composite discrete logarithm and secure authentication. In *PKC '00*, LNCS No. 1751, pages 113–128. Springer-Verlag, 2000.

[35] D. Poincheval and J. Stern. Security arguments for digital signatures and blind signatures. *Journal of Cryptology*, 2000.

[36] J. Pollard. Monte carlo methods for index computation (mod p). volume 32, pages 918–924. Mathematics of Computation, 1978.

[37] G. Poupard. *Authentification d'Entités, de Messages et de Clés Cryptographiques : Théorie et Pratique*. PhD thesis, Ecole Polytechnique, 2000.

[38] G. Poupard and J. Stern. Security analysis of a practical "on the fly" authentication and signature generation. In *Eurocrypt '98*, LNCS No. 1403, pages 422–436. Springer-Verlag, 1998.

[39] G. Poupard and J. Stern. On the fly signatures based on factoring. In *Proc. of the 6th CCS*, pages 48–57. ACM Press, 1999.

[40] P. Ribenboim. *The Book of Prime Number Records.* Splinger-Verlag, New York, 1988.

[41] R. L. Rivest. The MD5 message-digest algorithm. Internet Request for Comments, April 1992. RFC 1321.

[42] R. L. Rivest, A. Shamir, and L. M. Adleman. A method for obtaining digital signatures and public-key cryptosystems. *Communications of the ACM*, 21(2):120–126, 1978.

[43] J. Rompel. One way functions are necessary and sufficient for secure digital signatures. In *Proc. 22stC ACM Symposium on the Theory of Computing*, pages 387–394. ACM, 1990.

[44] RSA Data Security, Inc. *PKCS #1: RSA Encryption Standard*, June 1991. Version 1.4.

[45] K. Sakurai and H. Shizuya. A structural comparison of the computational difficulty of breaking discrete log cryptosystems. *Journal of Cryptology*, 11-1:29–43, 1998.

[46] C. P. Schnorr. Efficient identification and signatures for smart cards. pages 239–252. Springer, 1990. Lecture Notes in Computer Science No. 435.

[47] C. P. Schnorr. Efficient signature generation by smart cards. *Journal of Cryptology*, 4:161–174, 1991.

[48] A. Shamir. Identity-based cryptosystems and signature schemes. In *CRYPTO '84*, LNCS No. 196, pages 47–53. Springer-Verlag, 1985.

[49] C. E. Shannon. Communication theory of secrecy systems. *Bell Sys. Tech. J.*, 28:657–715, 1949.

[50] R. D. Silverman. A cost-based security analysis of symmetric and asymmetric key length. RSA Laboratories, CryptoBytes, Bulletins, Number 13, April 1999.

[51] Martin Tompa and Heather Woll. How to share a secret with cheaters. In A.M. Odlyzko, editor, *Proc. CRYPTO 86*, pages 261–265. Springer-Verlag, 1987. Lecture Notes in Computer Science No. 263.

[52] H. Woll. Reductions among number theoretic problems. In *Information and Computation*, volume 72, pages 167–179, 1976.

[53] K. Zhang. Threshold proxy signature schemes. In *ISW '97*, LNCS No. 1396, pages 191–197. Springer-Verlag, 1997.

Publications

Journal and Transactions:

1. T. Okamoto, M. Tada and A. Miyaji: "Proposal of Efficient Signature Schemes based on Factoring," Trans. IPSJ, Vol.42 No.8 (2001), pp. 2123-2133, 2001 (in Japanese).

2. S. Kim, M. Mambo, T. Okamoto, H. Shizuya, M. Tada and D. Won: "On the security of the Okamoto-Tanaka ID-based key exchange scheme against active attack," IEICE transactions fundamentals, vol.E84-A, no.1, pp.231-238, 2001.

International Conference:

1. T. Okamoto, M. Tada, E. Okamoto and K. Kamachi: "Revocable and renewable ID-based key distribution system for user's secret information", Proceedings of 1998 First Japan-Singapore Joint Workshop on Information Security (JWIS'98), pp.3-13, 1998.

2. T. Okamoto, M. Tada and E. Okamoto: "Extended proxy signature for smart cards", Proceedings of Information Security Workshop 1999 (ISW'99), Lecture Notes in Computer Science 1729, pp.247-258, Springer-Verlag, 1999.

3. T. Okamoto, M. Tada and A. Miyaji "Efficient "on the fly" signature schemes based on integer factoring", Proceedings of Indocrypt 2001, Lecture Notes in Computer Science 2247, Springer-Verlag, 2001.

4. T. Okamoto, M. Tada and A. Miyaji "Improved fast signature schemes without online multiplication", Proceedings of Financial Cryptography 2002 (FC'02), Springer-Verlag, 2002 (to appear).

National Technical Symposia, etc:

1. T. Okamoto, M. Tada and E. Okamoto: "Identity-based fault-tolerant key distribution system," Symposium on Cryptography and Information Security (SCIS'99), Vol I, pp.153-158, 1999.

2. T. Okamoto, M. Tada and A. Miyaji: "The Rigorous Security for Okamoto-Tanaka ID-based Key Exchange Scheme against Active Attack," Computer Security (CSEC), Vol.99, No.54, pp.7-12, 1999 (in Japanese).

3. T. Okamoto, M. Tada and A. Miyaji: "Proxy signatures based on original signer's message recovery," Computer Security Symposium (CSS'99), pp.19-24, 1999.

4. T. Okamoto, M. Tada and A. Miyaji: "Efficient signature schemes based on factoring," IEICE Technical Report, ISEC2000-61 (2000-9), pp.21-28, 2000.

5. T. Okamoto, M. Tada and A. Miyaji: "Proposal of Efficient Signature Schemes based on Factoring," IEICE Technical Report, ISEC2001-18 (2001-05), pp.59-66, 2001 (in Japanese).

6. T. Okamoto, M. Tada and A. Miyaji: "Security Analysis of Signature Schemes Based on Fast on-line Computation," IEICE Technical Report, ISEC2001-21 (2001-11), 2001.

7. T. Okamoto, M. Tada and A. Miyaji: "A proposal of identification scheme based on variant RSA problems," Symposium on Cryptography and Information Security (SCIS'02), 2002 (to appear).

Master's thesis:

1. T. Okamoto: " Studies on Identity-Based Fault-Tolerant Key Distribution Systems," A master's thesis at Japan Advanced Institute of Science and Technology, Mar. 2001.