Provably-Secure Remote User Authentication Without Special Devices

特別な装置を持たない利用者の遠隔認証とその証明可能安全性に関する研究

Kazukuni Kobara

古原 和邦

Preface

Since the commercial use of Internet started in the 1990s, Internet has been gaining popularity as a communication infrastructure with the help of its various services and contents and access methods, such as xDSL (Digital Subscriber Line), i-mode, WAP (Wireless Application Protocol), WLAN (Wireless LAN) from hot-spots and so on.

The various connection methods increase the opportunity for users to access the Internet from public places. They, however, increase the risk of attacks in the real world as well as over the network. For example, in public places, the risk of peep of passwords and the risk of having a device lost or stolen are definitely higher than at home. Therefore new authentication schemes secure even against the above attacks are required. The new schemes should be implemented with low cost over conventional terminals (that are equipped with only keyboards and displays as user interfaces, but no tamper resistant area or no bio-sensors) so that users can use any terminal in the world.

Formally, in this dissertation we study remote user authentication under the following scenario. As terminals, we assume conventional ones equipped with only keyboards and displays as user interfaces, but no special devices, such as bio-sensors and tamper resistant area. As adversaries, we assume real world adversaries and network adversaries. The real world ones can peep all the information that is displayed on the terminals and typed to the terminals. They can also steal all the private devices from users. The network adversaries can eavesdrop all the communication between terminals and servers, and also can set up fake servers and then let users connect them.

This dissertation is composed as follows: Chapter 1 gives a brief view of this dissertation. In Chapter 2, we focus on CRHI (Challenge-Response Human Identification) as a countermeasure against real world adversaries. We evaluate the exact resistance against peeping attacks on it, and propose a challenge-control method. While the original CRHI resists only one peep for practical parameters, our challenge-controlled CRHI can resist more peeps.

In Chapter 3, we propose a further improvement of the resistance against peeps. Precisely, we propose how to limit the visible space of the decoded image of VSS (Visual

Secret Sharing). We call it LVSVSS (Limiting Visible Space VSS). We evaluate the visibility of the space, and show that it is possible to transmit messages to a user in certain position. This means the combination of LVSVSS with CRHI can deal with real world adversaries. While this scheme requires users to possess slides, it cannot be any threat against the real world adversaries even if they get the slide since it is independent of the users' passwords.

In Chapter 4, we consider preventing attacks over network using PKCs (Public-Key Cryptosystems). Since (primitive) PKCs do not have desirable properties required for the ideal PKCs, we propose how to convert primitive PKCs into ideal ones. We evaluate the primitive PKCs based on the decoding problems, and show they can be ideally strong using our conversions.

In Chapter 5, we study PAKE (Password-Authenticated Key-Exchange) to remove the burden on users and administrators of public-key and certificate management. We propose a simple scheme that is almost as efficient as the Diffie-Hellman key-exchange in communication and computation costs, and prove that its security can be reduced to DDH (Decision Diffie-Hellman) problem under standard assumptions.

Chapter 6 presents a summary of this research and possible extensions. The combination of CRHI and LVSVSS can deal with real world adversaries, and the ideally strong PKC or PAKE can deal with network adversaries.

Acknowledgments

I particularly wish to express my sincere gratitude to Professor Hideki Imai of the University of Tokyo for his very important advice and suggestions in kind support of my dissertation. I greatly appreciate the important help of Associate Professor Kanta Matsuura of the University of Tokyo for his useful advice and suggestions. I also greatly appreciate the important help of Professor Toyoaki Nishida, Professor Hitoshi Aida, Professor Kiyoharu Aizawa and Associate Professor Kaoru Sezaki, all of the University of Tokyo.

I am also very grateful to Professor Tsutomu Matsumoto of Yokohama National University, Professor Kwangjo Kim of Information and Communications University, Professor Ji-Hwan Park of Pu-kyong National University and Professor Yuliang Zheng of University of North Carolina for their insightful feedback and valuable discussion on this work.

I would like to thank all the past and present members of Imai laboratory for their supports on various tasks including the administration of the laboratory LAN and the management of research projects. I also thank Dr. Michiharu Kudo of IBM Tokyo Research Laboratory for this LaTeXstyle file that was made by him, and are very helpful to compile my thesis.

Finally, I thank my wife Mayumi and my parents Koji and Kyoko very sincerely for their understanding and cheerful support.

March 1, 2003

Kazukuni Kobara
古原 和邦

概要

1990 年代に商用利用の始まったインターネットは、WWW (World Wide Web) の出現、i-mode, WAP (Wireless Application Protocol) などを用いた携帯電話からの利用、WLAN (Wireless LAN) やデータ通信カードを用いた PDA (Personal Digital Assistance) やノートＰＣなどからの利用、xDSL (Digital Subscriber Line) などの広帯域接続の低価格化などにも助けられ、普及の一途をたどり続けている。また、インターネット上で提供されるサービスも増加し、その範囲は、コンテンツのダウンロードや社内 LAN への接続からモバイルバンキングやモバイルトレーディングなどまで多岐にわたる。

インターネット上のサービスはインターネットに接続できさえすれば原則どこからでも利用できるという利点を持つが、利用者認証はその分厳密かつ安全に行う必要がある。例えば、インターネットカフェやキオスク端末などの公共の場所からインターネットに接続する際には、パスワードののぞき見などに注意する必要がある。また、自分の端末を外出中で利用する際には、端末の置き忘れや盗難などにも注意する必要がある。さらに、ネットワーク上では盗聴やサーバへの成りすましが行われている可能性もあり、それら全てに的確に対処する必要がある。

本研究では、以下のような状況において遠隔地にいる利用者を安全かつ安価に認証する方法、およびその安全性に関する研究を行う。まず、利用者の利用する端末はどこにでもある通常の端末を仮定する。つまり、ディスプレイとキーボードのみをユーザインターフェースとして備えており、身体的特徴を読み取る特殊な装置や耐タンパー装置などは備えていないものとする。この種の端末は特別なものではないため、どこにいても安価に調達できるという利点がある。次に、攻撃者としては、現実世界の攻撃者とネットワーク上の攻撃者を仮定する。現実世界での攻撃者は、認証の際に利用者に表示される画面と利用者の打ち込んだ情報を全てをのぞき見ることができるものとし、また、利用者が自分専用の端末（PDA、携帯電話、ノートＰＣなど）を持っている場合には、それらも入手可能であるとする。ネットワーク上の攻撃者は、ネットワークに流れる全データの盗聴、および偽のサーバを立ち上げそこに利用者を導き認証を行わせることができるものとする。

上記のような状況においても、安全かつ安価に遠隔地の利用者を認証することが本研究の目的であり、これに関して得られた結果を以下のような構成で記述する。まず、第 1 章では本学位論文で想定する環境や攻撃に関する説明を行う。第 2 章では、現実世界の攻撃者への対処方法として質問応答個人認証方式をとりあげ、そののぞき見に対する耐性の

厳密な評価をおこなう。また、質問の出し方を制御することにより耐性を向上させる方法の提案を行う。現実的な値を用いた場合、質問応答個人認証方式は 1 回程度ののぞき見にしか対処できないが、質問の出し方を制御することで、数回までののぞき見に対処できることを示す。第 3 章では、のぞき見に対する耐性を飛躍的に向上させる方法を提案する。具体的には、視覚復号型秘密分散方式を用いて可視空間を制限する方法を提案し、その可視空間特性を解析する。結果、特定の場所にいる利用者のみに情報を伝えることが可能であることを示す。このことは、質問応答個人認証方式と組み合わせることにより現実世界で行われるのぞき見に対処できることを意味する。この方式では利用者にスライドを持たせることになるが、これらのスライドが攻撃者に盗まれたとしても、攻撃者は利用者の秘密に関して何の情報も得ることはできないという利点を持つ。

第 4 章では、ネットワーク上で行われる攻撃を公開鍵暗号を用いて対処することを考える。すべての公開鍵暗号がネットワーク上で行われる攻撃に対して理想的な安全性を持っているとは限らないため、この章では理想的な公開鍵暗号を構成する方法に関する研究を行う。具体的には、関数の形によって（原始的な）公開鍵暗号を決定的関数、部分落し戸関数、完全落し戸関数に分類して、各関数がある条件を満たす場合にそれらを理想的な公開鍵暗号に変換する方法を提案する。また、線形符号の復号問題に基づく原始的な公開鍵暗号方式の安全性を評価し、それらがその条件を満たすことを示す。これらの結果より、ネットワーク上の攻撃は理想的な公開鍵暗号を用いることで対処可能であり、また、そのような暗号方式を構成することも可能であることを示す。公開鍵暗号を用いる対処方法は、しかしながら、利用者にサーバの公開鍵（もしくはその拇印）を持たせる必要がある。そこで第 5 章では、利用者が公開鍵を持たずにネットワーク上の攻撃に対処できる方式としてパスワード認証された鍵交換方式をとりあげ、その改良を行う。我々は、計算量および通信量の少ない方式を提案し、その方式の安全性が決定 Diffie-Hellman 問題に帰着できることを特殊な仮定を用いることなく示す。

第 6 章では全体の総括を行う。現実世界での攻撃は質問応用個人認証と可視空間を制限する視覚復号型個人認証方式を組み合わせることにより対処可能であり、ネットワーク上での攻撃は理想的な公開鍵暗号方式もしくはパスワード認証された鍵交換方式を用いることにより対処可能である。

Contents

List of Figures

List of Tables

Chapter 1

Introduction

As the services over Internet become popular, the importance of remote user authentication increases. The aim of this dissertation is at securing it against not only network adversaries but also real world adversaries with low cost. This chapter gives a brief view of this dissertation.

1.1 Background

Since the commercial use of Internet started in the 1990s, Internet has been widely spreading thanks to its various services and contents. People have access to it from not only home, offices, but also anywhere outside using public kiosk terminals in, e.g. airports or internet cafes. Authentication of remote users in public places is not so easy as that in secure places, such as at home, due to the risks of password peeping or device lost/stolen. In public places, typed passwords may be peeped by someone else, and personal devices may be lost or stolen. Unfortunately, current user authentications are either vulnerable to them or too expensive to employ. Therefore low-cost remote user authentications that are secure even against them are required especially when users may have access from public places.

1.2 Scope

The scope of this dissertation is illustrated in Fig. 1.1. Remote users try to connect with the server from remote terminals and the server authenticates them.

We assume that terminals and servers are reliable and not compromised, but there exists adversaries both in the public places of the real world and over the network. The

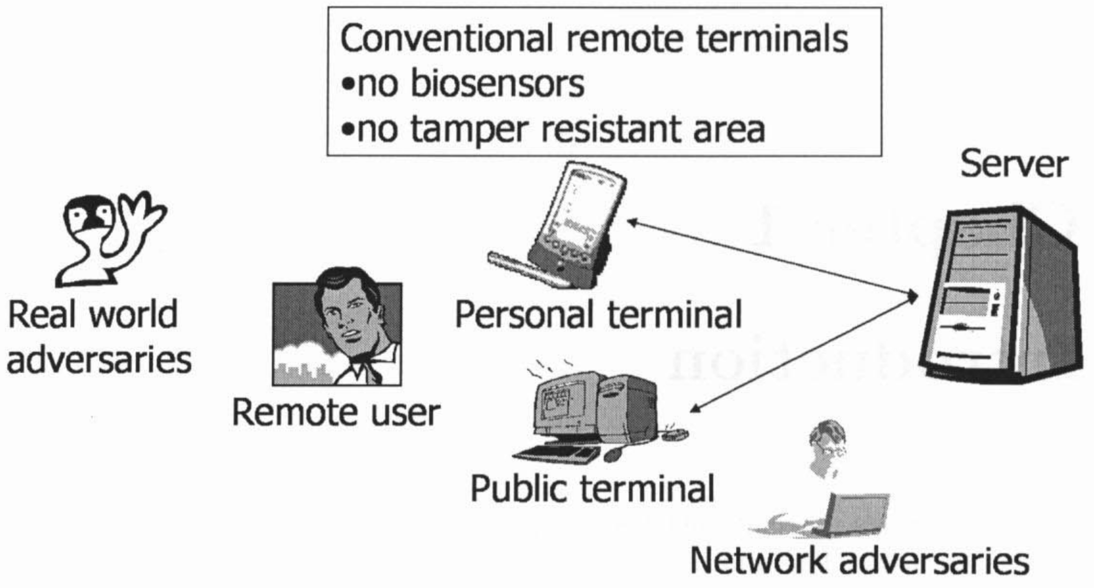

Figure 1.1: Environment dealt in this dissertation

goal of the adversaries is to impersonate authorized users. They are assumed to have the following abilities:

Real world adversaries:

- To peep all the displayed information on the terminals, and all the typed information to the terminals.
- To obtain all the private devices of users if they have.

Network adversaries:

- To eavesdrop all the communication data between terminals and servers.
- To set up fake servers and let users connect them.

We also assume that terminals are conventional ones and the requirements for them are minimal, i.e. they are equipped with only keyboards and displays as user interfaces, but neither bio-sensors nor tamper resistant areas. The advantage of such terminals is the cost. They are available anywhere with low prices and thus easy to update from old systems.

Table 1.1: Comparison of User Authentications

Schemes	Properties	Accuracy	Immunity to Device Theft	Cost	Burden	Refreshability
Individual Secret	Human-Memorized	○	○	○	×	○
	Recorded	○	×	△	△	○
Biometrics	Physical	×	○	×	○	△
	Action	×	○	×	△	△

○: Good, ×: Bad, △: Dependent on schemes

1.3 Related Works

Various studies have be conducted on user authentications. We categorize them and show the difference among them including ours.

1.3.1 Individual Secrets and Biometrics

Typical user authentication schemes identify the following information as the proof of the users:

Individual secrets

Human-memorized secrets: Secrets kept in mind, such as passwords.

Recorded secrets: Recorded secrets, such as secret codes on magnetic cards, secret keys in smart cards.

Biometrics

Physical features: Individual physical features, such as fingerprints, irises, retinas, hand geometry, facial features and so on.

Features of actions: Individual features of actions, such as voice, hand-written signatures and so on.

We make a rough comparison among them in Table 1.1.

Accuracy denotes how FAR (False Acceptance Rate) and FRR (False Rejection Rate) are small. Since FAR and FRR of biometrics are usually larger than that of individual secret and that some biometrics are forgeable [52], we rated them as ×.

Immunity to device theft denotes the resistance against adversaries who could obtain the users' possessions. Since the recorded secrets are vulnerable to them, we rated it as

$\times$. On the other hand, human-memorized passwords and biometrics have no risk of being theft (even though they might be vulnerable to peeping and forgery, respectively). Thus we rated them as ◯.

Cost denotes the cost of building up the authentication system. We rated biometrics as $\times$ since almost all of them require special devices, such as bio-sensors [1] We rated recorded secrets as $\triangle$ since their costs depend on the recording media, e.g. recording on paper is less expensive than smart cards.

Burden denotes how many steps are required for each authentication. We rated memorized-secrets as $\times$ since users need to remember them and type them. We rated the recorded-secrets as $\triangle$ since while the secrets on magnetic cards or in smart cards can be verified automatically by inserting them into scanners, the secrets on paper cannot usually and thus users have to type them by themselves. We rated actions as $\triangle$ since users have to take certain actions, which are troublesome in some cases.

Refreshability denotes whether the registered secrets or biometrics can be changed frequently or in case of compromise. We rated biometrics as $\triangle$ since most of them have only a small number of replacements.

While the advantage of the biometrics is users' burden, its accuracy and cost are not better than the individual secrets. Recall that our target is security and cost. Thus we focus the individual secrets in the next section.

1.3.2 Individual-Secret-Based Authentications

Currently available individual-secret-based authentications are categorized as follows according to the information the server verifies.

Schemes:

Plain Password (PW): Users transmit plain passwords to the server, and then the server verifies them. Needless to say, this is the most dangerous usage of passwords even though it is still widely used in POP, WEB accesses and so on.

Password protected by server's authorized public-key (PWpAPK): At first, users verify the server's public-key, and then establish a secure channel between the server. Then the users transmit their passwords over the secure channel and the server verifies them. This class includes PasswordAuthentication in SSH (Secure Shell) [80] and the classical simple password verification over secure channels established by the server authentication in SSL/TLS (Secure Socket Layer/Transport Layer Security)[18, 11] or IPsec [35].

[1]While some schemes including keystroke scanning, mouse-movement scanning require no special device, they have not been a complete technology yet to provide sufficient authentication by itself.

Password in challenge-response (PW in CR): The server verifies the passwords in a challenge-response way using cryptographic one-way functions[2]. This class includes the CHAP (Challenge-Response Handshake Protocol) [73].

Secret-key (SK in CR): The server verifies secret-keys in a challenge-response way. Note that the secret-keys here include both private-keys of asymmetric ciphers and symmetric-keys of symmetric ciphers. This class of private-keys includes RSAAuthentication, PubkeyAuthentication in SSH protocol version 1 and 2 respectively [80], the mutual authentication in SSL/TLS [18, 11] and so on. This class of symmetric-keys includes the CHAP using long-keys.

Secret-key protected by password (SKpPW in CR): This class is the same as the SK in CR except that the secret-key is encrypted with a password of the user.

Secret-key and password (SK&PW in CR): This class is the same as the SK in CR except that the responses are generated from both the secret-key and the password.

OTP derived from password (OTP from PW): The server verifies one-time passwords (OTPs) generated from users' passwords and (counter and/or time). This class includes S/KEY [29] and OPIE (One Time Passwords In Everything) [30].

OTP derived from secret-key (OTP from SK): The server verifies one-time passwords (OTPs) generated from secret keys and (counter and/or time).

OTP derived from password-protected secret-key (OTP from SKpPW): This class is the same as OTP from SK except that the secret-key is encrypted with the password of the user.

OTP derived from password-protected secret-key (OTP from SK&PW): This class is the same as OTP from SK except that OTP depends on the password of the user, too.

We see whether or not the above schemes can be secure against the following attacks:

Attacks:

Eavesdrop (E): Adversaries eavesdrop the communication channels.

Peep (P): Adversaries peep all the information typed by the users and displayed on the terminals.

[2]Challenge-response human identification in Chapter 2 do not use cryptographic one-way functions.

Theft (T): Adversaries steal all the personal belongings.

Eavesdrop then Off-line exhaustive search (EtO): Adversaries eavesdrop the communication channel and then perform off-line exhaustive search to extract secrets.

Theft then Off-line exhaustive search (EtO): Adversaries steal all the personal belongings and then perform off-line exhaustive search to extract secrets.

Peep then Theft (PtT): Adversaries peep the passwords at first and then steal the users' belongings.

Eavesdrop then Theft then Off-line exhaustive search (EtTtO): Adversaries eavesdrop the communication channel, steal the user's belongings, and then perform off-line exhaustive search to extract secrets.

Peep and Eavesdrop then Theft then Off-line exhaustive search (P&EtTtO): Adversaries eavesdrop the communication channel, steal the user's belongings, and then perform off-line exhaustive search to extract secrets.

We assume that the passwords used here are long enough to avoid on-line exhaustive searches, but too short to avoid off-line exhaustive searches. On-line exhaustive searches give candidate passwords to the server one-by-one and see whether they are accepted or not, whereas off-line exhaustive searches try to find secrets matching with the obtained data. The main difference between them is the search speed. One can execute off-line exhaustive searches highly in parallel.

The immunity against the above attacks is summarized in Table 1.2. As you can see, only PW is vulnerable to Eavesdrop. The schemes relying only on the human-memorized secrets are vulnerable to Peep, and that relying only on the recorded secrets are vulnerable to Theft. The schemes relying only on the human-memorized secrets are also vulnerable to EtO unless the communication is protected with PK or something. The recorded secrets protected by the human-memorized secrets are vulnerable to TtO. All the currently available schemes are unfortunately vulnerable to PtT and P&EtTtO.

The attacks including Peep and Theft are performed by real world adversaries whereas ones including Eavesdrop by network adversaries. The goal of our study is to improve the immunity against all the attacks performed by both real world and network adversaries.

1.4 Outline and Contributions

The outline and contributions of the following chapters are summarized as follows:

Table 1.2: Currently available individual-secret-based authentications and our target

Schemes \ Attacks	Eavesdrop	Peep	Theft	EtO	TtO	PtT	EtTtO	P&EtTtO
PW	×	×	○	×	×	×	×	×
PWpAPK	○	×	○	○	○	×	○	×
PW in CR	○	×	○	×	○	×	×	×
SK in CR	○	○	×	○	×	×	×	×
SKpPW in CR	○	○	○	○	×	×	×	×
SK&PW in CR	○	○	○	○	○	×	×	×
OTP from PW	○	×	○	×	○	×	×	×
OTP from SK	○	○	×	○	×	×	×	×
OTP from SKpPW	○	○	○	○	×	×	×	×
OTP from SK&PW	○	○	○	○	○	×	×	×
Our Target	○	○	○	○	○	○	○	○

○: Secure or can be secure, ×: Insecure

In Chapter 2, we focus on CRHI (Challenge-Response Human Identification) as a countermeasure against real world adversaries. We evaluate the exact resistance of it against peeping attacks, and propose a challenge-control method. While the original CRHI resists only one peep for practical parameters, our challenge-controlled CRHI can resist more peeps.

In Chapter 3, we propose a further improvement of the resistance against peeps. Precisely, we propose how to limit the visible space of the decoded image of VSS (Visual Secret Sharing). We call it LVSVSS (Limiting Visible Space VSS). We evaluate the visibility of the space, and show that it is possible to transmit messages to a user in certain position. This means the combination of LVSVSS and CRHI can deal with the real world adversaries. While this scheme requires users to possess slides, it has no threat even if an adversary gets the slide.

In Chapter 4, we consider preventing attacks over network using PKCs (Public-Key Cryptosystems). Since (primitive) PKCs do not have the desirable properties required for ideal PKCs, we propose how to convert primitive PKCs into ideal ones. We evaluate the primitive PKCs based on the decoding problems, and show they can be ideally strong PKCs using our conversions.

In Chapter 5, we study PAKE (Password-Authenticated Key-Exchange) to remove the burden of both users and administrators on public-key and certificate management. We propose a simple scheme that is almost as efficient as the Diffie-Hellman key-exchange in both communication and computation costs, and prove that its security can be reduced to DDH (Decision Diffie-Hellman) problem under standard assumptions.

Chapter 6 presents a summary of this research and further work. The combination of CRHI, LVSVSS and the ideally strong PKC or PAKE can prevent both the real and network adversaries.

Chapter 2

Challenge-Response Human Identification

In this chapter, we focus on the attacks performed in the real world, especially peeping of the input processes of the passwords. One direction to solve this problem was proposed in [53], and its simplified version was proposed in [32]. We evaluate the exact immunity against peeping attacks on the generalized version of [32], and then propose how to improve it by controlling a history of challenges so that the number of effective responses should be intended values.

2.1 Overview

Current human identification methods using secret codes or passwords are not secure enough against peeping at the input process. To overcome this problem, some schemes have been proposed by several researchers.

One such scheme uses the Zero-Knowledge Interactive Proof [25, 70, 75, 76] or One-Time password [29, 30]. However, provers need to have some extra devices since this scheme require computational power to calculate one-way functions or/and an enough memory to store large amounts of secret information. Moreover, using extra devices means that the verifier authenticates the devices, not human provers themselves. Another one is a scheme in which provers do not need to have any extra devices, and that attackers cannot get provers' secret by a couple of peeps [53, 32]. This scheme relies only on information theoretical security, but on computational security. Provers have to be able to make responses by themselves, and have to be able to keep their secret in mind. We call this scheme Challenge-Response Human Identification (CRHI).

CRHI can be classified into two classes according to the set of challenges. One includes all the uniform mappings from the candidate set, where the provers' secrets belong, into the response set. We call this scheme CRHI using uniform mapping. The other includes only the limited number of mappings that hold a linear algebraic relationship between a response and a prover's secret [50]. We call this scheme CRHI using linear algebra in this thesis. It is not so difficult to estimate the resistance against peeping attacks on CRHI using linear algebra since the success probability of impersonation can be easily derived using linear algebra. On the other hand, it is very complicated to estimate that using uniform mapping since the success probability varies according to the combination of challenges that adversaries peeped and that are displayed to the adversary in impersonation phase.

Therefore, the resistance against peeping attacks is evaluated in a heuristic way about some parameters, when attackers try only 1 response [54]. And the resistance against continuous peeping attacks is evaluated, when the challenges are controlled by the verifier in a deterministic way [36]. However, the exact resistance or properties have not been evaluated, especially when attackers try nt responses and challenges are selected randomly. In this chapter, we evaluate various properties and resistance against peeping attacks on CRHI using uniform mapping.

2.2 Challenge-Response Human Identification Schemes and Peeping Attacks

2.2.1 Challenge-Response Human Identification

In CRHI, the following four kinds of entities (persons or machines) appear:

Prover	:	A proper person who can take the service.
Attacker	:	An entity that tries to take the service illegally.
Uncertain person	:	A person who is going to be verified (Prover or Attacker).
Verifier	:	An entity that verifies a person.

First, a protocol designer determines S which is a finite set that provers' secret information belongs to , F a finite mapping set, and A a finite set of responses, where the amount of the secret information has to be in prover's (human) memory capacity and provers have to be able to make responses by themselves. We show the relation among

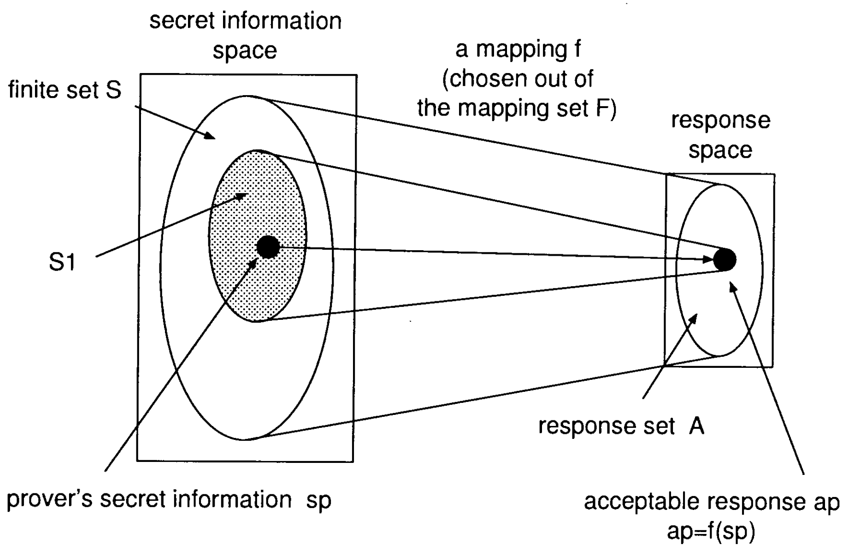
secret information space
finite set S
a mapping f (chosen out of the mapping set F)
response space
S1
response set A
prover's secret information sp
acceptable response ap ap=f(sp)

Figure 2.1: Relation among S, F and A

S, F and A in Fig. 2.1. Then, a prover selects an element from a finite set S as his/her secret information s_p. Both the prover and the verifier keep it secretly.

The protocol is composed of the following procedures.

{ A Protocol of CRHI }

1. **Challenge Process** The verifier selects an element (mapping) f (pairs of a and s, $\{a, s\}$, $a \in A$, $s \in S$) from F randomly and uniformly, and shows it to the uncertain person.

2. **Response Process** The prover (uncertain person) makes the correct response a_p by $a_p = f(s_p)$ (searching s_p from the pairs of a and s, {a,s}), and return it to the verifier.

3. **Verification Process** The verifier also makes the correct response $a_p{}'$ independently of the uncertain person, and compare it with a_p. If $a'_p = a_p$, the verifier accepts the uncertain person as a prover. Otherwise the verifier rejects him/her.

2.2.2 Peeping Attacks

Peeping attacks consist of the following two stages. In the first place, an attacker peeps at either only responses or the pairs of responses and their corresponding challenges, and

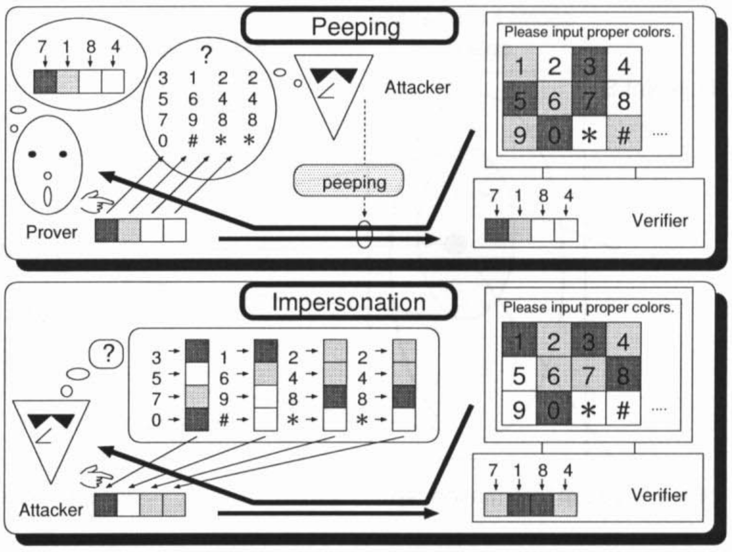

Figure 2.2: Example of CRHI and peeping attacks

then finds out the candidates for the secret information. We call this "peep stage." Then, he/she calculates the probability of each response in A by transforming the narrowed candidates to A according to the given challenge, and tries the acceptable responses. We call this "impersonation stage."

We show an example of CRHI and its peeping attacks in Fig 2.2. In the example, S is four figures of characters $\{0, \sim, 9, \#, *\}$, and A is four figures of colors.

2.3 Analysis in Peep Stage

2.3.1 Narrowing down candidates for prover's secret from only accepted responses

Let $P(s)$ be the probability of the prover's secret being s, and $(a_p^{(1)}, \cdots, a_p^{(n)})$ be the accepted n responses the attacker peeped. Then the probability of s after the attacker knows $(a_p^{(1)}, \cdots, a_p^{(n)})$ is given by the following equation:

$$P(s|a_p^{(1)}, \cdots, a_p^{(n)}) = \frac{P(a_p^{(1)}, \cdots, a_p^{(n)}|s)P(s)}{\sum_{\{s|s\in S\}} P(a_p^{(1)}, \cdots, a_p^{(n)}|s)P(s)}. \quad (2.1)$$

If all $(a_p^{(1)}, \cdots, a_p^{(n)}|s)$ are the same, then $P(s|a_p^{(1)}, \cdots, a_p^{(n)}) = P(s)$. Therefore the information about the prover's secret never leaks out.

2.3.2 Narrowing down candidates for prover's secret from ever accepted responses and their challenges

If an attacker knows accepted responses and their corresponding challenges, he/she can find out the candidates for the secret information.

Let a set of s satisfying the equation $a_p^{(i)} = f^{(i)}(s)$ be R_i. where $a_p^{(i)}$ and $f^{(i)}$ denote the i th response and its corresponding challenge the attacker peeped. A set of the candidates found out by n pairs of accepted responses and their challenges S_n is given by the equation $S_n = \cap_{i=1}^{n} R_i$. And the probability of s after n peeps is given by the following equation:

$$P(s|n) = \begin{cases} \frac{P(s)}{\sum_{S_n} P(s)} & (s \in S_n) \\ 0 & (s \notin S_n) \end{cases} . \tag{2.2}$$

If the probability of $P(s)$ is uniform, the equation (2.2) is expressed as follows:

$$P(s|n) = \begin{cases} \frac{1}{|S_n|} & (s \in S_n) \\ 0 & (s \notin S_n) \end{cases} , \tag{2.3}$$

where $|S_n|$ denotes the number of elements of S_n. That is, n peeps can reduce the amount of the information on the prover's secret from $\log_2 |S|$ to $\log_2 |S_n|$. $|S_n|$ varies according to the combination of the n accepted responses and their challenges the attacker peeped. Thus, we derive the probability distribution of $|S_n|$ and its expected value $E|S_n|$.

Usually, s and a are expressed with λ figures (characters) and mappings from s to a are made by transforming each figure of s to the corresponding figure of a. Let a set of characters of a figure of s and a be $\widehat{S}$ and $\widehat{A}$ respectively, and mapping from $\widehat{S}$ to $\widehat{A}$ be $\hat{f}$. Then the property of S_n can is derived from the property of $\widehat{S_n}$. So we consider the property density of $\widehat{S_n}$.

If $\hat{f}$ is uniform mapping, i.e. the numbers of $\hat{s}$ transformed into each $\hat{a}$ are the same) , $|\widehat{S_0}| = |\widehat{S}|$ and $|\widehat{S_1}| = |\widehat{S}|/|\widehat{A}|$. And the probability of the number of the candidates being k after n ($n \geq 2$) peeps is given by the following equation:

$$\begin{aligned} P(|\widehat{S_n}| = k) &= \sum_{|\widehat{S_{n-1}}|} P(|\widehat{S_n}|, |\widehat{S_{n-1}}|) \\ &= \sum_{|\widehat{S_{n-1}}|} P(|\widehat{S_n}| = k \,|\, |\widehat{S_{n-1}}| = x) P(|\widehat{S_{n-1}}| = x) \end{aligned} \tag{2.4}$$

If a challenge is chosen out of the set of all the uniform mapping from $\widehat{S}$ to $\widehat{A}$ at random, then the probability of k candidates to be x after one more peep is given by the following equation:

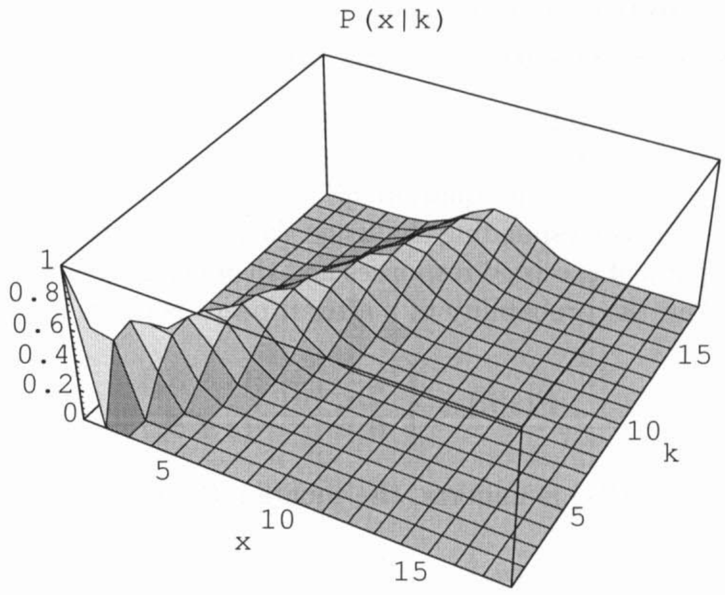

Figure 2.3: $P(|\widehat{S_{n+1}}| = x||\widehat{S_n}| = k)$ for $|\widehat{S}| = 36$ and $|\widehat{A}| = 2$, i.e. probability of k candidates to be x with one more peeping of an accepted response and its challenge

$$P(|\widehat{S_{n+1}}| = x||\widehat{S_n}| = k) = \frac{\begin{pmatrix} k-1 \\ x-1 \end{pmatrix}\begin{pmatrix} |\widehat{S}|-k \\ |\widehat{S}|/|\widehat{A}|-x \end{pmatrix}}{\begin{pmatrix} |\widehat{S}|-1 \\ |\widehat{S}|/|\widehat{A}|-1 \end{pmatrix}}. \tag{2.5}$$

The expected value of x, Ex, is given by the following equation,

$$Ex = \frac{(|\widehat{S}|/|\widehat{A}|-1)}{(|\widehat{S}|-1)}(k-1)+1 \tag{2.6}$$

since the correct one always remains there and the others remain there with probability of $(|\widehat{S}|/|\widehat{A}|-1)/(|\widehat{S}|-1)$. Note that one peeping approximately reduces k to $k/|\widehat{A}|$ when $|\widehat{S}| >> 1$.

By substituting equation (2.5) into equation (2.4), $P(|\widehat{S_n}| = k)$, the probability of the number of candidates to be k after n peeps, can be obtained. We show it in Fig. 2.3.2.

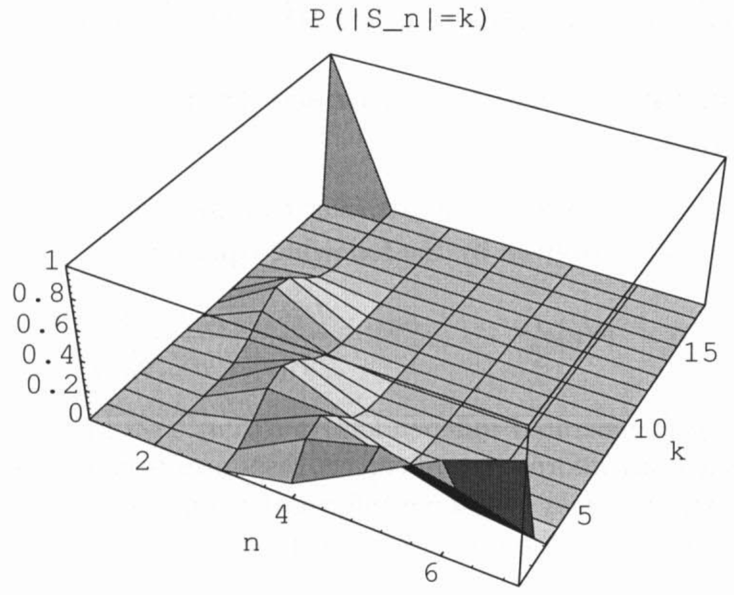

Figure 2.4: $P(|\widehat{S_n}| = k)$ for $|\widehat{S}| = 36$ and $|\widehat{A}| = 2$, i.e. probability of number of candidates to be k after n peeps

Note that the expected value of $|\widehat{S_n}|$ is approximately given by

$$E|\widehat{S_n}| \approx \left(\frac{|\widehat{S_1}| - 1}{|\widehat{S}| - 1}\right)^n (|\widehat{S}| - 1) + 1, \tag{2.7}$$

which can be derived from the following relationship

$$\begin{aligned} E|\widehat{S_n}| - 1 &\approx \left(\frac{|\widehat{S}|/|\widehat{A}| - 1}{|\widehat{S}| - 1}\right)(E|\widehat{S_{n-1}}| - 1) \\ &\approx \left(\frac{|\widehat{S}|/|\widehat{A}| - 1}{|\widehat{S}| - 1}\right)^n (|\widehat{S}| - 1). \end{aligned} \tag{2.8}$$

2.4 Analysis in Impersonation Stage

2.4.1 Guessing correct responses from only ever accepted responses

The probability of the responses an attacker can guess after he/she knows accepted responses $(a_p^{(1)}, \cdots, a_p^{(n)})$ is given by the following equation:

$$P(a|a_p^{(1)}, \cdots, a_p^{(n)}) = \frac{P(a, a_p^{(1)}, \cdots, a_p^{(n)})}{P(a_p^{(1)}, \cdots, a_p^{(n)})}. \tag{2.9}$$

If each challenge is generated independently, $P(a|a_p^{(1)}, \cdots, a_p^{(n)}) = P(a)$. Therefore attackers cannot get any information of correct responses from ever accepted responses. On the contrary, attackers may get some information on the correct responses of the next rounds, if they depend on the previously accepted ones.

2.4.2 Guessing correct responses from candidates

Attackers can estimate the probability of responses by transforming candidates for the prover's secret to responses. The probability of an attacker being accepted by a response a is given by the following equation :

$$P(a|n) = \sum_{S_a(n)} P(s|n), \tag{2.10}$$

where $S_a(n) = \{s|a = f(s), s \in S_n\}$, i.e. a subset of S_n transformed to the response a. If the prover selects his/her secret s_p from S uniformly, the equation (2.10) can be expressed as follows:

$$P(a|n) = \frac{|S_a(n)|}{|S_n|} \tag{2.11}$$

$P(a|n)$ is derived by looking over how S_n is divided by challenges.

Let s and a be expressed with λ figures (characters) and mapping from s to a be made by transforming each figure of s to the corresponding figure of a. Let $\{pn_1, \cdots, pn_i, \cdots, pn_{|\widehat{A}|}\}$ be a list of the number of candidates assigned to a response arranged in order of the size, and $\widehat{A}_{part}(k)$ be the set of all the lists. That is:

$$\begin{aligned} \widehat{A}_{part}(k) &= \{\{pn_1, \cdots, pn_i, \cdots, pn_{|\widehat{A}|}\}| \\ & \quad 0 \le pn_i \le |\widehat{S}|/|\widehat{A}|, \sum_i pn_i = k, pn_1 \ge pn_i \ge pn_{|\widehat{A}|}\}. \end{aligned} \tag{2.12}$$

The i th highest probability of a character in each figure of the response after the partition of $\widehat{S_n}(|\widehat{S_n}| = k) = \{pn_1, \cdots, pn_{|\widehat{A}|}\}$ is given by:

$$P_{\widehat{A}max(i)}(\hat{a}|\widehat{A}_{part}(k) = \{pn_1, \quad \cdots \quad, pn_{|\widehat{A}|}\}) = \frac{pn_i}{k}. \tag{2.13}$$

For instance, when 6 candidates are assigned to 3 responses, the following partition is possible.

$$\begin{aligned}&\widehat{A}_{part}(6)\\ &= \{\{6,0,0\},\{5,1,0\},\{4,2,0\},\{3,3,0\},\{4,1,1\},\{3,2,1\},\{2,2,2\}\}\end{aligned} \tag{2.14}$$

If partition $\{4, 2, 0\}$ has happened, probability that an attacker is accepted with the most acceptable response is $4/6$, and second one is $2/6$.

If the mapping set $|\widehat{F}|$ at the figure includes all the uniform mappings from $|\widehat{S}|$ to $|\widehat{A}|$ and every challenge is selected uniformly, probability that k candidates are divided as $\{pn_1, \cdots, pn_{|\widehat{A}|}\}$ is given by the following equation:

$$\begin{aligned}&P(\widehat{A}_{part}(k) = \{pn_1, \cdots, pn_i, \cdots, pn_{|\widehat{A}|}\})\\ &= \frac{|\widehat{A}|!}{|\widehat{F}|}\prod_{i=1}^{|\widehat{A}|}\binom{k - \sum_{j=1}^{i-1} pn_j}{pn_i}\cdot\binom{|\widehat{S}| - k - \sum_{j=1}^{i-1}(|\widehat{S}|/|\widehat{A}| - pn_j)}{|\widehat{S}|/|\widehat{A}| - pn_i},\end{aligned} \tag{2.15}$$

where

$$|\widehat{F}| = \frac{|\widehat{S}|!}{((|\widehat{S}|/|\widehat{A}|)!)^{|\widehat{A}|}}. \tag{2.16}$$

We can derive any probability about responses by using the probability of partition. For instance, we show the probability of pn_1 when $|\widehat{A}| = 2$, $|\widehat{S}| = 36$ in Fig. 2.4.2.

The probability of an attacker being accepted by trying the most acceptable response Let $P_{Amax(1)}(a|n)$ denote the probability of an attacker being accepted by trying the most acceptable response after transforming the candidates for the prover's secret found by n pairs of ever accepted responses and the corresponding challenges. It is given by:

$$P_{Amax(1)}(a|n) \quad = \quad (P_{\widehat{A}max(1)}(\hat{a}|n))^{\lambda}, \tag{2.17}$$

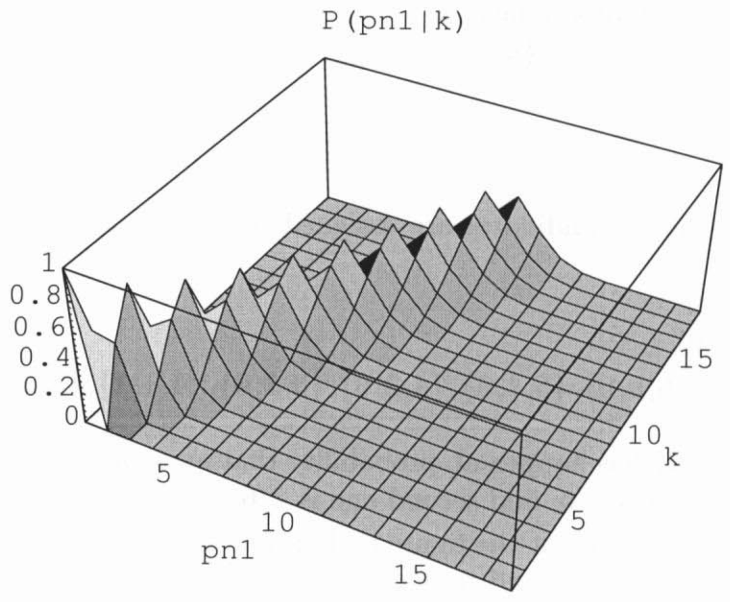

Figure 2.5: $P(pn_1|k)$ for ($|\widehat{A}| = 2$ and $|\widehat{S}| = 36$), i.e. probability that pn_1 out of k candidates are mapped onto most acceptable response

where

$$P_{\widehat{A}max(i)}(\hat{a}|n) = \sum_{\widehat{A}_{part}(|\widehat{S_n}|)} \sum_{|\widehat{S_n}|} P_{\widehat{A}max(i)}(\hat{a}|\widehat{A}_{part}(|\widehat{S_n}|)) \cdot P(\widehat{A}_{part}(|\widehat{S_n}|))P(|\widehat{S_n}|). \tag{2.18}$$

We show $P_{Amax(1)}(a|n)$ in Fig. 2.4.2.

The number of effective responses It is useful to get hold of the number of the responses which have probability to be accepted (effective responses) to roughly estimate the security. The number of effective responses after n peeps $|A_n|$ is derived from the same discussion as above.

2.4.3 When an attacker tries t most acceptable responses after n peeps

Let GL, OB $= n$, and TR $= t$ be the events that an attacker is accepted illegally, an attacker finds out the candidates from n pairs of ever accepted responses and their

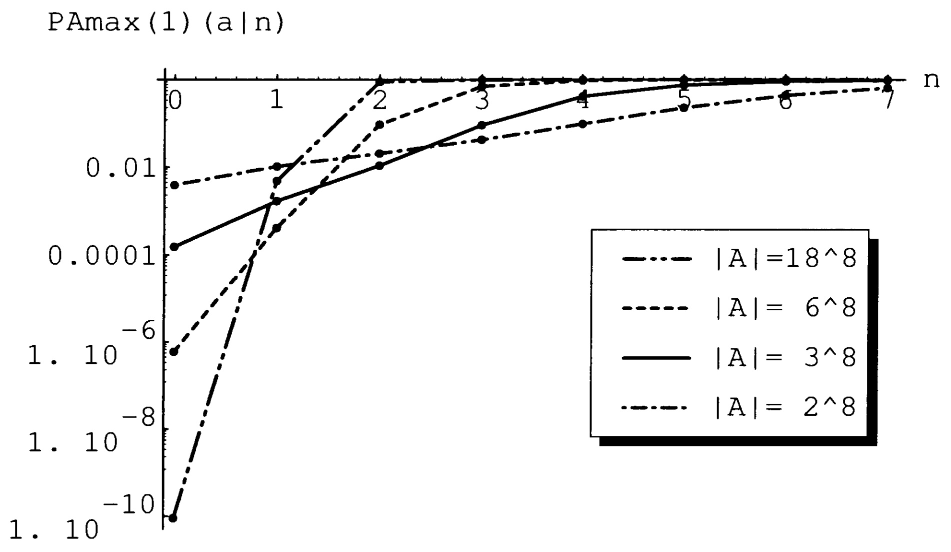

Figure 2.6: $P_{Amax(1)}(a|n)$ for $\lambda = 8$ and $|S| = 36^{\lambda}$, i.e. probability of an attacker being accepted by trying most acceptable response after n peeps

corresponding challenges, and an attacker has been rejected by ever tried $t-1$ responses and is going to try the t th response, respectively.

When an attacker tries only the most acceptable response

When an attacker tries only the most acceptable response after n peeps, the probability $P(\mathrm{GL}, \mathrm{TR} = 1|\mathrm{OB} = n)$ is given by:

$$P(\mathrm{GL}, \mathrm{TR} = 1|\mathrm{OB} = n) \quad = \quad P_{Amax(1)}(a|n). \tag{2.19}$$

When an attacker tries some of the most acceptable responses

When an attacker tries the t most acceptable responses after n peeps, its probability $P(\mathrm{GL}, \mathrm{TR} = t|OB = n)$ is bounded by:

$$P(\mathrm{GL}, \mathrm{TR} = t|\mathrm{OB} = n) \quad \leq \quad 1 - (1 - P_{Amax(1)}(a|n))^t \tag{2.20}$$

When an attacker tries all the acceptable responses

When an attacker tries all the acceptable responses (at most S_n responses), the probability that he/she is accepted is 1. Thus the probability is not necessarily suitable for the measurement of the resistance under this situation. We propose to use the following equation to evaluate the resistance under this situation:

$$L_A(n) = \sum_{i=1}^{|S_n|} P(\mathrm{GL}, \mathrm{TR} = i | \mathrm{OB} = n) \cdot i \tag{2.21}$$

where $P(\mathrm{GL}, \mathrm{TR} = i | \mathrm{OB} = n)$ denotes the probability that an attacker is accepted by i th response he/she tried after n peeps. $L_A(n)$ means the expected value of the number of responses that an attacker has to try to be accepted, when he/she tries all the acceptable responses one by one. We call it MSL (Mean Searching Length) in short.

Situation that verifiers give the same challenge, when an uncertain person returns the wrong response In this situation, $P(\mathrm{GL}, \mathrm{TR} = i | \mathrm{OB} = n) = P_{Amax(i)}(a|n)$, and thus

$$L_A(n) = \sum_{i=1}^{|A_n|} P_{Amax(i)}(a|n) \cdot i. \tag{2.22}$$

However, it may be heavy to calculate $L_A(n)$ with this equation for large parameters. Therefore we approximate it as follows. Let $H(S_n)$ and $H(A_n)$ denote the amount of information of S_n and A_n respectively. They have a relationship $H(A_n) = H(S_n) - H(S_{n+1}) \approx -\log_2 \frac{|S_{n+1}|}{|S_n|}$. That is, we can suppose that the number of the acceptable responses with high probability is approximately given by $|A'_n| = 2^{H(A_n)} = \frac{|S_n|}{|S_{n+1}|}$. When the attacker tries $|A'_n|$ responses one by one, he/she can be accepted with $(|S_n|/|S_{n+1}| + 1)/2$ trials on average. Therefore,

$$L_A(n) \approx \frac{E|S_n|/E|S_{n+1}| + 1}{2}. \tag{2.23}$$

In this connection, $L_A(n)$ satisfies the following relationship.

$$\frac{\frac{1}{P_{Amax(1)}(a|n)} + 1}{2} < L_A(n) < \frac{E|A_n| + 1}{2} \tag{2.24}$$

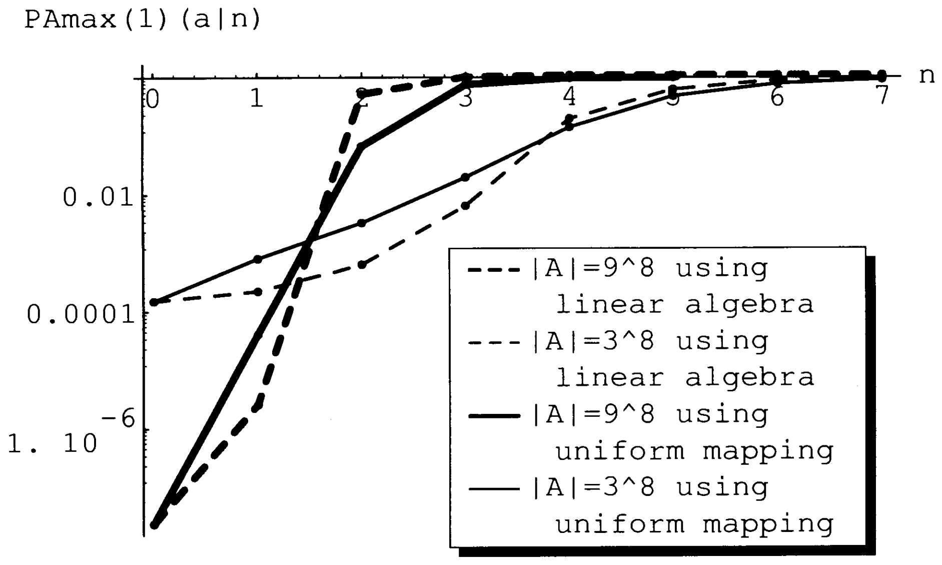

Figure 2.7: Comparison between linear algebra schemes and uniform mapping schemes: $P_{Amax(1)}(a|n)$ for $\lambda = 8$ and $|S| = 81^{\lambda}$

Situation that verifiers give another challenge after an uncertain person returns the wrong response When verifiers give the same challenge after an uncertain person returns the wrong response, the number of acceptable responses decreases one by one as he tries an acceptable response. Compared with this, the decrease of acceptable responses is slow when verifiers give another challenge. Especially, when the number of candidates is large, we can suppose that the number of acceptable responses is still $|A'_n|$ after the attacker tries some acceptable responses.

Therefore,

$$\dot{L}_A(n) \approx 2L_A(n). \tag{2.25}$$

In this connection, $E\dot{L}_A(n)$ has the following relationship.

$$\frac{1}{P_{Amax(1)}(a|n)} < E\dot{L}_A(n) < E|A_n| \tag{2.26}$$

2.4.4 Comparison between linear algebra schemes and uniform mapping schemes

In this subsection, we make a comparison between the linear algebra scheme [51] and the uniform mapping scheme [32]. The linear algebra scheme uses linear algebra to transform S into A instead of using uniform mapping to make the evaluation easier.

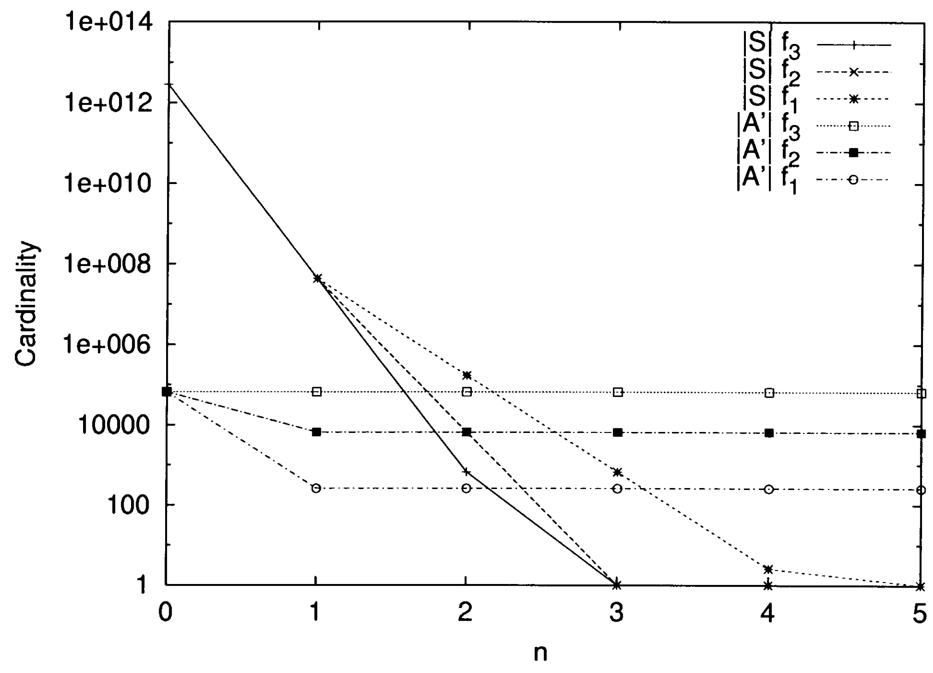

Figure 2.8: Relationship between $|S|$ and $|A'|$: Example I
($\lambda = 8$, $|S| = 36^\lambda$, $|A| = 4^\lambda$)

$$\begin{aligned} f_1 &: |A'_n| = 2^8 = 256 \\ f_2 &: |A'_n| = 3^8 = 6561 \\ f_3 &: |A'_n| = |A| = 4^8 = 6.6 \times 10^4 \end{aligned}$$

Let p and r denote a prime and a positive integer, respectively. Then, $\hat{a}$ in the linear algebra scheme is an element in Galois field $GF(p^r)$. $\hat{f}$ and $\hat{s}$ in it are a v dimensional row vector and a v dimensional column vector in $GF(p^r)$, respectively. The map from $\hat{s}$ to $\hat{a}$ is given by

$$\hat{a} = \hat{f} \cdot \hat{s}. \tag{2.27}$$

The main difference between the linear algebra scheme and the uniform mapping scheme is their mapping sets. While the uniform mapping scheme has $((|\widehat{S}|!)/(((|\widehat{S}|/|\widehat{A}|)!)^{|\widehat{A}|}))^\lambda$ mappings in total, the linear algebra scheme has only $p^{rv\lambda} = |S|$ mappings (if the 0 vectors are allowed). Fig. 2.4.4 shows the comparison of $P_{A_{max}(1)}(a|n)$ between them. While the linear algebra scheme seems to have a better performance, it has the following disadvantage caused by the 0 vectors in $\widehat{F}$ and $\widehat{S}$. That is, if we allow the 0 vector for $\hat{f}$, adversaries may wait until 0 is chosen for $\hat{f}$. If a prover chooses 0 for $\hat{s}$, $\hat{a}$ becomes

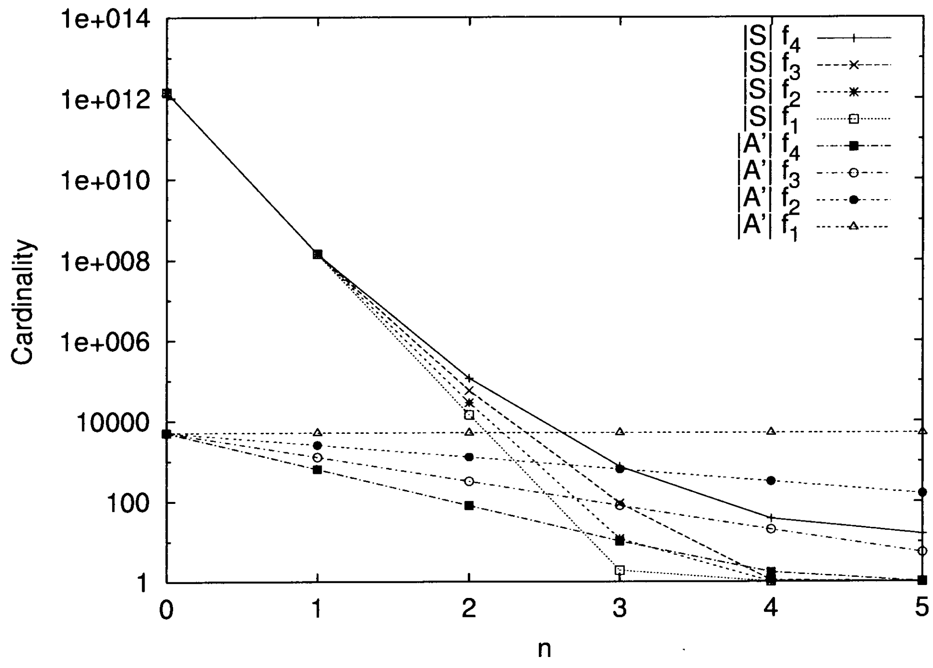

Figure 2.9: Relationship between $|S|$ and $|A'|$: Example II
($\lambda = 8$, $|S| = 36^{\lambda}$, $|A| = 10000$)

f_1 : $P_l = 1$
f_2 : $P_l = 1/2$
f_3 : $P_l = 1/4$
f_4 : $P_l = 1/8$

always 0 regardless $\hat{f}$. On the other hand, if we remove the 0 vectors from them the probability of $\hat{a}$ to be accepted is biased. Precisely, $P(\hat{a} = 0) = (|\widehat{S}|/|\widehat{A}| - 1)/(|\widehat{S}| - 1)$ and $P(\hat{a} = x) = (|\widehat{S}|/|\widehat{A}|)/(|\widehat{S}| - 1)$, $(x \neq 0)$. This bias is not negligible since $|\widehat{S}|/|\widehat{A}|$ is usually small.

2.5 Challenge Control

In this section, we propose how to improve the resistance against peeping by controlling a history of challenges so that the number of effective responses $|A_n|$ should take in an intended value $|A'_n|$. First of all, we discuss how $|A'_n|$ should be set in in the next subsection.

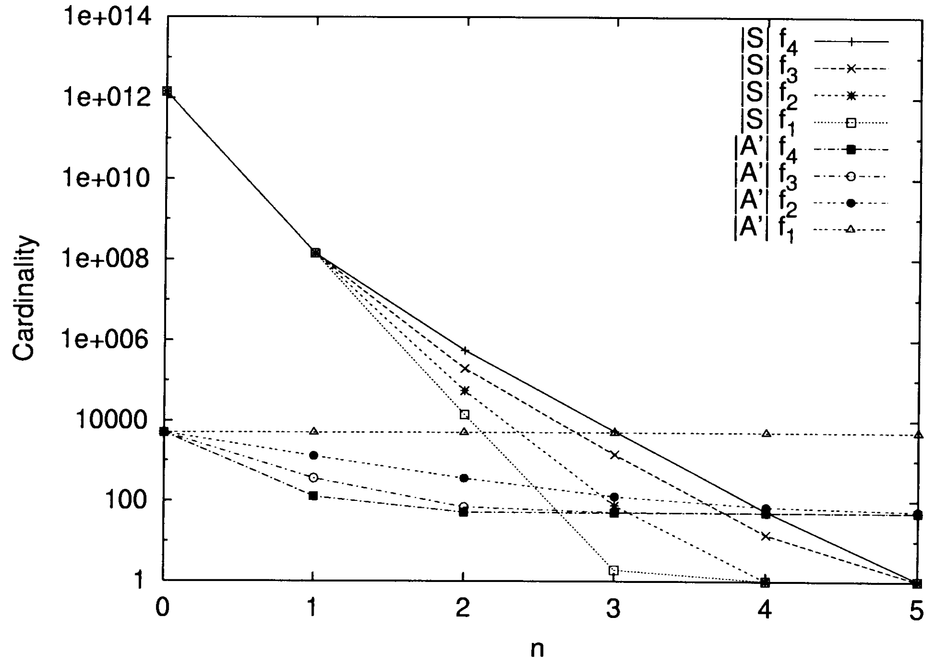

Figure 2.10: Relationship between $|S|$ and $|A'|$: Example III
($\lambda = 8$, $|S| = 36^{\lambda}$, $|A| = 10000$, $P_{safe} = 100$)

$f_1 \; : \; P_l = 1$
$f_2 \; : \; P_l = 1/4$
$f_3 \; : \; P_l = 1/16$
$f_4 \; : \; P_l = 1/64$

2.5.1 Relationship between $|A'|$ and $|S|$

For given $|S|$ and intended $|A'_n|$, practically available $|A_n|$ is upper-bounded by

$$|A_n| \;=\; \min\{|A'_n|, |S_n|\}. \tag{2.28}$$

We show some examples of the relationship between $|A'_n|$ and the corresponding $|S_n|$.

Example I: $|A'_n|$ is set in a constant value for $n \geq 1$. The corresponding $|S_n|$ and $|A'_n|$ are illustrated in Fig. 2.8.

Example II: $|A'_n|$ decreases gradually according to $|A'_n| = |A| \cdot P_l^n$. The corresponding $|S_n|$ and $|A'_n|$ are illustrated in Fig. 2.9.

Example III: $|A'_n|$ decreases gradually according to $|A'_n| = (|A| - P_{safe}) \cdot P_l^n + P_{safe}$. The corresponding $|S_n|$ and $|A'_n|$ are illustrated in Fig. 2.10.

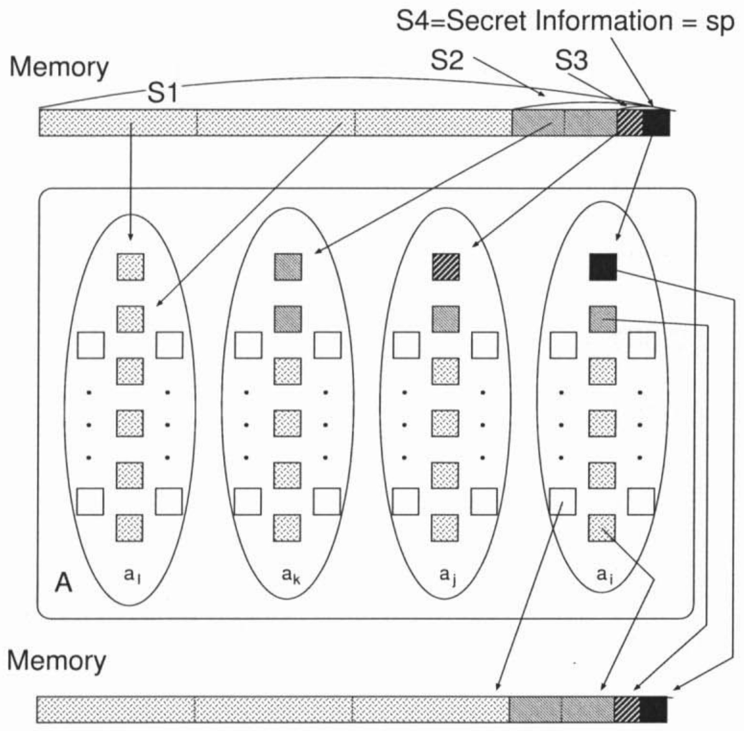

Figure 2.11: How to keep a history of challenges for $(|\widehat{A}_0|, |\widehat{A}_1|, |\widehat{A}_2|, |\widehat{A}_3|) = (4, 4, 3, 2)$

Let N be the number of $|\widehat{A}'_n|$'s that are greater than 1. Then $|\widehat{A}_n| = |\widehat{A}'_n|$ holds when $|\widehat{S}|$ is the composite number of them, i.e.

$$|\widehat{S}| \quad = \quad \prod_{n=0}^{N-1} |\widehat{A}'_n|. \tag{2.29}$$

2.5.2 How to Control $|A'_n|$

For simplicity, we assume (2.29). The following is the steps to keep a history of challenges with a small memory, and Fig. 2.11 illustrates an example of them for $(|\widehat{A}_0|, |\widehat{A}_1|, |\widehat{A}_0|, |\widehat{A}_1|) = (4, 4, 3, 2)$.

Step 1: Assign each member in $\widehat{S}_N$ to $|\widehat{A}_N|$ responses at random. (Note that $|\widehat{S}_N| = |\widehat{A}_N|$.)

Step 2: From $n = N - 1$ to 0

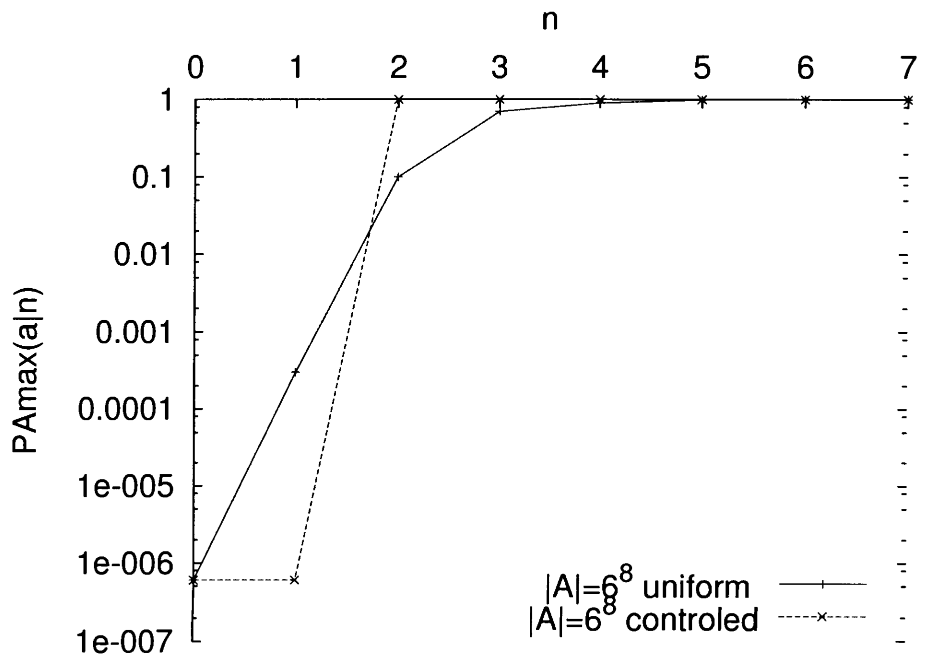

Figure 2.12: Comparison of $P_{A_{max}(1)}(a|n)$ between the uniform mapping scheme for $|\widehat{A}| = 6$ and the challenge-controlled scheme for $(|\widehat{A}_0|, |\widehat{A}_1|) = (6,6)$: $|\widehat{S}| = 36$, $\lambda = 8$

- Randomly assign each member in $\widehat{S}_n - \widehat{S}_{n+1}$ to $|\widehat{A}_n|$ responses, (where S_{n+1} members have already been assigned on the $|\widehat{A}_{n+1}|$ responses), so that $|\widehat{A}_{n+1}|$ members in $\widehat{S}_n$ should be assigned to the $|\widehat{A}_n|$ responses uniformly.

Step 3: If a prover is accepted, move the subsets in $\widehat{S}_n$ $(0 \leq n \leq N-1)$, which are assigned to the accepted response to $\widehat{S}_{n+1}$, respectively.

Fig. 2.12 and 2.13 show the comparison of $P_{A_{max}(1)}(a|n)$ between the uniform mapping schemes and the challenge-controlled schemes. As you can see, the challenge-controlled scheme improves the resistance to a couple peeps for practical parameters. Even though the resistance after several peeps seems reversal, the difference is negligible since both are in the range of on-line exhaustive search.

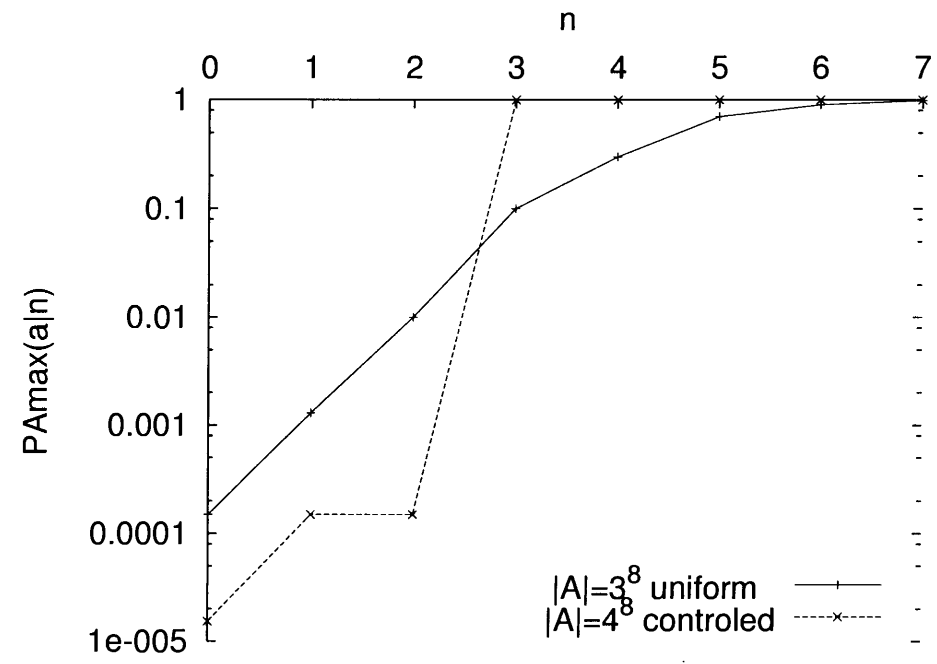

Figure 2.13: Comparison of $P_{A_{max}(1)}(a|n)$ between the uniform mapping scheme for $|\widehat{A}| = 3$ and the challenge-controlled scheme for $(|\widehat{A}_0|, |\widehat{A}_1|, |\widehat{A}_2|) = (4, 3, 3)$: $|\widehat{S}| = 36$, $\lambda = 8$

2.6 Summary

We theoretically analyzed the properties of the security against peeping attacks on challenge-response type human identification schemes. For any given parameters, λ, $|\widehat{A}|$ and $|\widehat{S}|$ s.t. $|\widehat{S}| = c|\widehat{A}|$ for a natural number c, we can estimate the properties of them using our results.

We then proposed how to improve the properties by controlling the history of challenges so that the number of effective responses after n peeps should take an intended curve. We showed some examples of the curves. Using our proposal, one can design the resistance to the continuous n peeps as they want within the capacity of the CRHI, $|S|$.

The further study will be the estimation of the following items, which will help find optimum parameters of λ, $|\widehat{A}|$, $|\widehat{S}|$ and the life time of a prover's secret.

Prover's burden The larger $|\widehat{S}|$ and λ, and the shorter the life time of provers' secrets would enhance the security of the system. It, however, increases the burden of the provers. We need to clarify how much burden a prover can accept to gain the security.

Damage In order to optimize the life time of the provers' secrets, we need to estimate both mental and physical damage after the compromise of the secrets, and the risk of using the same secrets in a long term.

Environment The expected damage depends on the environment where the system is placed, e.g. how often the system gets attacks, what kinds of attacks the system gets, how often provers use the system, etc. We need to estimate these parameters precisely.

The optimum value of $|\widehat{S}|$, λ and the life time of the prover's secret would be given by the meeting point of the prover's burden and the avoidable damage. $|\widehat{A}|$ and the resistance curve should be set so that the estimated damage to be minimum.

Chapter 3

Application of Visual Secret Sharing

This chapter also deals with the peeping attacks, but in a different way from Chapter 2. We propose a new usage of visual secret sharing, which limits the space from which one can see the decoded image of it. We investigate the visibility of the decoded image to the viewpoint, and categorize it according to the visibility. We show that the combination of our proposal with CRHI (Challenge Response Human Identification) prevents the peeping attacks perfectly with small cost.

3.1 Overview

It is very dangerous to trust only one person or only one organization to manage very important information. To deal with these kinds of situations, a scheme to share a secret with some members, called a secret sharing scheme or a (k, n) threshold scheme, was proposed by A. Shamir [69]. In a (k, n) threshold scheme, a secret is divided into n pieces. Each single piece looks like random data by itself. In order to decode the secret, members have to gather k pieces. That is, k persons' permission is required to decode the secret. Since then, various studies on secret sharing schemes have been carried out. In particular, visual secret sharing schemes (VSS), originally proposed by M. Naor and A. Shamir [60], are very interesting. In these schemes, members who have shared a secret can decode it without help of computers in the decoding process. Shared secret (image) are printed on transparencies as random patterns. Members can decode the secret (image) by stacking some of them, and see it.

However, even if one uses these secret sharing schemes, once an attacker peeps at the decoded image, it might be leaked out easily. To deal with this, some people may decode it after confirming that no attacker is around. However, even though that can be confirmed, preoccupation about peeping still exists. A video camera may be set on somewhere secretly. Some people may decode it after covering it by a piece of cloth, or a corrugated carton, or by hands. However, it is troublesome to cover it with worrying about others' eyes and cameras every time a secret is decoded. Moreover, watching something secretly is enough to arise suspicious of immoral behavior.

In this chapter, we propose new usage of visual secret sharing schemes. We call this scheme limiting the visible space visual secret sharing schemes (LVSVSS). That is, we use the VSS to limit the space from which one can see a decoded image. We investigate the visibility of the decoded image when the viewpoint is changed, and categorized the space where the viewpoint belongs according to the visibility. Finally, we consider the application of LVSVSS to human identification, and propose a secure human identification scheme. The proposed human identification scheme is secure against peeping, and can detect simple fake terminals. Moreover, it can be actualized easily at a small cost.

3.2 How to Limit Visible Space

The principle of limiting the visible space is very simple (see Fig.3.1). The patterns are printed on transparencies so that an image can be decoded when a space is left between the transparencies. The separation of the transparencies make the space from which the decoded image can be seen smaller. We call this space "the visible space". Attackers out of the visible space cannot see it.

Let the point from which the decoded image can be seen correctly be the origin of the coordinate axes, and the x, y, z axes are set. Transparency 1 is located on $x = x_1$ and transparency 2 on $x = x_2$. Let the points overlapping each other be (x_1, y_1, z_1) (transparency 1) and (x_2, y_2, z_2) (transparency 2), when the viewpoint is on the origin. Then the following equations are held:

$$y_1 = \frac{x_1}{x_2} y_2 \, , \qquad z_1 = \frac{x_1}{x_2} z_2 . \tag{3.1}$$

3.3 Relationship between Viewpoint and Distortion

For simplicity, we consider the $x - y$ plane where $z = 0$ (see Fig.3.2). Let a function to calculate a point $y_1^{'}$ on $x = x_1$ overlapping with $(x_2, y_2, 0)$ when you watch the point $(x_2, y_2, 0)$ from $(v_x^{'}, v_y^{'}, 0)$ be $f(v_x^{'}, v_y^{'}, x_2, y_2, x_1)$, and a function to calculate the difference

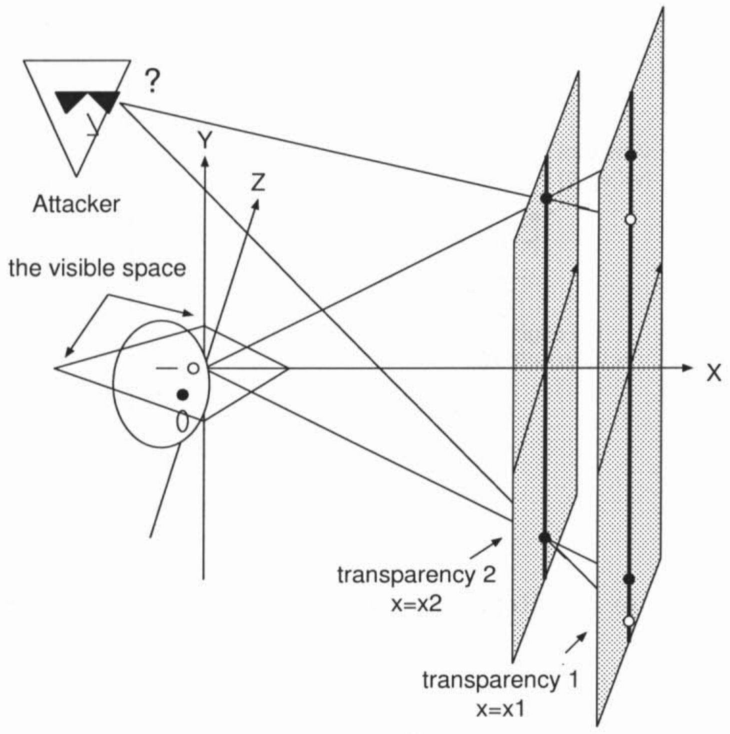

Figure 3.1: How to limit visible space

$g_y = y_1' - y_1$ be $g(v_x', v_y', x_2, y_2, x_1)$. Then, the functions are defined as follows:

$$\begin{aligned} f(v_x', v_y', x_2, y_2, x_1) &= a(v_x', x_2, x_1)y_1 + b(v_x', x_2, x_1, v_y') \\ &= a(v_x', x_2, x_1)(y_1 - b'(v_x', x_1, v_y')) + b'(v_x', x_1, v_y'), \end{aligned} \tag{3.2}$$

$$\begin{aligned} g(v_x', v_y', x_2, y_2, x_1) &= (a(v_x', x_2, x_1) - 1)y_1 + b(v_x', x_2, x_1, v_y') \\ &= a'(v_x', x_2, x_1)(y_1 - b'(v_x', x_1, v_y')), \end{aligned} \tag{3.3}$$

where

$$a(v_x', x_2, x_1) = \frac{(x_1 - v_x')x_2}{(x_2 - v_x')x_1} \tag{3.4}$$

$$a'(v_x', x_2, x_1) = \frac{(x_1 - x_2)v_x'}{(x_2 - v_x')x_1} \tag{3.5}$$

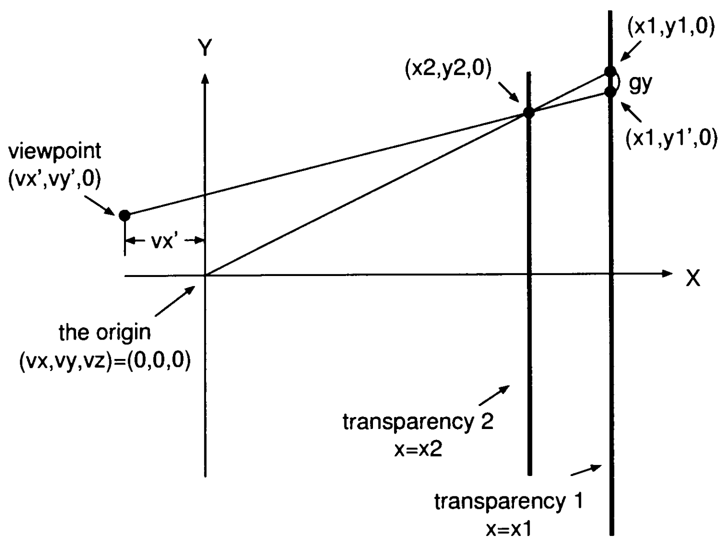

Figure 3.2: Relationship between viewpoint and distortion

$$b(v_x', x_2, x_1, v_y') = \frac{(x_2 - x_1)v_y'}{(x_2 - v_x')} \tag{3.6}$$

$$b'(v_x', x_1, v_y') = \frac{x_1 v_y'}{v_x'}. \tag{3.7}$$

As a matter of course:

$$y_1' = f(v_x', v_y', x_2, y_2, x_1) \tag{3.8}$$
$$g_y = g(v_x', v_y', x_2, y_2, x_1). \tag{3.9}$$

In the same way as the $x - y$ plane on $z = 0$, z_1' and g_z on the $x - z$ plane where $y = 0$ can be calculated.

$$z_1' = f(v_x', v_z', x_2, z_2, x_1) \tag{3.10}$$
$$g_z = g(v_x', v_z', x_2, z_2, x_1) \tag{3.11}$$

Therefore, a point on $x = x_1$ overlapping with a point (x_2, y_2, z_2) is (x_1, y_1', z_1'), and the vector $\vec{g}$ from (x_1, y_1, z_1) to (x_1, y_1', z_1') is $(0, g_y, g_z)$.

3.3.1 When viewpoint is on $y - z$ plane of $x = 0$

When the viewpoint is on the $y - z$ plane where $x = 0$, $a(v_x^{'}, x_2, x_1)$ in the equation (3.2) and (3.3) becomes 1. Therefore,

$$\begin{aligned}\vec{g} &= (0, b(0, x_2, x_1, v_y^{'}), b(0, x_2, x_1, v_z^{'})) \\ &= (0, \frac{x_1 - x_2}{x_2}(-v_y^{'}), \frac{x_1 - x_2}{x_2}(-v_z^{'})). \end{aligned} \tag{3.12}$$

It can be seen that the distortion vector $\vec{g}$ is independent from the points on the transparencies. So it looks as if all the points on the transparency 2 drifted from the corresponding points on the transparency 1 for the same length.

3.3.2 When viewpoint is in space of $x < x_2$ except $x = 0$

When the viewpoint is in the space $x < x_2$ except $x = 0$, the distortion vector $\vec{g}$ is as follows:

$$\begin{aligned}\vec{g} &= (0, a^{'}(v_x^{'}, x_2, x_1)(y_1 - b^{'}(v_x, x_1, v_y^{'})), a^{'}(v_x^{'}, x_2, x_1)(z_1 - b^{'}(v_x, x_1, v_z^{'}))) \\ &= (0, \frac{(x_1 - x_2)v_x^{'}}{(x_2 - v_x^{'})x_1}(y_1 - \frac{x_1 v_y^{'}}{v_x^{'}}), \frac{(x_1 - x_2)v_x^{'}}{(x_2 - v_x^{'})x_1}(z_1 - \frac{x_1 v_z^{'}}{v_x^{'}})) \end{aligned} \tag{3.13}$$

where $v_x^{'} \neq 0$. Therefore, it looks as if all the points on the transparency 2 radially drifted from the corresponding points on the transparency 1. The center is $(x_1, x_1 v_y^{'}/v_x^{'}, x_1 v_z^{'}/v_x^{'})$ and the length of the drift is $((x_1 - x_2)v_x^{'})/((x_2 - v_x^{'})x_1)$ times longer than the distance between the center and the point on the transparency 1.

3.4 Relationship between Shift and Contrast

We consider $(2, 2)$ threshold schemes. Transparencies consist of square cells with sides c. Each of cell has 2×2 square pixels with sides d. A pixel is black or transparent. Therefore, when the two transparencies are stacked each other, it looks like black if 4 pixels are black in the cell, and white if 2 pixels are black. Let me call the types of black cells and white cells B_i and W_i $(1 \leq i \leq 6)$ respectively. We show all kinds of the cells in Fig.3.3 with the situation of the shift. In order to measure the visibility, we define the density as the rate of black area in a cell first. Then let the length of the shift be g_z and g_y respectively, and the expected value of the density where the shift

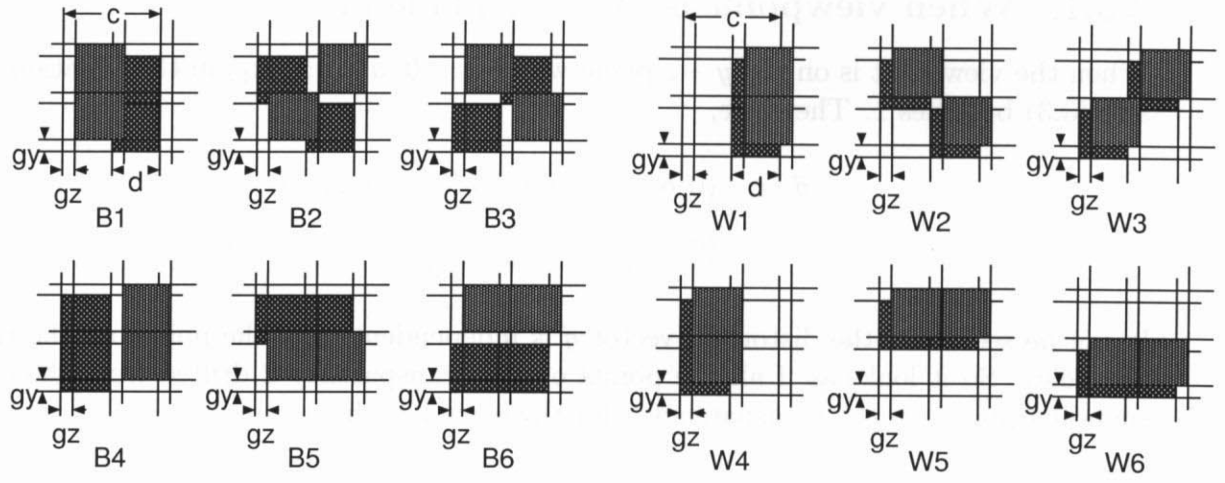

Figure 3.3: All kinds of cells with shift

is (g_y, g_z) be $EG_{B_i}(g_y, g_z)$ and $EG_{W_i}(g_y, g_z)$ respectively. $EG_{B_i}(g_y, g_z)$ and $EG_{W_i}(g_y, g_z)$ $(0 \le g_y, g_z < 2d)$ can be expressed as follows:

$$EG_{B_1}(g_y, g_z) = \begin{cases} 1 - \frac{1}{4d}g_z - \frac{1}{8d}g_y + \frac{1}{8d^2}g_z g_y & (0 \le g_z < d, 0 \le g_y < 2d) \\ \frac{3}{4} & (d \le g_z, 0 \le g_y) \end{cases} \tag{3.14}$$

$$EG_{B_2}(g_y, g_z) = \begin{cases} 1 - \frac{3}{8d}(g_y + g_z) + \frac{5}{8d^2}g_z g_y & (0 \le g_z < d, 0 \le g_y < d) \\ \frac{1}{2} + \frac{1}{8d}g_z + \frac{1}{2d}g_y - \frac{1}{4d^2}g_z g_y & (d \le g_z < 2d, 0 \le g_y < d) \\ \frac{1}{2} + \frac{1}{2d}g_z + \frac{1}{8d}g_y - \frac{1}{4d^2}g_z g_y & (0 \le g_z < d, d \le g_y < 2d) \\ \frac{5}{4} - \frac{1}{4d}(g_z + g_y) + \frac{1}{8d^2}g_z g_y & (d \le g_z < 2d, d \le g_y < 2d) \end{cases} \tag{3.15}$$

$$EG_{B_3}(g_y, g_z) = \begin{cases} 1 - \frac{3}{8d}(g_y + g_z) + \frac{1}{2d^2}g_z g_y & (0 \le g_z < d, 0 \le g_y < d) \\ \frac{1}{2} + \frac{1}{8d}g_z + \frac{1}{4d}g_y - \frac{1}{8d^2}g_z g_y & (d \le g_z < 2d, 0 \le g_y < d) \\ \frac{1}{2} + \frac{1}{4d}g_z + \frac{1}{8d}g_y - \frac{1}{8d^2}g_z g_y & (0 \le g_z < d, d \le g_y < 2d) \\ \frac{3}{4} & (d \le g_z < 2d, d \le g_y < 2d) \end{cases} \tag{3.16}$$

$$EG_{B_4}(g_y, g_z) = \begin{cases} 1 - \frac{1}{8d}g_y - \frac{1}{2d}g_z + \frac{1}{4d^2}g_z g_y & (0 \le g_z < d, 0 \le g_y < 2d) \\ \frac{1}{4} + \frac{1}{4d}(g_y + g_z) - \frac{1}{8d^2}g_z g_y & (d \le g_z, 0 \le g_y < 2d) \end{cases} \tag{3.17}$$

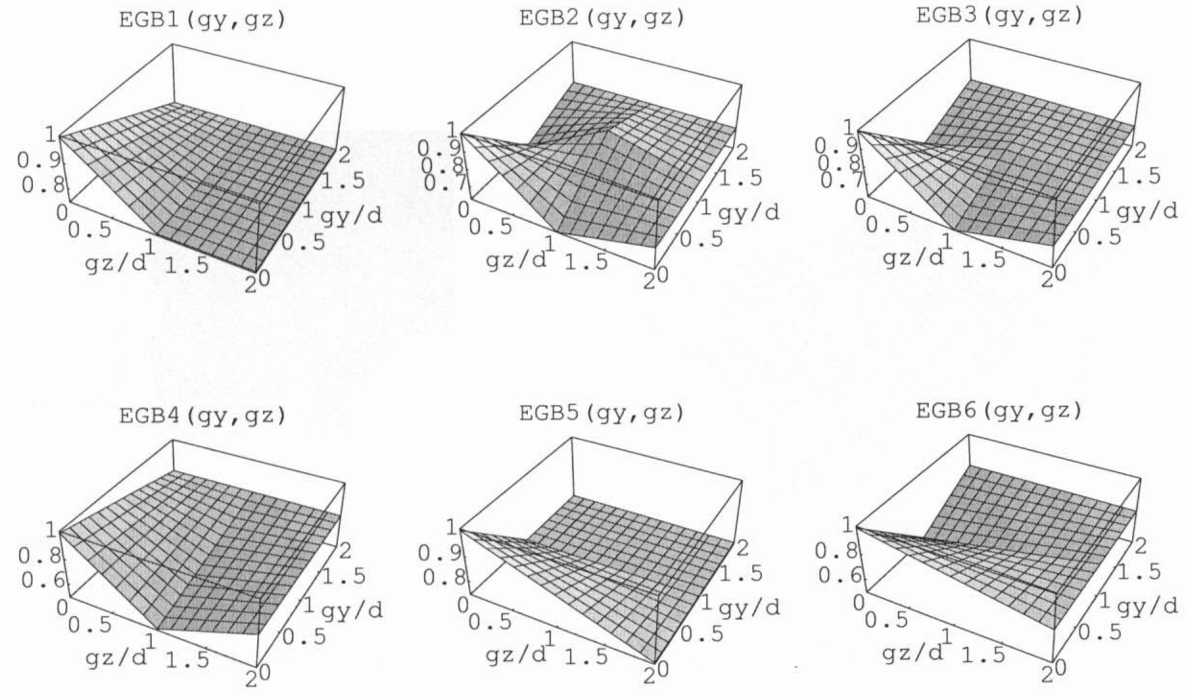

Figure 3.4: Expected value of density of black cells to shift, $EG_{B_i}(g_y, g_z)$

$$EG_{B_5}(g_y, g_z) = EG_{B_1}(g_z, g_y) \tag{3.18}$$

$$EG_{B_6}(g_y, g_z) = EG_{B_4}(g_z, g_y) \tag{3.19}$$

$$EG_{W_i}(g_y, g_z) = \frac{3}{2} - EG_{B_i}(g_y, g_z). \tag{3.20}$$

$EG_{B_i}(g_y, g_z)$ is shown in Fig.3.4. If $(2d \leq g_z)$ or $(2d \leq g_y)$, EG_{B_i} and EG_{W_i} take the same value 3/4. Let the density in a black part and in a white part of a decoded image be $EG_B(g_y, g_z)$ and $EG_W(g_y, g_z)$ respectively. If all kinds of cells are used uniformly, $EG_B(g_y, g_z)$ and $EG_W(g_y, g_z)$ can be expressed as follows:

$$EG_B(g_y, g_z) = \frac{1}{6}\sum_{i=1}^{6} EG_{B_i}(g_y, g_z) \tag{3.21}$$

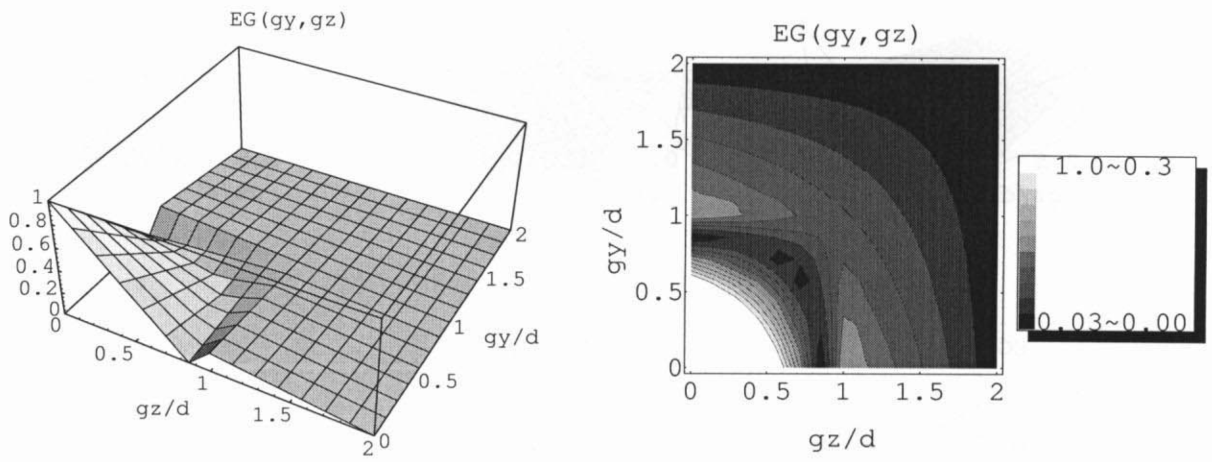

Figure 3.5: Contrast $EG(g_y, g_z)$ of decoded image for shift (g_y, g_z)

$$\begin{aligned} EG_W(g_y, g_z) &= \frac{1}{6}\sum_{i=1}^{6} EG_{W_i}(g_y, g_z) \\ &= \frac{3}{2} - EG_B(g_y, g_z). \end{aligned} \tag{3.22}$$

The visibility of a part of a decoded image depends on the difference of the density between black cells and white cells of which the part of the image consists. Therefore, we use the normalized value $2|EG_B(g_y, g_z) - EG_W(g_y, g_z)|$ as a measure of the visibility $EG(g_y, g_z)$.

$$EG(g_y, g_z) = 2|EG_B(g_y, g_z) - EG_W(g_y, g_z)| \tag{3.23}$$

$EG(g_y, g_z)$ is shown in Fig.3.5. You should pay attention to the region around $(g_y, g_z) = (\pm d, 0)$ or $(g_y, g_z) = (0, \pm d)$. In these regions, the visibility is a little bit higher than the neighborhood and the black and white are reversed.

3.5 Categorization of Space

In this section, we categorize the space where the viewpoint belongs according to the visibility of the decoded image. For simplicity, we consider a $x - y$ plane where $z = 0$, and suppose the visibility of a cell is categorized as follows:

$$\begin{array}{lll} 0 \leq |g_y| < g_0 & : & \text{clearly visible} \\ g_0 \leq |g_y| < c & : & \text{slightly visible} \\ c \leq |g_y| & : & \text{invisible} \end{array}$$

where g_y, g_0 and c denotes the length measured on the transparency 1. c is a length of a side of the cell, and g_0 depends on the sensitivity of a person. When $v_x^{'} \neq 0$, we should consider the difference of the size of the corresponding two cells. However, if $a(v_x^{'}, x_2, x_1) \simeq 1$, or the corresponding two cells do not overlap at all, the effect is very small or not at all. Therefore we can ignore the difference of the size under those conditions.

The visibility of the whole image can be guessed from the visibility of the image on the boundary $(x_1, \pm r_1, 0)$ and the size of the clearly visible region or slightly visible region on the image. The size of the region can be derived from the length from the most visible point to the point where the corresponding two points are shifted as g_y on $x = x_1$ and $z = 0$. Let the length be s_y. s_y is given by the following equation:

$$s_y = \begin{cases} g_y \frac{(x_2 - v_x^{'})x_1}{(x_1 - x_2)v_x^{'}} & (x_2 > v_x^{'} > 0) \\ \infty & v_x^{'} = 0 \\ -g_y \frac{(x_2 - v_x^{'})x_1}{(x_1 - x_2)v_x^{'}} & (v_x^{'} < 0) \end{cases} . \tag{3.24}$$

By substituting g_y for g_0 or c, the size of the slightly visible region and the clearly visible region on the image can be derived. We show s_y versus $v_x^{'}$ in Fig.3.6.

Then the viewpoints where the difference between $y_1^{'}$ and y_1 is g_y can be derived by the following equation:

$$v_y^{'} = \left(\frac{g_y}{x_1 - x_2} + \frac{y_1}{x_1} \right) v_x^{'} - \frac{x_2 g_y}{x_1 - x_2}. \tag{3.25}$$

Therefore, by substituting g_y for $\pm g_0$ or $\pm c$, and y_1 for $\pm r_1$ respectively, the space where the viewpoint belongs can be categorized as follows (see Fig.3.7):

visible space One (or an attacker) can see the whole decoded image clearly.

partly visible space One (or an attacker) cannot see the whole decoded image clearly, but can see a part of it.

space 1 One (or an attacker) may see the region around the center of the decoded image clearly, but cannot see the region around the boundary at all.

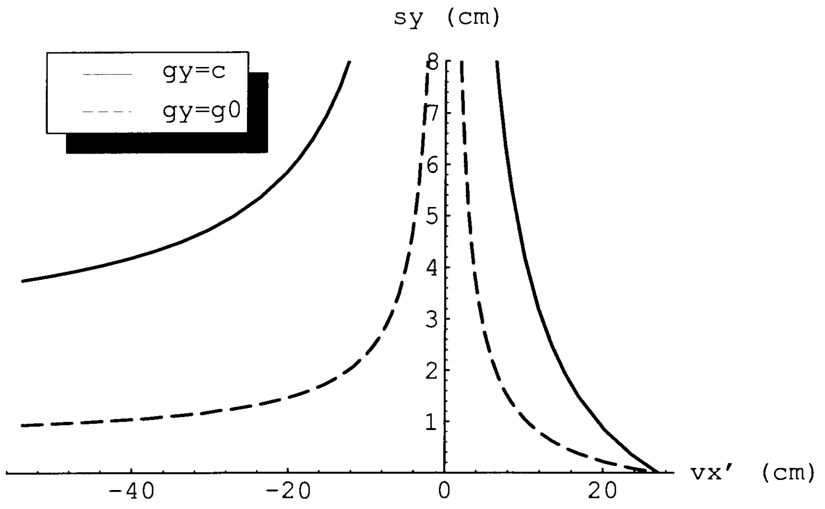

Figure 3.6: Visible region of decoded image for $x_1 = 30cm$,$x_2 = 27cm$, $c = 0.25cm$ and $g_0 = c/4$

space 2 One (or an attacker) may see a region somewhere between the center and one boundary of the decoded image clearly, but cannot see the region around the opposite boundary at all.

space 3 One (or an attacker) may see a region around one boundary of the decoded image clearly, but cannot see the region around the opposite boundary at all.

space 4 One (or an attacker) may see the region around the center of the decoded image clearly, and may also see the region around boundary slightly.

space 5 One (or an attacker) may see a region somewhere between the center and one boundary of the decoded image clearly, and may also see the region around the opposite boundary slightly.

slightly visible space One (or an attacker) cannot see the decoded image clearly, but may see the whole decoded image or a part of the decoded image slightly.

space 6 One (or an attacker) may see a region around one boundary of the decoded image slightly, but cannot see the opposite boundary at all.

space 7 One (or an attacker) may see the whole decoded image slightly, but cannot see it clearly.

invisible space One (or an attacker) cannot see the decoded image at all.

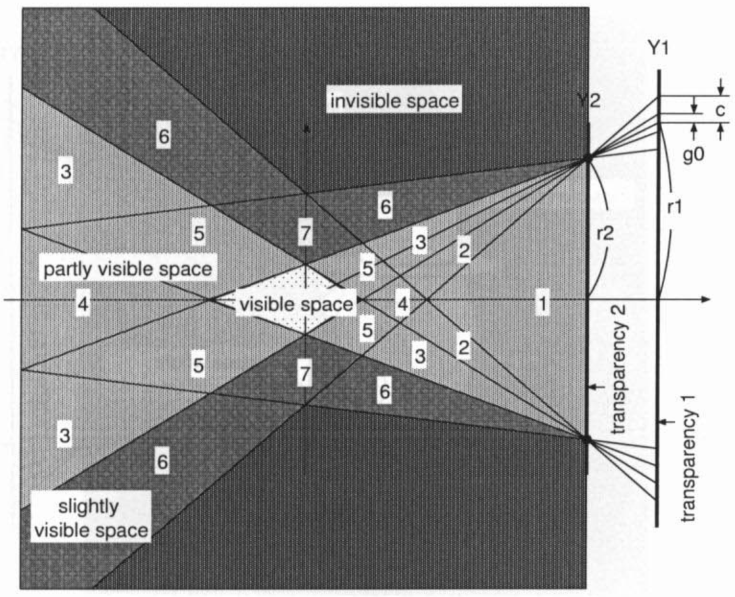

Figure 3.7: Classification of visible space

If the size of the section of the slightly visible space and the visible space is designed to be smaller than the size of one's head or face, attackers cannot see the decoded image at all from everywhere. Because when an attacker see it from behind the person, a part of the decoded image where the attacker can see is hidden behind the person's head, and when from before the person, the attacker can be detected before the image is decoded (see Fig.3.8). The size of the section of the slightly visible space and the visible space can be changed by controlling the length of the sides of the cells c. Let l be the length from the origin $(0, 0, 0)$ to the border between invisible space and slightly visible space on $x = 0$, $z = 0$. The relation between c and l is given by the following equation:

$$c = \frac{(x_1 - x_2)}{x_2} l. \tag{3.26}$$

3.6 Applications to Human Identification

Current human identification schemes using secret codes or passwords are not secure enough against peeping at the input process. To overcome this problem, some schemes have been proposed by several researchers.

One such schemes use the Zero-Knowledge Interactive Proof [16] [76] or One-Time password [29, 30]. These schemes are robust against wire-tapping. However, in these

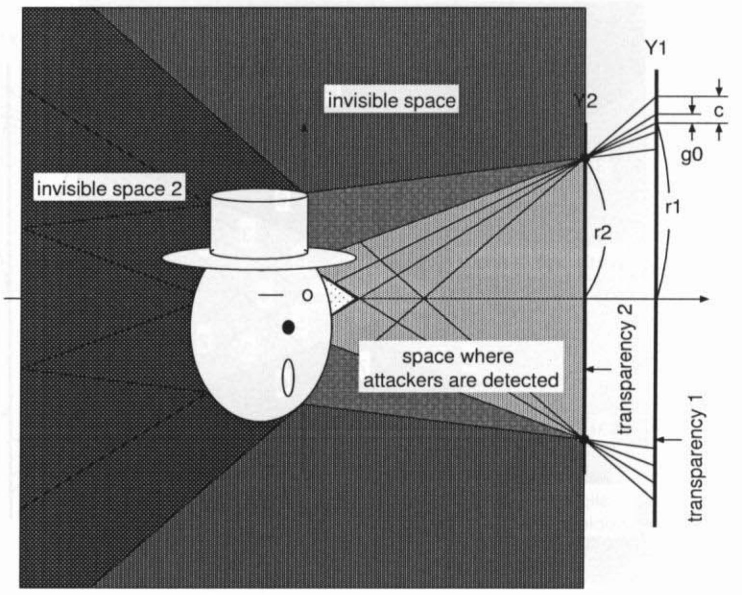

Figure 3.8: Classification of visible space in practice

schemes, verifiers do not verify human provers themselves, although they verify whether the devices are identical. Therefore we call these schemes "indirect human identification schemes" to tell them apart from "direct human identification schemes" in which verifiers can verify the provers themselves. On the other hand, direct human identification schemes which are a little bit robust against peeping have been proposed [53] [32]. These schemes use simple challenge-response protocols so that human provers can make responses by themselves. (We call these schemes "challenge-response type direct human identification schemes" (CRHI).) These schemes are certainly secure against peeping at either of challenges or accepted responses, but not so secure against both of them [37], because attackers can guess provers' secret from several pairs of challenges and their corresponding responses. This is a serious problem of these schemes.

However, if we can prevent attackers peeping at either the challenges or the accepted responses, we can make these schemes extremely secure against peeping. That is the reason why we propose to apply LVSVSS to CRHI. In the proposed scheme, verifiers display challenges by using LVSVSS. The detail is as follows (see Fig.3.9). First, a verifier makes a transparency 2 and give it to a prover secretly. The prover selects a secret as his/her secret s_p and inform it to the verifier secretly. In the identification process, the verifier makes a pattern so that a challenge is decoded when the transparency 2 is stacked on it, and displays it. The prover stacks his/her transparency (transparency 2) on the

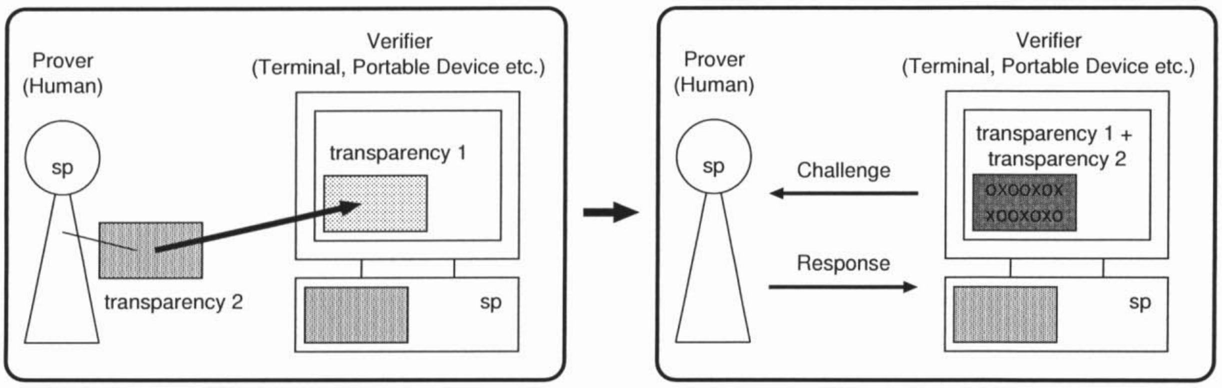

Figure 3.9: Application of LVSVSS to challenge-response human identification

display and see the decoded challenge. Then, he/she makes the response from s_p and the decoded challenge, and returns it. Finally the verifier verify the response. The prover can see the decoded challenges, but attackers cannot peep at them. Therefore, the security against peeping becomes exceedingly higher. Moreover, by using the proposed scheme, it is possible to detect simple fake terminals before they input a response. Because simple fake terminals cannot display proper patterns which proper challenges are decoded by stacking the prover's transparency on, although high-grade fake terminals may be able to do it.

Another advantage of the proposed identification scheme is that it can be actualized easily at a small cost.

3.7 Summary

We proposed a new usage of visual secret sharing schemes, which can limit the visible space of the decoded image. We named it limiting-the-visible-space visual secret sharing schemes (LVSVSS). We investigated the visibility of the decoded image to the viewpoint, and then categorized the space according to the visibility. Finally, we proposed an application of LVSVSS to human identification schemes, which can prevent peeping attacks with small cost. It can also be used to detect simple fake terminals that simply store typed passwords.

Chapter 4

Provably-Secure Public-Key Cryptosystems

In this chapter, we consider preventing attacks over network, using public-key cryptosystems. While the combination of CRHI in Chapter 2 and LVSVSS in Chapter 3 can prevent the peeping in the real world, it does not prevent wiretapping over network since the network adversaries can obtain shared images and decrypt the corresponding challenges displayed to the user using them. In this chapter, we study what properties an ideal public-key cryptosystem should have, and then propose some cryptosystems having the ideal properties using the decoding problem as their underlying hard problem.

4.1 Overview

Since the concept of public-key cryptosystem (PKC) was introduced by Diffie and Hellman [12], many researchers have proposed numerous PKCs based on various problems, such as integer factoring, discrete logarithm, decoding a large linear code, knapsack, inverting polynomial equations, lattice and so on. While some of them are still alive, most of them were broken by cryptographers due to their intensive cryptanalysis. Consequently, almost all of the current secure systems employ a small class of PKCs, such as ElGamal (over a finite field or an elliptic curve) and RSA, which are based on either integer factoring problem (IFP) or discrete logarithm problem (DLP). This situation would cause a serious problem after someone discovers one practical algorithm that breaks both IFP and DLP in polynomial-time. Actually, Shor has already found a (probabilistic) polynomial-time algorithm in [71], even though it requires a quantum computer that is impractical so far. In order to prepare for that unfortunate situation, we need to find another secure scheme relying on neither IFP nor DLP.

While the class, where quantum computation can solve in polynomial time but classical one cannot, has not rigorously been clarified yet, the following facts are known. The problems related with cycle can be solved in polynomial-time over a quantum computer using quantum Fourie transformation. Note that IFP and DLP are categorized in this class. On the other hand, the best current algorithm for the combinatorial problems where one tries to find a right combination out of given pieces, is Grover's one [26] that unfortunately takes $O(\sqrt{2^k})$ steps for 2^k entries of combinations.

The McEliece PKC [55] and the Niederreiter PKC [62] are based on the combinatorial problem, and a few alternatives for IFP- or DLP-based PKCs. Precisely, they are based on the decoding problem of a large linear code with no visible structure, which is conjectured to be an NP-complete problem.[1] While no polynomial-time algorithm has been discovered yet for the decoding problem on both quantum and classical computers, a lot of attacks (some of them work in polynomial-time) are known to the cryptosystems [1, 6, 9, 28, 44, 77, 46, 38].

In this chapter, we summarize these attacks in Section 4.3, and then point out that all the polynomial-time attacks on them require either decryption oracles or partial knowledge on the plaintext. And then without them, no polynomial-time attack is known to invert the code-based PKCs (whose parameters are carefully chosen). Under the assumption that this inverting problem is hard, we convert this problem into enhanced code-based PKCs against adaptive chosen-ciphertext attacks (CCA2) by applying appropriate conversions. In Section 4.4, we introduce the ever known conversions. In Section 4.5, we propose our conversions. The advantage of our conversions is that the data redundancy (defined by the difference between the ciphertext size and the plaintext size) can be smaller than the other conversions.

[1]The complete decoding problem of an arbitrary linear code is proven to be NP-complete in [79].

4.2 Primitive Public-Key Cryptosystems Based on Decoding Problem

4.2.1 Notations

N	:	The number of combinations taking t out of n elements.
$Prep(m)$	:	Preprocessing to a message m, such as data-compression, data-padding and so on. Its inverse is represented as $Prep^{-1}()$.
$Hash(x)$	:	One-way hash function of an arbitrary length binary string x to a fixed length binary string. When the output domain is Z_N, we use $Hash_z(x)$ instead of $Hash(x)$.
$Conv(\bar{z})$	:	Bijective function that converts an integer $\bar{z} \in Z_N$ into the corresponding error vector z. Its inverse is represented as $Conv^{-1}()$. The corresponding algorithm is given in Section 4.2.2.
$Gen(x)$	:	Generator of a cryptographically secure pseudo random sequence of arbitrary length from a fixed length seed x.
$Len(x)$	:	Bit-length of x.
$Hw(x)$	:	The Hamming weight of a binary string x.
$Msb_{x_1}(x_2)$	:	The left x_1 bits of x_2.
$Lsb_{x_1}(x_2)$	:	The right x_1 bits of x_2.
$Const$	:	Predetermined public constant.
$Rand$	:	Random source that generates a truly random (or computationally indistinguishable pseudo random) sequence.
$\mathcal{E}^{McEliece}(x, z)$	:	Encryption of x using the primitive McEliece PKC with an error vector z.
$\mathcal{D}^{McEliece}(x)$	:	Decryption of x using the primitive McEliece PKC.
$\mathcal{E}^{Niederreiter}(x)$	:	Encryption of x using the primitive Niederreiter PKC.
$\mathcal{D}^{Niederreiter}(x)$	:	Decryption of x using the primitive Niederreiter PKC.

4.2.2 Function *Conv*()

Conv() is a function to obtain an n-dimensional vector of weight t corresponding to a given message m, which is represented as an integer in the range of $0 \leq m < \binom{n}{t}$. *Conv*() can be constructed as follows, which is slightly more efficient than the algorithm in [17] [2].

[2]In the algorithm in [17], "$w \leftarrow w - 1$" is missing before the step (d).

Algorithm 1 ($Conv()$)

```
Input: Positive integers $n$, $t$ and $m$ where $t \le n$, $0 \le m < \binom{n}{t}$ and $n \ge 2$
Output: $z = \{z_n, \cdots, z_1\}$
Step 1: temp := $\binom{n-1}{t-1}$
Step 2: For $i := n$ to 2
    If $m <$ temp $z_i := 1$
        • temp := temp $\cdot \frac{t-1}{i-1}$
        • $t := t - 1$
    Otherwise $z_i := 0$
        • $m := m -$ temp
        • temp := temp $\cdot \frac{i-t}{i-1}$
Step 3: $i := 1$
    If $m <$ temp $z_i := 1$
    Otherwise $z_i := 0$
Step 4: Output $z$
```

An intuitive understanding of this algorithm is that there are $\binom{n}{t}$ vectors of length n and weight t, and then they consist of $\binom{n-1}{t}$ vectors being $z_n = 0$ and $\binom{n-1}{t-1}$ ones being $z_n = 1$.

4.2.3 Primitive McEliece Public-Key Cryptosystem

The McEliece PKC is the first code-based PKC proposed by R.J. McEliece in [55]. Its system is described as follows:

Algorithm 2 (Primitive McEliece PKC)

Key generation: Generate the following three matrices G,S and P:

G: $k \times n$ generator matrix of an $(n, k, 2t + 1)$ binary Goppa code $\mathcal{C}$ for which an efficient decoding algorithm $\Phi()$ is known where $\Phi()$ accepts a syndrome and then outputs the corresponding error vector.

S: $k \times k$ random binary non-singular matrix

P: $n \times n$ random permutation matrix.

Then, compute the $k \times n$ matrix $G' = SGP$.

Secret key: (S, P) and $\Phi()$

Public key: (G', t)
such that $G' \times H' = 0$.

Encryption: The ciphertext cpr_M of a message msg_M is given by

$$cpr_M = msg_M G' \oplus z \tag{4.1}$$

where msg_M is a k-dimensional binary vector, cpr_M is a n-dimensional binary vector and z is a random n-dimensional binary vector of weight t.

Decryption: The plaintext msg_M of cpr_M is given as follows. At first, the error in cpr_M is removed using $\Phi()$. Let cpr'_M denote the result. It is given by $cpr'_M := cpr_M \oplus z = cpr_M \oplus \Phi(cpr_M S^{-1})P^{-1})$. Since $cpr'_M = msg_M G'$ is linear, Gaussian elimination outputs msg_M.

4.2.4 Primitive Niederreiter Public-Key Cryptosystem

The Niederreiter PKC (primitive Niederreiter PKC) is a knapsack-type cryptosystem not employing super-increasing numbers but a linear code to make a trapdoor. While Niederreiter showed two examples using a $(104, 24, 31)$ binary concatenated code and a $(30, 12, 19)$ Reed-Solomon code over $GF(31)$, we strongly recommend to employ a Goppa code instead of them due to the reasons summarized in Section 4.3. The system is described as follows:

Algorithm 3 (Primitive Niederreiter PKC)

Key generation: Generate the following three matrices H,S and P:

H: $n \times (n-k)$ parity check matrix of an $(n, k, 2t+1)$ binary Goppa code $\mathcal{C}$ for which an efficient decoding algorithm $\Phi()$ is known where $\Phi()$ accepts a syndrome and then outputs the corresponding error vector.

S: $(n-k) \times (k-k)$ random binary non-singular matrix

P: $n \times n$ random permutation matrix.

Then, compute the $n \times (n-k)$ matrix $H' = PHS$.

Secret key: (S, P) and $\Phi()$

Public key: (H', t)
It is possible to make H' systematic. This reduces the size of H' to $k \times (n-k)$.

Encryption: The ciphertext cpr_N of a message msg_N is given by $cpr_N = zH'$ where msg_N is an integer in the range of $0 \leq msg_N < \binom{n}{t}$ and z is an n-dimensional binary vector of weight t. z is given by $z := Conv(msg_N)$.

Decryption: The plaintext msg_N of cpr_N is given by $msg_N := Conv^{-1}(\Phi(cpr_N S^{-1})P^{-1})$.

4.3 Security of Primitive Code-Based PKCs

In this section, we investigate and categorize currently known attacks to both the McEliece PKC and the Niederreiter PKC.

4.3.1 Attacks on Public-Keys

While no efficient algorithm has been discovered yet for decomposing G' into (S, G, P) [56] (or equivalently H' into (S, H, P)), a structural attack has been discovered in [46]. This attack reveals part of structure of weak G' (or H') that is generated from a "binary" Goppa polynomial. This attack, however, can be avoided by simply avoiding the use of such weak public keys. (This implies G and H should not be a BCH code since it is equivalent to a Goppa code whose Goppa polynomial is $1 \cdot x^{2t}$, i.e. "binary".) Next case we have to consider is that a public matrix G' (or H') happens to be a generator matrix of a code whose decoding algorithm is known. This probability is estimated in [1, 22], and then shown to be negligibly small.

The following attacks try to decrypt given ciphertexts without breaking public-keys. We divide them into two categories, critical attacks and non-critical attacks, according to whether they can be avoided simply by enlarging the parameter size or not. If avoided, we categorize it in the non-critical attacks. Otherwise, in the critical ones. Interestingly, all the critical attacks require either additional information, such as partial knowledge on the target plaintexts, or an decryption oracle which can decrypt arbitrarily given ciphertexts except the challenge ciphertexts. The point is that with neither the additional information nor the decryption ability, no efficient algorithm is known to decrypt an arbitrarily given ciphertext of the code-based cryptosystems.

4.3.2 Non-Critical Attacks on Ciphertexts

Both the GISD (Generalized Information-Set-Decoding) attack [1, 44] and the FLWC (Finding-Low-Weight-Codeword) attack [74, 9] are categorized in this class since they require exponential computation costs to invert a given cipher to the security parameter size.

Generalized Information-Set-Decoding

While we describe this attack using the McEliece PKC, it is easily convertible on the Niederreiter PKC.

Let G'_k denote k independent columns picked out of G', and then let cpr_{Mk} and z_k denote the corresponding k coordinates of cpr_M and z, respectively. They have the following relationship

$$cpr_{Mk} = msg \cdot G'_k \oplus z_k. \tag{4.2}$$

If $z_k = 0$ and G'_k is non-singular, msg can be recovered [1] by

$$msg_M = (cpr_{Mk} \oplus z_k) G'^{-1}_k. \tag{4.3}$$

Even if $z_k \neq 0$, msg_M can be obtained by guessing z_k among small Hamming weights [44], i.e. $Hw(z_k) \leq j$ for small j. The correctness of the recovered plaintext msg_M is verifiable by checking whether the Hamming weight of

$$cpr_M \oplus msg_M \cdot G' \;=\; cpr_M \oplus cpr_{Mk} G'^{-1}_k \cdot G' \oplus z_k G'^{-1}_k \cdot G' \tag{4.4}$$

is t or not.

The corresponding algorithm is summarized as follows:

Algorithm 4 (GISD)

Input: a ciphertext cpr_M, a public key (G', t) and an attack parameter $j \in Z$.
Output: a plaintext msg_M.

1. Choose k independent columns out of G', and then calculate $\hat{G'}_k := G'^{-1}_k G'$. Let I denote the set of the indexes of the k chosen columns, and then J denote the set of the remaining columns.

2. Do the following until msg_M is found:

   (a) Calculate $\hat{z} := cpr_M \oplus cpr_{Mk}\hat{G'}_k$. If $Hw(\hat{z}) = t$, output $msg_M := cpr_{Mk}G'^{-1}_k$.

   (b) For i_1 from 1 to j do the following:

      i. For i_2 from 1 to $\binom{n}{i_1}$ do the following:

         A. Choose a new z'_k, such that $Hw(z'_k) = i_1$.

         B. If $Hw(\hat{z} \oplus z'_k\hat{G'}_k) = t$, output $msg_M := (cpr_{Mk} \oplus z'_k)G'^{-1}_k$.

   (c) Replace one coordinate in I with a coordinate in J, and then renew the $\hat{G'}_k := G'^{-1}_k G'$ using Gaussian elimination.

We estimate the binary work factor of the above GISD attack as follows. In Step 1, $G'^{-1}_k G'$ is the $k \times n$ matrix where the chosen k columns make the identity matrix. It can be obtained by the Gaussian elimination with the work factor of

$$\sum_{i=1}^{k} \frac{(k-1)(n-i+1)}{4} = \frac{k(k-1)(2n+1-k)}{8} \tag{4.5}$$

bit operations. When one checks the Hamming weight in Step 2.1 and Step B, he/she does not need to calculate the whole n coordinates of $cpr_M \oplus cpr_{Mk}\hat{G'}_k$ in Step 2.1 and $\hat{z} \oplus z'_k\hat{G'}_k$ in Step B, respectively, since he/she can know whether their weight exceeds t or not with around $2t$ coordinates in J provided that wrong cases have the average weight of $n/2$. Thus the binary work factor for calculating the $2t$ coordinates of $cpr_M \oplus cpr_{Mk}\hat{G'}_k$ in Step 2.1 is $t \cdot k/2$, and that of $\hat{z} \oplus z'_k\hat{G'}_k$ in Step B is $t \cdot i_1$. Accordingly, the work factor for Step 2.2 is

$$V_j = \sum_{i_1=1}^{j} t \cdot i_1 \cdot \binom{k}{i_1}. \tag{4.6}$$

In Step 2.3, one needs to update $\hat{G'}_k = G'^{-1}_k G'$ whose binary work factor is

$$\frac{(k-1)(n-k)}{4}. \tag{4.7}$$

Since Step 2 is repeated around T_j times where:

$$T_j = \frac{\binom{n}{k}}{\sum_{i=0}^{j}\binom{t}{i}\binom{n-t}{k-i}}, \tag{4.8}$$

the total work factor is given by

$$W_j \approx \left\{\frac{(k-1)(n-k)}{4} + \frac{t\cdot k}{2} + V_j\right\}\cdot T_j \tag{4.9}$$

When n is given, designers of the cryptosystem can optimize both k and t to make (4.9) higher, and then attackers can optimize the attack parameter j to make it lower. For $n = 2^{10}$, $\min_j(\max_{k,t}(W_j)) \approx 2^{67}$, which can be achieved when $j = 1$, $t = 38$ to 40 and $k = n - m\cdot t = 644$ to 624, respectively. For $n = 2^{11}$, $\min_j(\max_{k,t}(W_j)) \approx 2^{113}$, which can be achieved when $j = 1$, $t = 63$ to 78 and $k = n - m\cdot t = 1355$ to 1190, respectively.

Finding-Low-Weight-Codeword Attack

This attack uses an algorithm which accepts both an arbitrary generator matrix and a positive integer w, and then finds out a codeword of weight w [74, 9]. Since the codeword of weight t of the following $(k+1)\times n$ generator matrix

$$\begin{bmatrix} G' \\ cpr_M \end{bmatrix} \tag{4.10}$$

is the error vector z where $c = msg_M \cdot G' \oplus z$, this algorithm can be used to recover msg_M from given cpr_M and G'. Note that this algorithm can also be used to invert a given ciphertext of the Niederreiter PKC.

This algorithm for the McEliece PKC is summarized as follows:

Algorithm 5 (FLWC)

Input: a ciphertext cpr_M, a public key (G', t) and attack parameters $(p, \rho) \in Z \times Z$.
Output: a plaintext msg_M.

1. Choose $k+1$ independent columns from (4.10) and then apply Gaussian elimination to obtain a $(k+1)\times n$ matrix where chosen $k+1$ columns make the identity matrix. Let I denote a set of the indexes of the $k+1$ chosen coordinates, and J denote those of the remaining coordinates.

2. Do the following until a code word z of weight t is found:

(a) Split I into two subsets I_1 and I_2 at random where $|I_1| = \lfloor (k+1)/2 \rfloor$ and $|I_2| = \lceil (k+1)/2 \rceil$. The rows of the $(k+1) \times (n-k-1)$ matrix M corresponding to J are also split into two parts, a $(\lfloor (k+1)/2 \rfloor) \times (n-k-1)$ matrix M_1 and a $(\lceil (k+1)/2 \rceil) \times (n-k-1)$ matrix M_2 according to I_1 and I_2, respectively, i.e. if I_1 includes i-th coordinate, the i-th row of M is included in M_1.

(b) Select a ρ-element subset J_ρ of J at random.

(c) For i from 1 to $\binom{|I_1|}{p}$ do the following:

   i. Select a new set of p rows of the matrix M_1. Let $\mathcal{P}_{1,i}$ denote the set.

   ii. Sum up the chosen p rows of M_1 in Z_2. Let $\Lambda_{1,i|J_\rho}$ denote the chosen ρ coordinates of the result.

   iii. Store both $\mathcal{P}_{1,i}$ and $\Lambda_{1,i|J_\rho}$ in a hash table with 2^ρ entries using $\Lambda_{1,i|J_\rho}$ as an index.

(d) For j from 1 to $\binom{|I_2|}{p}$ do the following:

   i. Select a new set of p rows of the matrix M_2. Let $\mathcal{P}_{2,j}$ denote the set.

   ii. Sum up the chosen p rows of M_2 in F_2. Let $\Lambda_{2,j|J_\rho}$ denote the chosen ρ coordinates of the result.

   iii. Store both $\mathcal{P}_{2,j}$ and $\Lambda_{2,j|J_\rho}$ in a hash table with 2^ρ entries using $\Lambda_{2,j|J_\rho}$ as an index.

(e) Using the hash table, find all pairs of sets $(\mathcal{P}_{1,i}, \mathcal{P}_{2,j})$ such that $\Lambda_{1,i|J_\rho} = \Lambda_{2,j|J_\rho}$ and check whether $Hw(\Lambda_{1,i|J} \oplus \Lambda_{2,j|J}) = t - 2p$ where $\Lambda_{1,i|J}$ and $\Lambda_{2,j|J}$ denote the sums of the p rows of M_1 and M_2 corresponding to $\mathcal{P}_{1,i}$ and $\mathcal{P}_{2,j}$, respectively. If found, output the code word.

(f) Replace one coordinate in I with a coordinate in J, and then make the chosen $k+1$ columns be the identity matrix using Gaussian elimination.

3. Apply the information-set decoding to $c \oplus z$, and then recover the corresponding message msg_M.

We estimate the binary work factor of the above FLWC attack as follows. Under the assumption that each iteration is independent, one needs to repeat Step 2 around $T_{p,\rho}$ times where

$$T_{p,\rho} = \frac{\binom{k-2p}{\frac{k}{2}-p}\binom{2p}{p}\binom{-k+n+2p-t}{\rho}}{\binom{k}{\frac{k}{2}}\binom{-k+n}{\rho}} \cdot \frac{\binom{n-t}{k-2p}\binom{t}{2p}}{\binom{n}{k}}. \tag{4.11}$$

In Step 2.1 to 2.4, one needs to compute both $\Lambda_{1,i|J_\rho}$ and $\Lambda_{2,j|J_\rho}$ for about $\binom{(k+1)/2}{p}$ combinations, respectively, whose binary work factor is around

$$\Omega_1(p,\rho) = p \cdot \rho \cdot \binom{(k+1)/2}{p}. \tag{4.12}$$

In Step 2.5, around $\binom{(k+1)/2}{p}^2/2^\rho$ pairs of $(\mathcal{P}_{1,i}, \mathcal{P}_{2,j})$ satisfy $\Lambda_{1,i|J_\rho} \oplus \Lambda_{2,j|J_\rho} = 0$, and for each pair one needs to check the weight of $\Lambda_{1,i|J} \oplus \Lambda_{2,j|J}$. In the same way as Algorithm 4, one can know that $Hw(\Lambda_{1,i|J} \oplus \Lambda_{2,j|J}) \neq t - 2p$ by calculating the weight of around $2(t-2p)$ coordinates in J. Thus the binary work factor for Step 2.5 is around

$$\Omega_2(p,\rho) = 2(t-2p) \cdot p \cdot \frac{\binom{(k+1)/2}{p}^2}{2^\rho}. \tag{4.13}$$

The binary work factor for updating the generator matrix in Step 2.6 is

$$\Omega_3(p,\rho) = \frac{k(n-k-1)}{4}. \tag{4.14}$$

Thus the total binary work factor is given by

$$W_{p,\rho} \approx (\Omega_1(p,\rho) + \Omega_2(p,\rho) + \Omega_3(p,\rho)) \cdot T_{p,\rho}. \tag{4.15}$$

For $n = 2^{10}$, $\min_{p,\rho}(\max_{k,t}(W_{p,\rho})) \approx 2^{62}$, which can be achieved when $(p,\rho) = (2,19)$, $t = 36$ to 43 and $k = n - m \cdot t = 664$ to 594, respectively. For $n = 2^{11}$, $\min_{p,\rho}(\max_{k,t}(W_{p,\rho})) \approx 2^{106}$, which can be achieved when $(p,\rho) = (2,22)$, $t = 63$ to 79 and $k = n - m \cdot t = 1355$ to 1179, respectively.

4.3.3 Critical Attacks on Ciphertexts

The following attacks are categorized in this class since they cannot be avoided by enlarging the security parameter size.

Known-Partial-Plaintext Attack

The partial knowledge on the target plaintext drastically reduces the computational cost of the attacks to the McEliece PKC [9, 38].

For example, let m_l and m_r denote the left k_l bits and the remaining k_r bits in the target plaintext msg_M, i.e. $k = k_l + k_r$ and $msg_M = (m_l || m_r)$. Suppose that an adversary knows m_r. Then the difficulty of recovering unknown plaintext m_l of the McEliece PKC with parameters (n, k) is equivalent to that of recovering the full plaintext of the McEliece PKC with parameters (n, k_l) since

$$\begin{aligned} cpr_M &= msg_M G' \oplus z \\ cpr_M &= m_l G'_l \oplus m_r G'_r \oplus z \\ cpr_M \oplus m_r G'_r &= m_l G'_l \oplus z \\ cpr'_M &= m_l G'_l \oplus z, \end{aligned} \tag{4.16}$$

where G'_l and G'_r are the upper k_l rows and the remaining lower k_r rows in G', respectively.

If k_l is fixed to a small value, the computational cost of recovering the unknown k_l bits from cpr_M, m_r and G' is a polynomial of n.

Related-Message Attack

This attack uses the knowledge on the relationship between the target plaintexts of the McEliece PKC [6].

Suppose two messages msg_{M1} and msg_{M2} are encrypted to cpr_{M1} and cpr_{M2}, respectively, where $cpr_{M1} = msg_{M1}G' \oplus z_1$, $cpr_{M2} = msg_{M2}G' \oplus z_2$, and $z_1 \neq z_2$. If an adversary knows their linear relation between the plaintexts, e.g. $\delta msg_M = msg_{M1} \oplus msg_{M2}$. Then the adversary can efficiently apply the GISD attack to either cpr_{M1} or cpr_{M2} by choosing k coordinates whose values are 0 in $(\delta msg_M G' \oplus cpr_{M1} \oplus cpr_{M2})$. Since $z_1 \oplus z_2 = \delta msg_M G' \oplus cpr_{M1} \oplus cpr_{M2}$ and the Hamming weight t of the error vector z is far smaller than $n/2$. Therefore a coordinate being 0 in $(\delta msg_M G' \oplus cpr_{M1} \oplus cpr_{M2})$ should also be 0 in both z_1 and z_2 with the high probability of $1/(1 + (t/(n-t))^2)$.

When the same message is encrypted twice (or more) using different error vectors z_1 and z_2, the value $z_1 \oplus z_2$ is simply given by $cpr_{M1} \oplus cpr_{M2}$. This case is referred to as the message-resend attack [6].

Reaction Attack

This attack might be categorized as a chosen-ciphertext attack (CCA), but uses a weaker assumption [28] than the CCA: the adversary observes only the reaction of the receiver who has the private-key, but does not need to receive its decrypted plaintext. (Similar attack is independently proposed in [77], in which an adversary receives the corresponding plaintexts. Therefore this attack is categorized in CCA.)

The idea of this attack is the following. The adversary flips one or a few bits of the target ciphertext cpr_M. Let cpr'_M denote the flipped ciphertext. The adversary transmits cpr'_M to the proper receiver and observes his/her reaction. The receiver's reactions can be divided into the following two:

Reaction A: Return a repeat request to the adversary due to uncorrectable error or due to the meaningless plaintext.

Reaction B: Return an acknowledgment or do nothing since the proper plaintext msg_M is decrypted.

If the total weight of the error vector does not exceed t after the flipping, the reaction B is observed. Otherwise the reaction is A. Therefore by repeating the above observations polynomial times of n, the adversary can determine the error vector. Once the error vector is determined, the corresponding plaintext is easily decrypted using the GISD attack.

This attack is extensible to the primitive Niederreiter PKC as follows. The adversary adds the i-th row of H' to the target ciphertext cpr_N

$$cpr'_N \quad = \quad cpr_N \oplus H'[i]. \tag{4.17}$$

This flips one bit of the error vector z corresponding to cpr_N. Note that $cpr'_N = zH \oplus H'[i] = z'H$. Then the adversary transmits cpr'_N to the proper receiver who has the private-key for cpr_N, and observes the receiver's reaction. If the total weight of the flipped error vector z' exceeds t, it must return a repeat-request to the adversary This repeat-request reveals that the i-th coordinate of z is 0. Otherwise, it is 1. Thus by repeating the above observation at most n times, the adversary can determine the entire z. Once z is determined, the corresponding plaintext is given by $msg_N = Conv^{-1}(z)$.

Malleability Attack

This attack allows an adversary to generate a new ciphertext cpr' from a given ciphertext cpr where a certain relationship $R(msg', msg)$ holds between the corresponding plaintexts of cpr' and cpr [38, 77].

This attack on the primitive McEliece PKC is described as follows. Let $G'[i]$ denote the i-th row of the public matrix G' and $I = \{i_1, i_2, \cdots\}$ denote a set of coordinates i_j whose value is 1 in $\delta msg_M = msg_M \oplus msg'_M$. Then the ciphertext cpr'_M is given by

$$cpr'_M \quad = \quad cpr_M \bigoplus_{i \in I} G'[i] = (msg_M \oplus \delta msg_M)G' \oplus z = msg'_M G' \oplus z. \tag{4.18}$$

This attack tells us that the McEliece PKC does not satisfy non-malleability[14] even against chosen-plaintext attacks. And then under chosen-ciphertext scenario where an adversary can ask decryption oracles to decrypt a polynomial number of ciphertexts (excluding the target ciphertext cpr_M), the adversary can decrypt any given ciphertext cpr_M using this malleability attack as follows. First the adversary asks the oracle to decrypt

cpr'_M, then the oracle returns $msg'_M = msg_M \oplus \delta msg_M$. Thus he/she can recover the target plaintext of cpr_M with $msg_M = msg'_M \oplus \delta msg_M$.

This attack is also applicable to the primitive Niederreiter PKC. It is given as follows. Let $H'[i]$ denote the i-th row of the public matrix H' and $I = \{i_1, i_2, \cdots\}$ denote a set of coordinates i_j whose value is 1 in $\delta z = Conv(msg_N) \oplus Conv(msg'_N)$. The adversary adds all the rows in I of H' to the target ciphertext cpr_N

$$cpr'_N = cpr_N \bigoplus_{i \in I} H'[i] = (Conv(msg_N) \oplus \delta z)H' = Conv(msg'_N)H'. \tag{4.19}$$

This flips $|I|$ bits of the error vector $z = Conv(msg_N)$.

If the total weight of the corresponding error vector z' of cpr' does not exceed t, cpr' is a valid ciphertext of msg'_N. Thus with the help of the decryption oracle, the adversary can obtain msg'_N. The target plaintext msg_N of cpr_N is given by

$$msg_N = Conv^{-1}(\delta z \oplus Conv(msg'_N)). \tag{4.20}$$

4.3.4 OW-CPA of Code-Based PKCs

OW-CPA (One-Wayness against Chosen-Plaintext Attacks) is said to be satisfied if no polynomial time algorithm is known to invert an arbitrarily given ciphertext.

As we have seen in the previous sections, all the critical attacks require either additional information, such as partial knowledge on the target plaintexts, or decryption oracles. And then without the additional information and the ability, no efficient algorithm is known to decrypt an arbitrarily given ciphertext of the primitive McEliece and Niederreiter PKCs. Thus we can assume that they satisfy OW-CPA.

4.4 Conversion Schemes

Conversion techniques can be used to convert relatively weak cryptographic primitives into strong cryptosystems that do not reveal any partial information on the plaintexts even against chosen ciphertext attacks. In this section, we introduce and categorize them according to the applicable function types.

4.4.1 Classification of Function Types

Primitive PKCs can be divided into either deterministic or probabilistic according to whether or not they use random inputs, i.e.

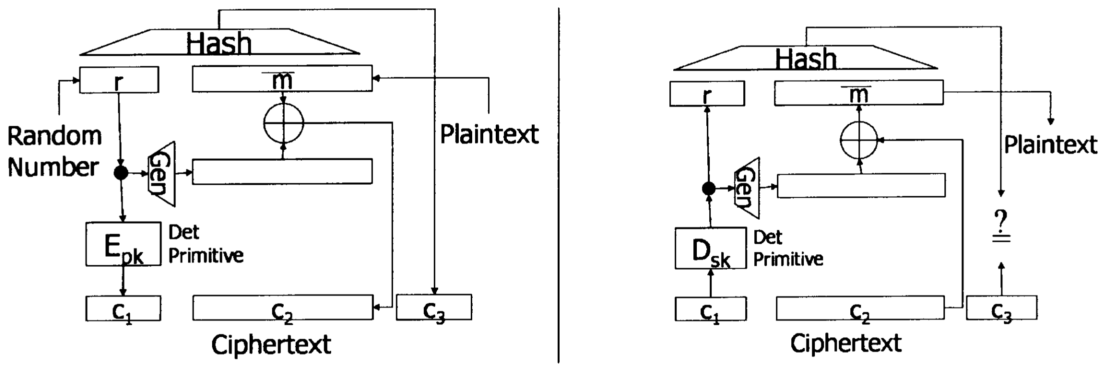

Figure 4.1: Bellare-Rogaway conversion: Encryption (left) and decryption (right)

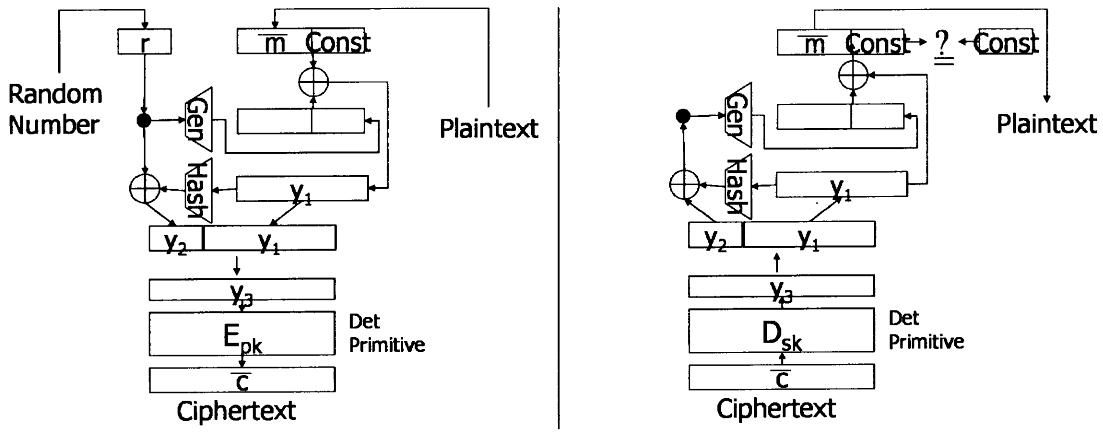

Figure 4.2: OAEP conversion: Encryption (left) and decryption (right)

Definition 1 (Deterministic Primitive PKC) *A PKC is said to be deterministic if its ciphertext is deterministic to its plaintext.*

Definition 2 (Probabilistic Primitive PKC) *A PKC is said to be probabilistic if its ciphertext is deterministic to both a random number and its plaintext.*

We further divide the probabilistic primitives into the following two classes according to whether or not the randomness can be recovered.

Definition 3 (Fully Trapdoor Probabilistic Primitive PKC) *A probabilistic primitive PKC is said to be fully trapdoor if the whole randomness can be recovered.*

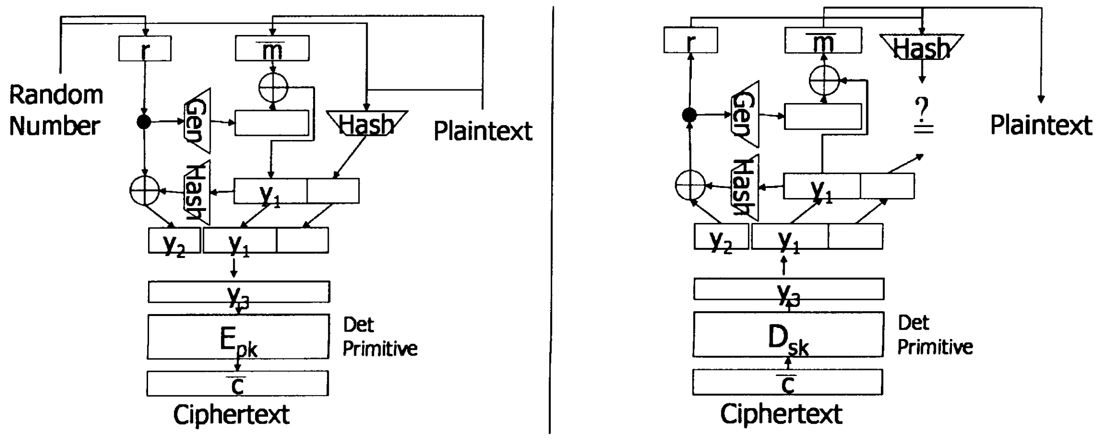

Figure 4.3: OAEP+ conversion: Encryption (left) and decryption (right)

Definition 4 (Partially Trapdoor Probabilistic Primitive PKC) *A probabilistic primitive PKC is said to be partially trapdoor if it is hard to recover the whole randomness even if its decryption key is used.*

For example, RSA [68], Rabin [67] and Niederreiter [62] PKCs are categorized in the deterministic primitives. McEliece [55], NTRU [31] and S-Paillier [10] PKCs are categorized in the fully trapdoor primitives. ElGamal [15], Elliptic Curve Cryptosystem (ECC) [58, 41], XTR [45], Okamoto-Uchiyama [64] and Paillier [65] PKCs are categorized in the partially trapdoor primitives.

The applicability of the conversions depends on both the function type and the security assumption of the primitives.

4.4.2 Conversions for Deterministic Primitives

In this section, we introduce specific conversions being applicable only to deterministic primitives under certain conditions.

The point of the deterministic functions is that the modification of the output can be detected by seeing the corresponding input. On the other hand, if the primitive function is probabilistic, receivers cannot necessarily detect the modification of the output by seeing only the corresponding message (of the primitive) since adversaries may be able to modify the output without modifying the corresponding message. This may accept the malleability attack.

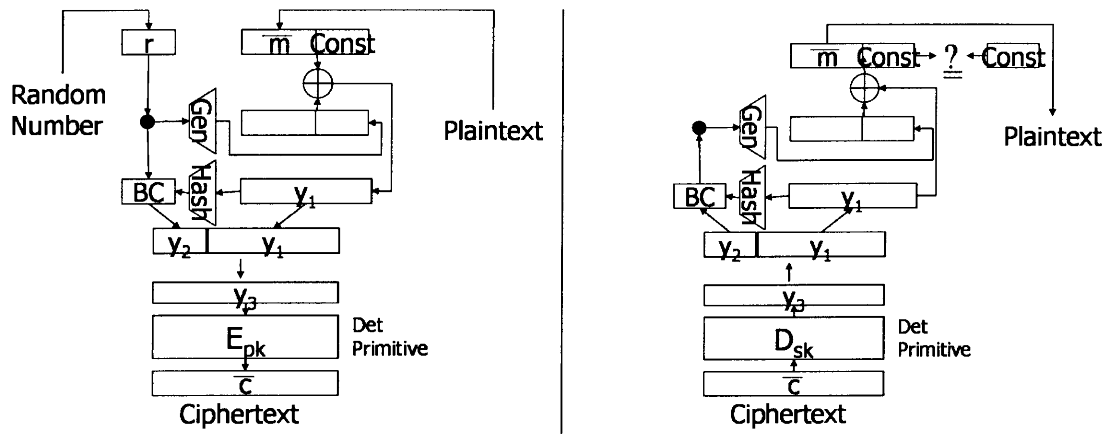

Figure 4.4: OAEP^{++} conversion: Encryption (left) and decryption (right)

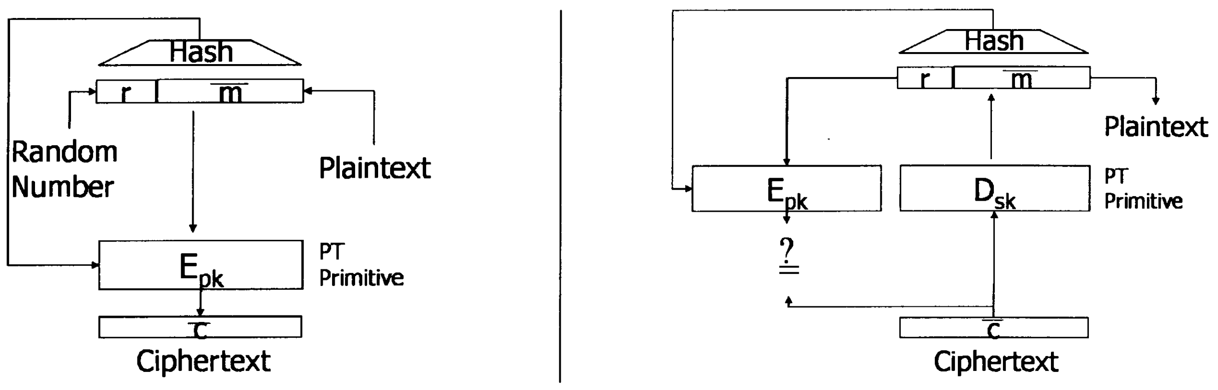

Figure 4.5: Fujisaki-Okamoto simple conversion: Encryption (left) and decryption (right)

Bellare-Rogaway Conversion

This is the first conversion proven to be secure in the random oracle model [3]. It can generate, in the Random Oracle Model (ROM), a PKC satisfying IND-CCA2 under the assumption that the underlying primitive function is deterministic and satisfies OW-CPA. This conversion is illustrated with Fig. 4.1.

OAEP

OAEP (Optimal Asymmetric Encryption Padding) is a conversion proposed by Bellare and Rogaway in [4]. Originally it was believed to be able to generate in ROM an IND-CCA2 cipher under the assumption that the underlying primitive function is determin-

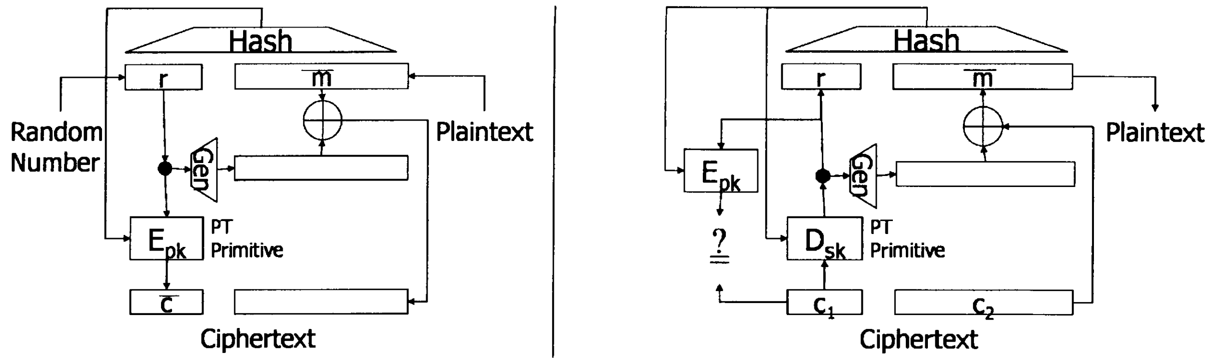

Figure 4.6: Fujisaki-Okamoto conversion: Encryption (left) and decryption (right)

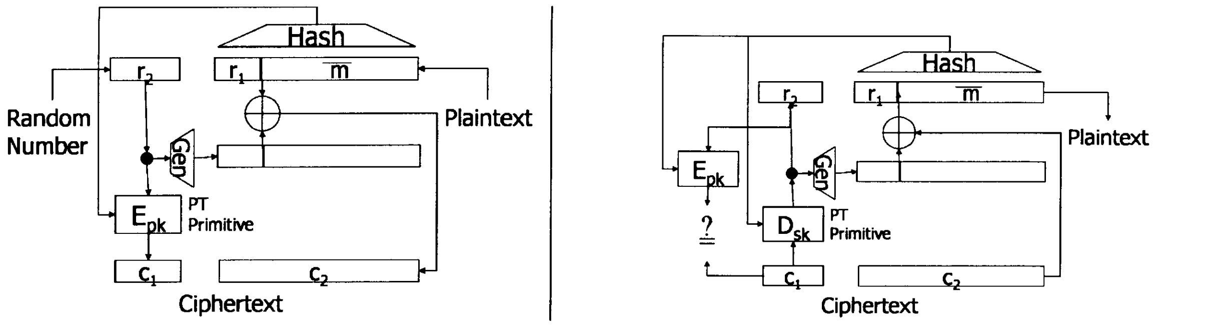

Figure 4.7: Pointcheval conversion: Encryption (left) and decryption (right)

istic and OW-CPA. Unfortunately, that was myth. Shoup showed in [72] OAEP cannot generate it from a certain class of the random permutations, named XOR-malleable permutations, even if they satisfy OW-CPA.

Currently, it has been proven, in the Random Oracle Model (ROM), that PDOW-CPA (Partial-Domain One-Wayness against CPA) is a sufficient condition for OAEP to convert a deterministic function to an IND-CCA2 PKC [21]. PDOW is satisfied if there exists no polynomial-time algorithm to recover a certain part of the input of the underlying primitive function from an arbitrarily given output of it. OAEP conversion is illustrated with Fig. 4.2. In the figure, the certain part corresponding to the partial domain is y_1.

OAEP+

This is a modified version of the OAEP proposed by Shoup in [72] to fix the weakness in OAEP. It can generate, in the random oracle model, a PKC satisfying IND-CCA2 in ROM under the assumption that the underlying primitive function is deterministic and

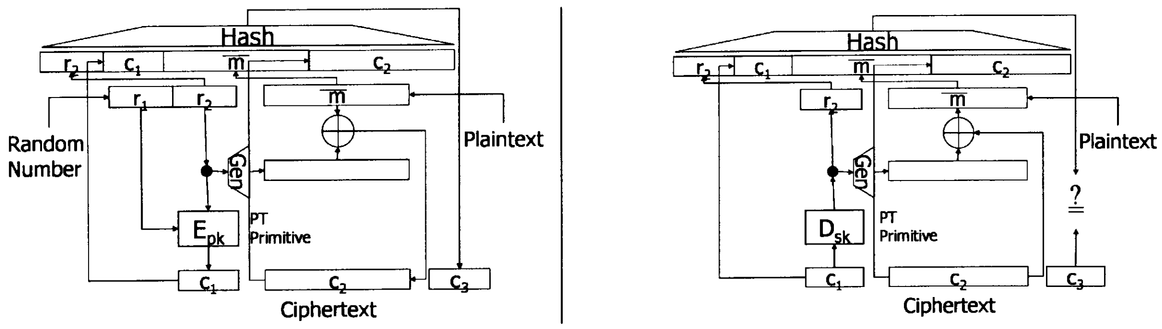

Figure 4.8: Conversion REACT: Encryption (left) and decryption (right)

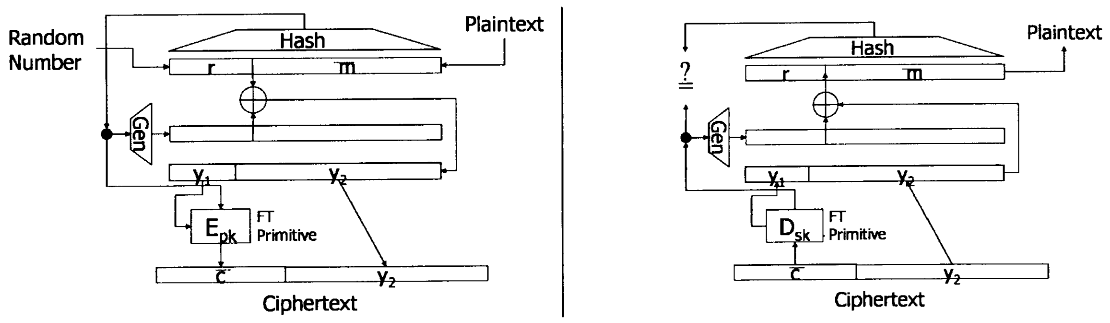

Figure 4.9: Conversion α: Encryption (left) and decryption (right)

OW-CPA. It is illustrated with Fig. 4.3.

OAEP^{++}

This is also a modified version of the OAEP proposed by Jonsson in [33], which employs a block cipher of non-malleable against CPA. It can generate, in the random oracle model, a PKC satisfying IND-CCA2 in ROM under the assumption that the underlying primitive function is deterministic and OW-CPA. It is illustrated with Fig. 4.4.

4.4.3 Conversions for Partially Trapdoor Primitives

In this section, we introduce conversions for partially trapdoor primitives. They are applicable not only to partially trapdoor primitives, but also to fully trapdoor primitives and deterministic ones under certain assumptions.

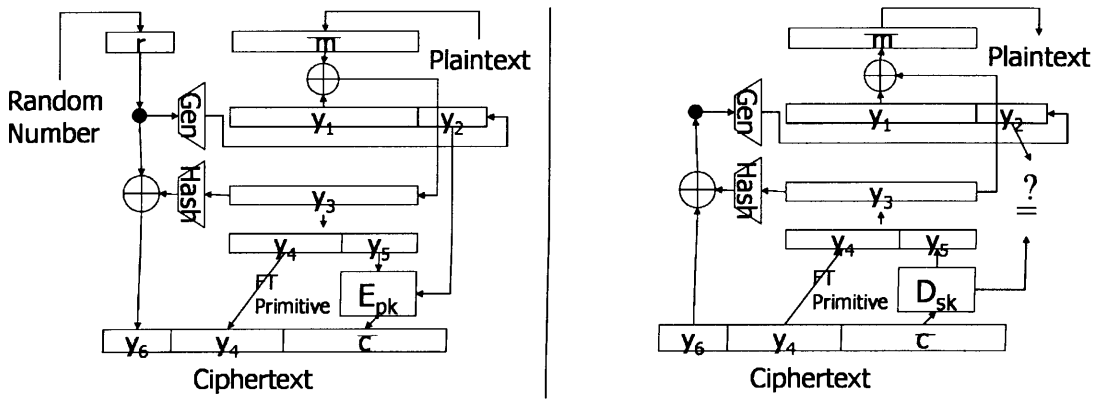

Figure 4.10: Conversion β: Encryption (left) and decryption (right)

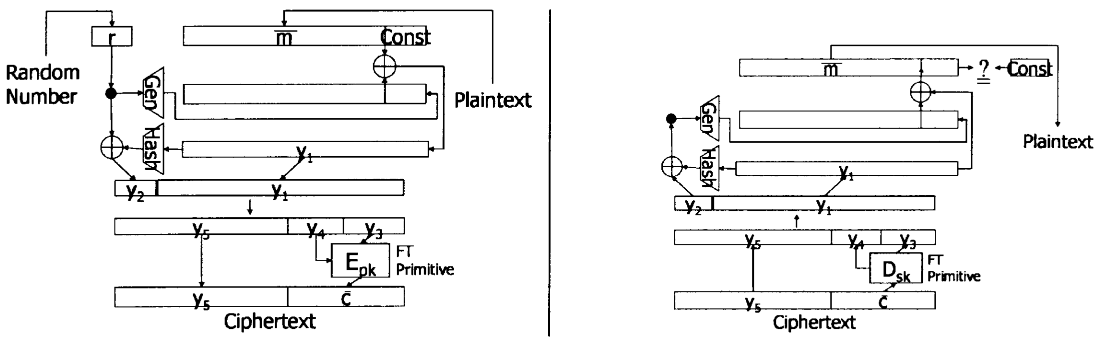

Figure 4.11: Conversion γ: Encryption (left) and decryption (right)

Fujisaki-Okamoto Simple Conversion

In [19], Fujisaki and Okamoto proposed a simple conversion that can generate a PKC satisfying IND-CCA2 in ROM under the assumption that the underlying primitive function is IND-CPA. Note that IND-CPA is a stronger assumption than OW-CPA. It is illustrated with Fig. 4.5. When it is applied to a deterministic function, the hashed value $Hash(r||\bar{m})$ is concatenated with the ciphertext of the primitive deterministic PKC.

Fujisaki-Okamoto Conversion

In [20], Fujisaki and Okamoto proposed another conversion that can generate a PKC satisfying IND-CCA2 in ROM under the assumption that the underlying primitive function is OW-CPA. It is illustrated with Fig.4.6. When it is applied to a deterministic function, the hashed value $Hash(r||\bar{m})$ is concatenated with the ciphertext of the primitive determin-

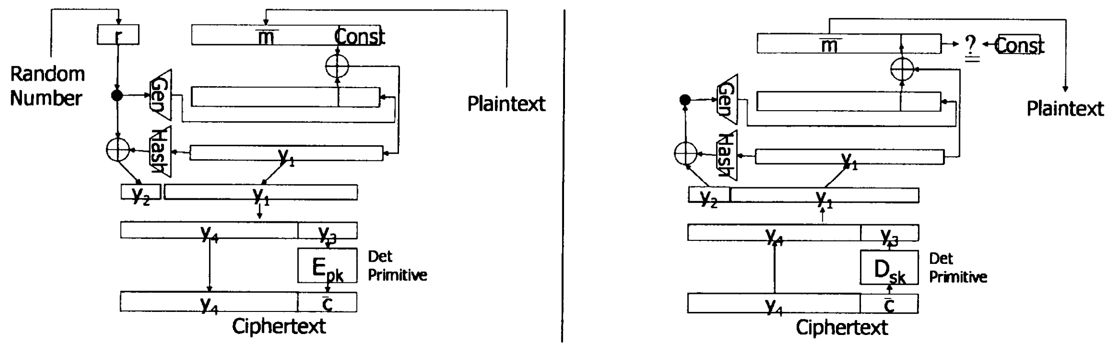

Figure 4.12: OAEP++ conversion: Encryption (left) and decryption (right)

istic PKC, which is interestingly the same as the previously proposed Bellare-Rogaway's conversion[3].

Pointcheval's Conversion

In [66], Pointcheval proposed a generic conversion that can generate a PKC satisfying IND-CCA2 in ROM under the assumption that the underlying primitive function is OW-CPA. It is illustrated with Fig.4.7. It is also applicable to deterministic primitives by concatenating the hashed value $Hash(r_1||\bar{m})$ with the ciphertext of the primitive deterministic PKC.

REACT

In [63], Okamoto and Pointcheval proposed the generic conversion REACT (Rapid Enhanced-security Asymmetric Cryptosystem Transform). It can generate a PKC satisfying IND-CCA2 in ROM under the assumption that the underlying primitive function is OW-PCA. Note that PCA is not a typo of CPA. It stands for Plaintext-Checking Attacks where an adversary has access to a plaintext-checking oracle that takes a pair of a plaintext m' and a ciphertext c', and then outputs whether c' is a ciphertext of m' or not. REACT is illustrated with Fig.4.8.

4.5 Our Proposal

4.5.1 New Conversions

Conversion α

This is a conversion for the fully trapdoor primitives proposed by us in [11]. It can generate IND-CCA2 PKCs in ROM under the assumption of OW-PCA of the primitives. Fig. 4.9 illustrates the structure of it. It can also be applied to the deterministic primitives under the assumption of OW-CPA by putting out the randomness for the probabilistic primitives as a part of the ciphertexts. Unfortunately, they cannot be applied to the partially trapdoor primitives since the randomness cannot be recovered in them. The advantage of α is that the ciphertext size becomes smaller when it is applied to the fully trapdoor primitives than applying conversions for the partially trapdoor primitives to the fully trapdoor ones. We will show the comparative results in Section 4.5.3.

Conversion β

This is a conversion for the partially trapdoor primitives proposed by us in [11]. It can generate IND-CCA2 PKCs in ROM under the assumption of OW-PCA of the primitives. Fig. 4.10 illustrates the structure of it. It can also be applied to the fully trapdoor primitives and the deterministic primitives under the assumption of OW-CPA of the primitives. For the deterministic primitives, it outputs the randomness for the probabilistic primitives as a part of the ciphertexts. The advantage of β is that the ciphertext size becomes smaller than the other conversions for the partially trapdoor primitives. We will show the comparative results in Section 4.5.3.

Conversion γ

This is also a conversion for the fully trapdoor primitives proposed by us in [11]. It can generate IND-CCA2 PKCs in ROM under the assumption of OW-CPA of the primitives. Fig. 4.11 illustrates the structure of it. In the same way as α, it can also be applied to the deterministic primitives. Unfortunately, it cannot be applied to the partially trapdoor ones since the random inputs cannot be inverted in them. The advantage of γ is that the ciphertext size becomes smaller than α. We will show the comparative results in Section 4.5.3.

OAEP++

This is a conversion for the deterministic primitives. It is a very slight extension of OAEP and a variant of γ for the deterministic primitives. Fig. 4.12 illustrates its structure. When $Len(y_3) = Len(y_1||y_2)$, i.e. $Len(y_4) = 0$, this is equivalent to OAEP. Thus IND-CCA2 is satisfied in ROM under the assumption of PDOW-CPA of the deterministic primitive [21]. Even when $Len(y_1||y_2) > Len(y_3) > Len(y_1)$, IND-CCA2 is satisfied in ROM under the same assumption as PDOW-CPA since we can see that the underlying primitive function takes $(y_3||y_4)$ as its input and then outputs $(\mathcal{E}(y_3)||y_4)$. The point of this conversion is that IND-CCA2 is satisfied in ROM under the assumption of OW-CPA when $Len(y_3) \leq Len(y_1)$ holds due to Theorem 1 in Section 4.5.2.

4.5.2 Security of Our Conversions

In this section, we show how to prove the security of our conversions. We will start at describing both the strongest security notion and the random oracle model.

Strongest Security Notion

In this section, we describe the strongest security notion IND-CCA2, the indistinguishability of encryption against adaptive chosen ciphertext attacks.

In the notion of the indistinguishability of encryption [24], an adversary $\mathcal{A}$ selects two distinct plaintexts m_0 and m_1 of the same length in the find stage, and then, in the guess stage, $\mathcal{A}$ is given c which is the encryption of m_b where b is either 0 or 1 with the probability of 1/2. Then $\mathcal{A}$ tries to guess b. The advantage of $\mathcal{A}$ is defined by $2Pr(\text{Win}) - 1$ where $Pr(\text{Win})$ denotes the expected probability of $\mathcal{A}$ guessing b correctly. If $\mathcal{A}$ has a decryption oracle D (which rejects invalid ciphertexts or decrypts any other valid ones than the target one c), it is called that this experiment is in the adaptive-chosen-ciphertext scenario. Otherwise, if $\mathcal{A}$ does not have it, it is called that this experiment is in the adaptive-chosen-plaintext scenario.

Random Oracle Model

The Random Oracle Model (ROM) is one of the models to prove the security of cryptosystems. In this model, cryptographic modules, such as hash functions and/or pseudo-random number generators are assumed to be ideal, i.e. they are assumed to return truly random numbers distributed uniformly over the output region for a new query, and to return the same value for the same queries. The outputs of such ideal functions are given by a random oracle that defines the ideal functions.

The following lemma holds in the random oracle model:

Lemma 1 *Suppose that f is a random oracle. Then it is impossible to get any significant information on $f(x)$ without asking x to the oracle, even if one knows all the other input-output pairs of f except x.*

It is obvious that Lemma 1 is true since the output value of f is determined truly at random.

Adaptive-Chosen-Ciphertext Security

At first, we prove Theorem 1, which can be accomplished by proving Lemma 2 and 3.

Theorem 1 *To break the indistinguishability of encryption of OAEP++ using CCA2 is polynomial equivalent in ROM to break OW-CPA of the underlying deterministic function when at least $Len(y_3) \leq Len(y_1)$ holds.*

Lemma 2 (Adaptive-Chosen-Plaintext Security) *Suppose that there exists, for any Hash and any Gen, an algorithm $\mathcal{A}$ which accepts m_0, m_1 and c of conversion OAEP++ where c is the ciphertext of m_b and $b \in \{0,1\}$, asks at most q_G queries to Gen, asks at most q_H queries to Hash, runs in at most τ steps and guesses b with advantage of ϵ. Then one can design an algorithm $\mathcal{B}$ which accepts a ciphertext $\bar{c}$ of the primitive PKC, runs in τ' steps and decrypts it with probability ϵ' where*

$$\begin{aligned} \epsilon' &\geq \epsilon - \frac{q_G}{2^{Len(r)}}, \\ \tau' &= \tau + Poly(n, q_G, q_H) \end{aligned}$$

and $Poly(n, q_G, q_H)$ denotes a polynomial of n, q_G and q_H.

Proof.

The algorithm $\mathcal{B}$ can be constructed as follows. First the algorithm $\mathcal{B}$ simulates both *Gen* and *Hash* referred by the algorithm $\mathcal{A}$. From the assumption of $\mathcal{A}$ in Lemma 2, $\mathcal{A}$ must be able to distinguish b with the advantage of ϵ for any *Gen* and any *Hash* as long as the algorithm $\mathcal{B}$ simulates them correctly.

$\mathcal{B}$ begins by initializing two lists, G-list and H-list, to empty. These G-list and H-list are the tables of inputs and the corresponding outputs for describing *Gen* and *Hash*, respectively. It runs $\mathcal{A}$ as the find-stage mode simulating $\mathcal{A}'s$ oracles as follows. When $\mathcal{A}$ makes an oracle call h of *Hash*, $\mathcal{B}$ provides it with a random string H, and adds h and H to the H-list. Similarly when $\mathcal{A}$ makes an oracle call g of *Gen*, $\mathcal{B}$ provides it with a random string G, and adds g and G to the G-list. Let (m_0, m_1) be the output of $\mathcal{A}$.

Let $y_4 = (y_5||y_2)$, i.e. $(y_1||y_2) = (y_3||y_5||y_2)$. $\mathcal{B}$ chooses $b \in \{0, 1\}$, r and $(y_5||y_2)$ at random, and then defines both *Hash* and *Gen* so that the ciphertext of m_b should be $(\bar{c}||y_5||y_2)$ where $\bar{c}$ is a ciphertext of the primitive PKC which $\mathcal{B}$ wants to decrypt. That is,

$$Gen(r) \stackrel{\text{def}}{=} (Prep(m_b) \quad || \quad Const) \oplus (y_3||y_5) \tag{4.21}$$

$$Hash(y_3||y_5) \stackrel{\text{def}}{=} y_2 \oplus r. \tag{4.22}$$

and $\mathcal{B}$ adds r and $(Prep(m_b)||Const) \oplus (y_3||y_5)$ to the G-list, and $(y_3||y_5)$ and $y_2 \oplus r$ to the H-list. $\mathcal{B}$ runs $\mathcal{A}$ as the guess-stage mode. For these *Gen* and *Hash*, $\mathcal{A}$ must be able to distinguish b with the advantage of ϵ from the assumption in Lemma 2 as long as $\mathcal{B}$ simulates them correctly.[3]

Can $\mathcal{B}$ simulate them correctly for any queries? The answer is "no" since $\mathcal{B}$ does not know y_3, and thus $\mathcal{B}$ cannot simulate *Gen* correctly when r is asked to it. We consider the following two events:

- **AskH** denotes the event that $(y_3||*)$ is asked to *Hash* among the q_H queries to *Hash* and that this query is performed before r is asked to *Gen* where $*$ denotes any string.

- **AskG** denotes the event that r is asked to *Gen* among the q_G queries to *Gen* and that this query is performed before $(y_3||*)$ is asked to *Hash*.

Since $Pr(\text{AskG} \wedge \text{AskH}) = 0$ in the above definition, the following holds

$$\begin{aligned} & Pr(\text{AskG} \vee \text{AskH}) \\ = \; & Pr(\text{AskG}) + Pr(\text{AskH}). \end{aligned} \tag{4.23}$$

Next, we estimate the upper-limit of $Pr(\text{Win})$, the probability of $\mathcal{A}$ guessing b correctly. Since the mapping from $(r||Prep(m_b)||Const)$ to $(y_3||y_5||y_2)$ is bijective defined by *Gen* and *Hash* where Lemma 1 holds, one cannot get any information on the connectivity between $(y_3||y_5||y_2)$ and $(r||Prep(m_b)||Const)$ without asking r to *Gen* or asking $(y_3||y_5)$ to *Hash*. That is, one cannot guess b with a significant probability after the event $(\neg\text{AskG} \wedge \neg\text{AskH})$. After the other event, i.e. after the event $(\text{AskG} \vee \text{AskH})$, $\mathcal{A}$

[3]If $\mathcal{A}$ distinguishes b only for certain combinations of *Hash* and *Gen*, then the fault must be in either *Gen* or *Hash*, or in both. This implies this fault can be easily removed just avoiding these combinations of *Gen* and *Hash*. Otherwise, i.e. if $\mathcal{A}$ distinguishes b for any *Hash* and any *Gen*, the fault must be in the conversion structure itself.

might guess b with more significant probability. By assuming this probability to be 1, the upper-limit of $Pr(\text{Win})$ is obtained as follows:

$$\begin{aligned} Pr(\text{Win}) &\leq Pr(\text{AskG} \vee \text{AskH}) \\ &\quad + \frac{(1 - Pr(\text{AskG} \vee \text{AskH}))}{2} \\ &\leq \frac{Pr(\text{AskG} \vee \text{AskH}) + 1}{2}. \end{aligned} \tag{4.24}$$

From the definition of advantage, i.e. $Pr(\text{Win}) = (\epsilon + 1)/2$, the following relationship holds

$$Pr(\text{AskG} \vee \text{AskH}) \geq \epsilon. \tag{4.25}$$

Since r is chosen at random by $\mathcal{B}$, $\mathcal{A}$ cannot know it (without asking $(y_3||y_5)$ to $Hash$). Thus the probability of one query to Gen accidentally being r is $1/2^{Len(r)}$, and then that of at most q_G queries is given by

$$\begin{aligned} &Pr(\text{AskG}) \\ &\leq 1 - \left(1 - \frac{1}{2^{Len(r)}}\right)^{q_G} \leq \frac{q_G}{2^{Len(r)}}. \end{aligned} \tag{4.26}$$

The algorithm $\mathcal{B}$ can simulate both Gen and $Hash$ correctly unless the event AskG happens. And then, after the event AskH, $\mathcal{B}$ can recover the whole plaintext of the target ciphertext $\bar{c}$ of the primitive PKC. From (4.23), (4.25) and (4.26), the lower-limit of this probability is given by

$$\begin{aligned} &Pr(\neg\text{AskG} \wedge \text{AskH}) \\ &= Pr(\text{AskH}) \\ &= Pr(\text{AskG} \vee \text{AskH}) - Pr(\text{AskG}) \\ &\geq \epsilon - \frac{q_G}{2^{Len(r)}}. \end{aligned} \tag{4.27}$$

The number of steps of $\mathcal{B}$ is at most $\tau + (T_{Enc} + T_H) \cdot q_H + T_G \cdot q_G$ where T_G is both for checking whether a query to Gen is new or not and for returning the corresponding value, and then T_H is that of $Hash$. T_{Enc} is the number of steps for checking whether a new query h_j to $Hash$ satisfies the event AskH. Since these parameters, T_{Enc}, T_G and T_H can be written in a polynomial of n, q_G and q_H, the total number of steps of $\mathcal{B}$ is also written in a polynomial of them.

□

Lemma 3 (Adaptive-Chosen-Ciphertext Security) *Suppose that there exists, for any Hash and Gen, an algorithm $\mathcal{A}$ which accepts m_0, m_1 and c of OAEP++, asks at most q_G queries to Gen, asks at most q_H queries to Hash, asks at most q_D queries to a decryption oracle D, runs in at most τ steps and guesses b with advantage of ϵ. Then one can design an algorithm $\mathcal{B}$ which accepts a ciphertext $\bar{c}$ of the primitive PKC, runs in τ' steps and decrypts it with probability ϵ' where*

$$\begin{aligned}\epsilon' &\geq \epsilon - \frac{q_G}{2^{Len(r)}} - \frac{q_D(q_G+1)}{2^{Len(r)}} - \frac{q_D}{2^{Len(Const)}}, \\ \tau' &= \tau + Poly(n, q_G, q_H, q_D)\end{aligned}$$

and $Poly(n, q_G, q_H, q_D)$ denotes a polynomial of n, q_G, q_H and q_D.

Proof.

From the assumption of $\mathcal{A}$ in Lemma 3, $\mathcal{A}$ must be able to distinguish the given ciphertext with advantage of ϵ as long as $\mathcal{B}$ simulates them correctly. How to simulate both *Gen* and *Hash* is the same as in the proof of Lemma 2. The decryption oracle D can be simulated using the following plaintext-extractor. It accepts a ciphertext, say $(\bar{c}'||y_5'||y_2')$, and then either outputs the corresponding plaintext or rejects it as an invalid ciphertext.

It works as follows. Let g_i and G_i denote the i-th pair of query and its answer for *Gen*. And then let h_j and H_j denote the j-th pair of query and its answer for *Hash*. From the queries and the answers obtained while simulating *Gen* and *Hash*, the plaintext-extractor finds y_3' satisfying below:

$$y_3' = Msb_{k_1}(h_j) \tag{4.28}$$

$$\bar{c}' = \mathcal{E}(y_3') \tag{4.29}$$

where k_1 is the bit length of the input size for the primitive function. If found, it evaluates $H' := Hash(y_3'||y_5')$, $G' := Gen(H' \oplus y_2')$ and then checks whether $Lsb_{Len(Const)}(G' \oplus y_2') \oplus (y_3'||y_5')) = Const$. If so it outputs $Msb_{Len(m')}(G' \oplus y_2') \oplus (y_3'||y_5'))$. Otherwise, it rejects $(\bar{c}'||y_5'||y_2')$.

If $\mathcal{A}$ asks a valid ciphertext to D without asking $(y_3'||*)$ to *Hash*, it rejects the valid ciphertext, and therefore does not simulate D correctly. However it is a small chance for $\mathcal{A}$ to generate it without asking it. Since the definition of "valid" is to satisfy

$$\begin{aligned}&Lsb_{Len(Const)}(Gen(Hash(y_3'||y_5') \oplus y_2') \\ &= Const \oplus Lsb_{Len(Const)}(y_3'||y_5')\end{aligned} \tag{4.30}$$

and, from Lemma 1, it is impossible for $\mathcal{A}$ to know whether (4.30) is true or not without asking $(y_3'||y_5')$ to *Hash*.

We evaluate the possibility that one ciphertext $c' = (\bar{c}'||y_5'||y_2')$ can be valid without asking $(y_3'||y_5')$ to $Hash$. We consider the following events

- **AskG'** denotes the event that $(Hash(y_3'||y_5') \oplus y_2')$ is asked to Gen among at most q_G queries from $\mathcal{A}$.
- **AskH'** denotes the event that $(y_3'||*)$ is asked to $Hash$ among at most q_H queries from $\mathcal{A}$.
- **ValidR1** denotes the event that the given ciphertext satisfies

$$\begin{aligned} & Hash(y_3'||y_5') \oplus y_2' \\ & \quad = \ Hash(y_3||y_5) \oplus y_2, \qquad (4.31) \\ & Lsb_{Len(Const)}(y_3'||y_5') \\ & \quad = \ Lsb_{Len(Const)}(y_3||y_5), \qquad (4.32) \\ & (y_2', Msb_{Len(m')}(y_3'||y_5')) \\ & \quad \neq \ (y_2, Msb_{Len(Prep(m_b))}(y_3||y_5)) \qquad (4.33) \end{aligned}$$

and thus (4.30) where y_2, y_3 and y_5 are variables of a valid challenge ciphertext satisfying (4.30).

- **ValidC1** denotes the event that the given ciphertext satisfies both (4.30) and

$$Hash(y_3'||y_5') \oplus y_2' \ \neq \ Hash(y_3||y_5) \oplus y_2. \qquad (4.34)$$

- **Valid1** denotes the event that the given ciphertext satisfies (4.30). Note that

$$\begin{aligned} & Pr(\text{Valid1}) \\ & = \ Pr(\text{ValidR1} \vee \text{ValidC1}) \\ & = \ Pr(\text{ValidR1}) + Pr(\text{ValidC1}|\neg\text{ValidR1}) \\ & \quad \cdot Pr(\neg\text{ValidR1}). \qquad (4.35) \end{aligned}$$

- **Fail1** denotes the event that the above plaintext-extractor outputs a wrong answer against one given ciphertext to D.

Since it does not return any plaintext from an invalid ciphertext, and also it returns the correct answer after the events AskH'. Thus

$$\begin{aligned} Pr(\text{Fail1}) \ = \ & Pr(\text{Valid1}|\text{AskG'} \wedge \neg\text{AskH'}) \\ & \cdot Pr(\text{AskG'} \wedge \neg\text{AskH'}) \\ & + Pr(\text{Valid1}|\neg\text{AskG'} \wedge \neg\text{AskH'}) \\ & \cdot Pr(\neg\text{AskG'} \wedge \neg\text{AskH'}) \qquad (4.36) \end{aligned}$$

where

$$\begin{aligned} Pr(\text{AskG'} \wedge \neg\text{AskH'}) &\leq \frac{q_G}{2^{Len(r')}} && (4.37)\\ Pr(\text{Valid1}|\text{AskG'} \wedge \neg\text{AskH'}) &\leq 1 && (4.38)\\ Pr(\neg\text{AskG'} \wedge \neg\text{AskH'}) &\leq 1 && (4.39)\end{aligned}$$

and

$$\begin{aligned} &Pr(\text{Valid1}|\neg\text{AskG'} \wedge \neg\text{AskH'})\\ &= Pr(\text{ValidR1}|\neg\text{AskG'} \wedge \neg\text{AskH'})\\ &\quad + Pr(\text{ValidC1}|\neg\text{ValidR1} \wedge \neg\text{AskG'} \wedge \neg\text{AskH'})\\ &\quad \cdot Pr(\neg\text{ValidR1}|\neg\text{AskG'} \wedge \neg\text{AskH'}). \qquad (4.40)\end{aligned}$$

Since $Pr(\text{ValidR1}|\neg\text{AskG'} \wedge \neg\text{AskH'}) = \frac{1}{2^{Len(r')}}$ and $Pr(\text{ValidC1}|\neg\text{ValidR1} \wedge \neg\text{AskG'} \wedge \neg\text{AskH'}) = \frac{1}{2^{Len(Const)}}$, the upper-limit of $Pr(\text{Fail1})$ is given by

$$Pr(\text{Fail1}) \leq \frac{q_G + 1}{2^{Len(r')}} + \frac{1}{2^{Len(Const)}}. \qquad (4.41)$$

Next, we consider the following event Fail where

- **Fail** denotes the event that the above plaintext-extractor outputs at least one wrong answer against at most q_D queries to D.

The upper-limit of $Pr(\text{Fail})$ is given by

$$\begin{aligned} Pr(\text{Fail}) &\leq 1 - (1 - Pr(\text{Fail1}))^{q_D}\\ &\leq \frac{q_D(q_G + 1)}{2^{Len(r')}} + \frac{q_D}{2^{Len(Const)}}. \qquad (4.42)\end{aligned}$$

Unless either Fail or AskG happens, $\mathcal{B}$ can correctly simulate the oracles referred by $\mathcal{A}$. In addition, when AskH happens, $\mathcal{B}$ can recover the whole plaintext of $\bar{c}$, the ciphertext of the primitive PKC. The lower-limit of this probability $Pr(\text{AskH} \wedge \neg\text{AskG} \wedge \neg\text{Fail})$ is given by

$$\begin{aligned} &Pr(\text{AskH} \wedge \neg\text{AskG} \wedge \neg\text{Fail})\\ &= Pr(\text{AskH} \wedge \neg\text{AskG})\\ &\quad - Pr(\text{AskH} \wedge \neg\text{AskG} \wedge \text{Fail})\\ &\geq Pr(\text{AskH} \wedge \neg\text{AskG}) - Pr(\text{Fail})\\ &\geq \epsilon - \frac{q_G}{2^{Len(r)}} - \frac{q_D(q_G + 1)}{2^{Len(r)}} - \frac{q_D}{2^{Len(Const)}}. \qquad (4.43)\end{aligned}$$

The number of steps of $\mathcal{B}$ is at most $\tau + (T_{Enc} + T_H) \cdot q_H + T_G \cdot q_G + T_D \cdot q_D$ where T_{Enc}, T_G and T_H are the same as the parameters in the proof of Lemma 2. The number of steps T_D is that of the knowledge-extractor to verify whether (4.30) holds and then to return the result. Since these parameters, T_{Enc}, T_G, T_H and T_D can be written in a polynomial of n, q_G, q_H and q_D, the total number of steps of $\mathcal{B}$ is also written in a polynomial of them.

$\square$

In the same way, the lower limit of ϵ's for conversions α, β and γ are given by

$$\epsilon' \geq \epsilon - \frac{q_H}{2^{Len(r)+1}} - \frac{q_D}{2^{Len(Hash)}}, \tag{4.44}$$

$$\epsilon' \geq \epsilon - \frac{q_G}{2^{Len(r)}} - \frac{q_D(q_G+1)}{2^{Len(r)}} - \frac{q_D}{2^{Len(y_2)}} \tag{4.45}$$

and

$$\epsilon' \geq \epsilon - \frac{q_G}{2^{Len(r)}} - \frac{q_D(q_G+1)}{2^{Len(r)}} - \frac{q_D}{2^{Len(Const)}}, \tag{4.46}$$

respectively.

4.5.3 Application to Code-Based PKCs

In this section, we apply all the conversions we introduced in Section 4.4 and 4.5 to the code-based PKCs. We exclude the Fujisaki-Okamoto simple conversion since it requires IND-CPA and both the McEliece PKC and the Niederreiter PKC do not satisfy IND-CPA. The Niederreiter PKC accepts all the conversions requiring OW-CPA or OW-PCA and being applicable to the deterministic primitives. The McEliece PKC accepts all the conversions requiring OW-CPA or OW-PCA and being applicable to the fully-trapdoor primitives.

We compare the data redundancy (which is defined by the difference between the ciphertext size and the plaintext size) among them in Table 4.1 and 4.2. Table 4.1 shows that our conversion γ makes the data redundancy for the enhanced McEliece PKC most compact. Table 4.2 shows that the OAEP variants including our OAEP++ make it for the enhanced Niederreiter PKC most compact. Both OAEP and OAEP+, however, have a disadvantage that the acceptable plaintext size is too small to encrypt something. For example, Niederreiter-{OAEP,OAEP+} cannot encrypt more than 7-bits and 20-bits for $(n,k) = (2048, 1289)$ and $(n,k) = (4096, 2560)$, respectively. These sizes are too small to encrypt even one 40-bit session key. While OAEP++ and OAEP^{++} do not have the problem, OAEP^{++} has another disadvantage that it requires non-malleable block cipher. Thus OAEP++ fits with the Niederreiter PKC best.

4.6 Summary

We carefully reviewed the currently known attacks to the code-based PKCs and then showed that, without any help of decryption oracles and any knowledge on the plaintexts, no polynomial-time algorithm is known for inverting the code-based PKCs (whose parameters are carefully chosen). Under the assumption that this inverting problem is hard, we investigated, in the random oracle model, how to convert this hard problem into the hard problem of breaking the indistinguishability of encryption with CCA2. While some of the generic conversions are applicable to the code-based PKCs, they have a disadvantage in data redundancy. A large amount of redundant data is needed for them since the plaintext size of them is relatively large. On the other hand, our conversions can reduce the redundancy.

Table 4.1: Comparison of Enhanced McEliece PKCs Satisfying IND-CCA2

Conversion Scheme	Assumption of Primitive Function	Binary Work Factor*2	2^{62}	2^{106}	2^{199}
		Data Redundancy*1			
		n	1024	2048	4096
		k	644	1289	2560
		t	38	69	128
Pointcheval's [66]	OW-CPA	$n + Len(r)$	1148	2260	4494
REACT [63]	OW-PCA*3	$n + Len(Hash())$	1148	2260	4494
Fujisaki -Okamoto's [20]	OW-CPA	n	1024	2048	4096
α (Our Proposal)	OW-PCA*3 & FT*4	$n - k + Len(r)$	504	971	1934
β (Our Proposal)	OW-PCA*3	$n - k + Len(r)$	504	971	1934
γ (Our Proposal)	OW-CPA & FT*4	$n - k - \lfloor \log_2 \binom{n}{t} \rfloor + Len(r) + Len(Const)$	398	751	1516
Original McEliece	None	$n - k$	380	759	1536

*1: Ciphertext Size - Plaintext Size. The numerical results are obtained under the setting that $Len(r) = Len(Const) = Len(Hash()) = Len(Hash'()) = 2 \cdot \log_2 W$ where W is the binary working factor for the corresponding parameter (n, k, t).

*2: The binary work factor to break OW-CPA.

*3: For the McEliece PKC, OW-PCA is equivalent to OW-CPA.

*4: Fully trapdoor, i.e. random inputs for the cryptosystem can be recovered.

Table 4.2: Comparison of Enhanced Niederreiter PKCs Satisfying IND-CCA2

Conversion Scheme	Assumption of Primitive Function	Binary Work Factor[*2]	2^{62}	2^{106}	2^{199}
		Data Redundancy[*1]			
		n	1024	2048	4096
		k	644	1289	2560
		t	38	69	128
Pointcheval's [66]	OW-CPA	$n - k + Len(Hash()) + Len(r)$	628	1183	2332
REACT [63]	OW-PCA[*3]	$n - k + Len(Hash())$	504	971	1934
Fujisaki -Okamoto's [20]	OW-CPA[*4]	$n - k + Len(Hash())$	504	971	1934
Bellare -Rogaway's [3]	OW-CPA & Deterministic	$n - k + Len(Hash())$	504	971	1934
OAEP [4]	PDOW-CPA[*5] & Deterministic	$n - k - \lfloor \log_2 \binom{n}{t} \rfloor + Len(r) + Len(Const)$	- (0)[*6]	751 (7)[*6]	1516 (20)[*6]
OAEP+ [72]	OW-CPA & Deterministic	$n - k - \lfloor \log_2 \binom{n}{t} \rfloor + Len(r) + Len(Hash'())$	- (0)[*6]	751 (7)[*6]	1516 (20)[*6]
OAEP++ [39] (Our Proposal)	OW-CPA[*7] & Deterministic	$n - k - \lfloor \log_2 \binom{n}{t} \rfloor + Len(r) + Len(Const)$	398	751	1516
OAEP^{++} [33]	OW-CPA & Deterministic & (NM-CPA BC)	$n - k - \lfloor \log_2 \binom{n}{t} \rfloor + Len(r) + Len(Const)$	398	751	1516
Primitive Niederreiter	-	$n - k - \lfloor \log_2 \binom{n}{t} \rfloor$	150	328	720

*1: Ciphertext Size - Plaintext Size. The numerical results are obtained under the setting that $Len(r) = Len(Const) = Len(Hash()) = Len(Hash'()) = 2 \cdot \log_2 W$ where W is the binary working factor for the corresponding parameter (n, k, t).

*2: The binary work factor to break OW-CPA.

*3: For the Niederreiter PKC, OW-PCA is equivalent to OW-CPA.

*4: For deterministic functions, its structure is almost the same as the Bellare-Rogaway conversion.

*5: For the Niederreiter PKC, PDOW-CPA is equivalent to OW-CPA.

*6: The maximum plaintext size the scheme can encrypt by itself. The others support variable length encryption.

*7: When $Len(y_3) > Len(y_1)$, the assumption is PDOW-CPA. Note that PDOW-CPA is equivalent to OW-CPA for the Niederreiter PKC.

Chapter 5

Password-Authenticated Key-Exchange

This chapter also deals with the attacks over network, but by using PAKE (Password-Authenticated Key-Exchange) instead of PKC (Public-Key Cryptosystems). The advantage of PAKE compared with PKC is that users do not need to manage or verify any certificates, private-keys and public-keys. Thus users' burden can be reduced using PAKE. We propose a pretty-simple PAKE that is proven to be secure in the standard model under the following three assumptions. (1) DDH (Decision Diffie-Hellman) problem is hard. (2) The entropy of the password is large enough to avoid on-line exhaustive search. (3) MAC is unforgeable.

5.1 Overview

We consider the following password-authenticated key-exchange protocol, by which two entities can share a fresh authenticated session-key (being secure against off-line attacks) by using a pre-shared human-memorable password (or pass phrases), which may be insecure against off-line attacks but secure against on-line attacks.

The on-line attack is a serial exhaustive search for a secret performed on-line using a server that verifies the secret (see Section 5.2), and the off-line attack is that performed off-line in parallel using recorded transcripts of a protocol. While the on-line attacks can be prevented by letting the server take appropriate intervals between invalid trials, the off-line attacks cannot be prevented by such measures since the attack is performed off-line and independently of the server. Thus the off-line attacks are critical to most of the protocols using human-memorable passwords not having enough entropy to avoid off-line exhaustive search.

While PKI (Public-Key Infrastructures) can realize an authenticated key-exchange or key-transport (being secure against off-line attacks) like SSH (Secure SHell), SSL/TLS (Secure Socket Layer/Transport Layer Security), Station-to-Station protocol [13] and the protocols in [27] do, we have to recall that the receivers of public-keys must verify them using the fingerprints (digests) of them or the verification keys of digital signatures attached with them. This means the entities must carry about something, which is hard to remember. On the other hand, PAKE (Password-Authenticated Key-Exchange) protocols do not require their entities to carry something hard to remember (except a password) to verify something.

The studies on the PAKE with formal security proof have appeared in [23, 48, 47, 34, 2, 8, 49, 43]. Unfortunately, they are either by far inefficient or the proofs are given only in the random oracle model. In the random oracle model, the mapping of the underlying hash and encryption functions is assumed not to be fixed in advance, and then gradually determined by the random oracle at random every after the evaluation of them. And then simulators (for proving the security reduction) are assumed to know all the evaluated input-output pairs of the functions by simulating the random oracles [3]. While the proof in the random oracle model may give one reason to conjecture that the practical version (which uses conventional fixed functions instead of random oracles) might also be secure, it does not give any formal validation of the security of the practical version.

On the other hand, [23, 34] give their security proofs in the standard model where the mapping of the underlying hash and encryption functions is fixed in advance, and then simulators for showing the security reduction do not need to know the evaluated input-output pairs of the functions. Unfortunately, the protocol proposed in [23] is too inefficient to use in practice since it employs techniques from generic multi-party computations, such as non-malleable commitments, secure polynomial evaluations and zero-knowledge proofs. While [34] is more efficient than [23], it still requires large communication costs and computation costs.

In this paper, we propose a more efficient protocol that is also provably secure in the standard model. Comparative results with the previous schemes [23, 34] are summarized in Table 5.1. As shown in the table, our protocol is efficient in both the communication costs and the computation costs. It requires only about 2.34 modular exponentiations of each entity whereas more than 6.5 modular exponentiations are required in the previous schemes. If pre-computation is used, ours requires only 1 and 2 modular exponentiations of the server and of the client respectively, whereas more than 3.2 and 4.2 modular exponentiations are required of them respectively in the previous schemes. (The difference between the server and the client in the pre-computation phase is whether passwords are stored in advance or given every time.)

Table 5.1: Comparison of PAKEs proven to be secure in the standard model

Schemes	Computation Costs*1		Communication Costs*2		Core Hard Problems*3
	Client C	Server S	C to S	S to C	
Efficient Construction of [23]*4	*5 $\geq (m+1)n$, ($\geq mn$)	*5 $\geq 2n$, ($\geq n$)	*5 $\geq (m+1)n$	*5 $\geq 2m+n$	ITP, PR and DDH *6
Scheme in [34]	≥ 7.75, (≥ 4.29)	≥ 6.58, (≥ 3.29)	≥ 6	5	DDH
Our Proposal	2.34, (2)	2.34, (1)	2	2	DDH

*1: The number of modular exponentiations where the costs for one simultaneous calculation of two bases and five bases are converted into 1.17 and 1.29, respectively [57]. The figures in the parentheses are the remaining costs after pre-computation.

*2: The number of data units to be sent where one data unit denotes either a member of the underlying field or one hashed value, such as a MAC.

*3: In addition to the core hard problems, all the schemes commonly require: (1) Passwords chosen securely against on-line attacks, (2) Unforgeable MACs or signatures against chosen-message attacks.

*4: Efficient construction using the polynomial evaluation in [59] and the efficient oblivious transfer in [78].

*5: Only the costs in the pre-key exchange phase are shown. Both n and m depends on the security parameter. Currently, at least $\binom{m}{n_1} > 2^{80}$ must hold for $n \geq n_1$ to make the underlying polynomial reconstruction problem hard (which is required in the efficient polynomial evaluation)[7].

*6: ITP and PR denote Inversion of Trapdoor Permutation and Polynomial Reconstruction, respectively. Inefficient construction of [23] assumes only trapdoor permutations as its core hard problems.

Ours has an advantage in communication costs too. It requires only 4 data unit exchange whereas more than 11 data unit exchange is required in the previous schemes where one data unit denotes either a member of the underlying field or one hashed value, such as a MAC. In addition, our protocol ends in only one round in parallel since both y_1 and y_2 in our protocol can be calculated independently and then sent independently. The implementation overhead of our protocol is very small since the clients and the servers use almost the same algorithm, and it can be obtained with small modification of the widely used Diffie-Hellman key-exchange protocol.

Our protocol has a formal validation of security in the standard model. The intuitive explanation of the result is that even if adversaries can abuse entities as oracles, the possibility for them obtaining some significant information on the session-key of the challenge session can be negligibly small if DDH (Decision Diffie-Hellman) problem is hard, passwords are unguessable with on-line exhaustive search and MACs are selectively unforgeable against partially chosen message attacks, (which is weaker than being existentially unforgeable against chosen message attacks).

This paper consists as follows: in Section 5.2, we explain both on-line and off-line attacks that are crucial to the password-based protocols. Then, in Section 5.3, we propose a pretty-simple protocol which has an immunity against off-line attacks. And finally, in Section 5.5, we show the formal validation of security of our protocol in the standard model.

5.2 On-line and Off-line Attacks

Since on-line attacks and off-line attacks are crucial to the password-based protocols, we explain them in this section using some examples.

At first, we consider the following password-based challenge-response protocol where a server gives a random challenge r to a client, and then the client returns the server $res := E_{pass}(r)$, the encryption of r using a pre-shared (hashed) password $pass$ as its symmetric key. An adversary, in the on-line attack, runs a protocol with the server impersonating the client, and then tries guessed passwords $pass'$ on-line returning $res' := E_{pass'}(r)$ to the server. If it is accepted, $pass'$ is the target password with high probability.

While almost all of the password-based protocols accept this kind of attack, it can be prevented by letting the server take appropriate intervals between invalid trials. On the other hand, off-line attacks, described bellow, are more powerful since they cannot be prevented by the above measures. Adversaries, in the off-line attack, firstly obtain valid pairs of r and res by eavesdropping honest executions of the protocol, and then finds $pass'$ satisfying $res = E_{pass'}(r)$ off-line in parallel. Since the attack is performed off-line in parallel and the entropy of a password is usually not large enough, they can find it in a practical time with high probability.

The off-line attack is also applicable to DH-EKE (Diffie-Hellman Encrypted Key-Exchange) [5] if the underlying group size $\log_2 |\mathcal{G}| = \log_2 q$ is smaller than the encryption size[1]. Note that the above condition is usually true when a prime order subgroup and a conventional stream cipher or a block cipher, such as AES, are used. DH-EKE is a protocol, in which two entities exchange $y_1 := E_{pass}(g^{r_1})$ and $y_2 := E_{pass}(g^{r_2})$ respectively,

[1]While DH-EKE is proven to be secure in the random oracle model in [2], the proof is given under the assumption that the underlying group size is at least the same as the encryption size. Thus the proof cannot be applied when the group size is smaller than the encryption size.

and then share $D_{pass}(y_1)^{r_2} = D_{pass}(y_2)^{r_1} = g^{r_1 \cdot r_2}$ as a fresh secret where g is a generator of a finite cyclic group $\mathcal{G} =< g >$, $E_{pass}()$ and $D_{pass}()$ are encryption and decryption functions using a (hashed) password *pass* as its symmetric key. The off-line attack on DH-EKE is performed as follows: adversaries obtain some y_1 and y_2 eavesdropping the protocol, and then see off-line whether $D_{pass'}(y_1)$ and $D_{pass'}(y_2)$ (for obtained y_1 and y_2) represent right members in $\mathcal{G}$ for guessed passwords *pass'*. If at least one of them is not a right member, the guessed password is wrong. By continuing the above process, they can find the correct password.

Our protocol, proposed below, has the immunity against these off-line attacks.

5.3 Proposal: Pretty-Simple PAKE

Our protocol is defined over a finite cyclic group $\mathcal{G} =< g >$ where $|\mathcal{G}| = q$ and q is a large prime (or a positive integer divisible by a large prime). While $\mathcal{G}$ can be a group over an elliptic curve, we assume, in this paper, $\mathcal{G}$ is a prime order subgroup over a finite field F_p. That is, $\mathcal{G} = \{g^i \mod p : 0 \leq i < q\}$ where p is a large prime number, q is a large prime divisor of $p-1$ and g is an integer such that $1 < g < p-1$, $g^q = 1$ and $g^i \neq 1$ for $0 < i < q$. A generator of $\mathcal{G}$ is any element of $\mathcal{G}$ except 1.

Both g and h are two generators of $\mathcal{G}$, chosen so that its DLP (Discrete Logarithm Problem), i.e. calculating

$$a = \log_g h, \tag{5.1}$$

should be hard[2] for each entity. Both g and h may be given as system parameters or chosen with the negotiation between entities. For example, g is a random generator of $\mathcal{G}$ and $h := Hash(g)^{(p-1)/q} \mod p$, or one entity $\mathcal{A}$ chooses $g := g_b^{s_1}$ for a random $s_1 \in (Z/qZ)^*$ and a public generator g_b, and then sends the commitment $Hash(g)$ to the other entity $\mathcal{B}$, $\mathcal{B}$ replies $h := g_b^{s_2}$ for a random $s_2 \in (Z/qZ)^*$, and finally $\mathcal{A}$ reveals g to $\mathcal{B}$.

The protocol consists of the following three phases: a secrecy-amplification phase, a verification phase and a session-key generation phase. In the secrecy-amplification phase, the secrecy of the pre-shared weak secret, i.e. a human memorable password that may be vulnerable against off-line attacks, is amplified to a strong secret (we call it a keying material) that is secure even against off-line attacks. In the verification phase, entities confirm whether they can share the same keying material or not using a challenge-response protocol with the keying material as its key. In the session-key generation phase, a session-key is generated using the keying material.

[2]It is reasonable to assume that DLP is hard since our protocol is based on the difficulty of DDH (Decision Diffie-Hellman) problem, and DLP is harder than DDH.

$$\begin{array}{lcl}
\text{Client (Party A)} & & \text{Server (Party B)} \\
r_1 \in (Z/qZ)^* & \xrightarrow{y_1 := g^{r_1} \cdot h^{pass_c}} & r_2 \in (Z/qZ)^* \\
 & \xleftarrow{y_2 := g^{r_2} \cdot h^{pass_s}} & \\
km_c := (y_2 \cdot h^{-pass_c})^{r_1} & & km_s := (y_1 \cdot h^{-pass_s})^{r_2}
\end{array}$$

Figure 5.1: Secrecy-amplification phase of our protocol

$$\begin{array}{rcl}
\text{Client (Party A)} & & \text{Server (Party B)} \\
km_c & \xleftarrow{v_2 := MAC_{km_s}(Tag_s||y_1||y_2)} & km_s \\
\text{If } v_2 = MAC_{km_c}(Tag_s||y_1||y_2), & \xrightarrow{v_1 := MAC_{km_c}(Tag_c||y_1||y_2)} & \text{If } v_2 = MAC_{km_c}(Tag_s||y_1||y_2), \\
sk_c := MAC_{km_c}(Tag_{sk}||y_1||y_2). & & sk_s := MAC_{km_s}(Tag_{sk}||y_1||y_2).
\end{array}$$

Figure 5.2: Verification phase and session-key generation phase of our protocol

5.3.1 Secrecy-Amplification Phase

The secrecy-amplification phase is illustrated in Fig. 5.1. The client chooses a random number $r_1 \in (Z/qZ)^*$ and then calculates $y_1 := g^{r_1} \cdot h^{pass_c}$ using its (hashed) password $pass_c$, which is shared with the server. It sends y_1 to the server. The server also calculates $y_2 := g^{r_2} \cdot h^{pass_s}$ using its (hashed) password $pass_s$ (shared with the client) and a random number $r_2 \in (Z/qZ)^*$, and then sends it to the client The client's keying material is $km_c = (y_2 \cdot h^{-pass_c})^{r_1}$ and the server's one is $km_s = (y_1 \cdot h^{-pass_s})^{r_2}$.

Only when they use the same password, they can share the same keying material. Otherwise guessing the other's keying material is hard due to the DLP between g and h (see also Section 5.5.1). Adversaries cannot determine the correct password of the other entity with off-line attacks since they cannot know the keying material of it, which is required to narrow down the password.

This phase ends in only one pass in parallel since both y_1 and y_2 can be calculated and sent independently (where g^{r_1} and y_2 are pre-computable). This speeds up the protocol. The implementation cost of this phase is very low since it can be obtained with a very small modification of widely used Diffie-Hellman key exchange protocols.

5.3.2 Verification Phase

This phase is illustrated in Fig. 5.2. In this phase, entities verify whether they share the same keying material or not with a challenge-response protocol using the keying material calculated in the secrecy-amplification phase.

The client and the server calculate

$$v_1 := MAC_{km_c}(Tag_c||y_1||y_2) \tag{5.2}$$
$$v_2 := MAC_{km_s}(Tag_s||y_1||y_2) \tag{5.3}$$

using a MAC generation function $MAC_k()$ and the keying materials as its key k. Both Tag_s and Tag_c are pre-determined distinct values, e.g. $Tag_s = (ID_c||ID_s||00)$ and $Tag_c = (ID_c||ID_s||01)$ where ID_c and ID_s are IDs of the client and the server. The client and the server exchange v_1 and v_2 each other, and then they verify $v_1 = MAC_{km_c}(Tag_s||y_1||y_2)$ and $v_2 = MAC_{km_s}(Tag_c||y_1||y_2)$ respectively. If at least one of them does not hold, the corresponding entities wipe off all the temporally data including the keying materials, and then close the session. Otherwise they proceed to the session-key generation phase.

Adversaries can try off-line exhaustive search for the keying material using $(Tag_c||y_1||y_2)$ and v_1 or $(Tag_s||y_1||y_2)$ and v_2. The success probability achieved within a polynomial time t can be negligible if a strong secret can be shared in the secrecy-amplification phase and an appropriate MAC generation function, whose keys are unguessable, is used.

5.3.3 Session-Key Generation Phase

If the above verification phase succeeds in, the entities generate their session keys using the verified keying materials as follows:

$$sk_s := MAC_{km_s}(Tag_{sk}||y_1||y_2) \tag{5.4}$$
$$sk_c := MAC_{km_c}(Tag_{sk}||y_1||y_2) \tag{5.5}$$

where Tag_{sk} is a pre-determined distinct value from both Tag_{v_2} and Tag_{v_1}, e.g. $Tag_{sk} = (ID_c||ID_s||11)$. The generated session keys are then used in the subsequent application.

The requirement for the MAC generation function in this phase and the previous phase is $\epsilon_{mac}(k_2, t, i)$ given in Definition 8 can be negligibly small for practical security parameter k_2 and i (that is a polynomial of k_2) since if adversaries cannot forge a MAC corresponding to $(Tag_{sk}||y_1||y_2)$ and km_s or km_c with a significant probability, they cannot obtain any significant information of the session-key.

This requirement can be satisfied by using a universal one-way hash function [61] or by using a practical MAC generation function, such as HMAC-SHA-1 [42] (and even

KeyedMD5) so far since no effective algorithms are known so far to make $\epsilon_{mac'}(k_2, t, i)$ non-negligible where $\epsilon_{mac'}(k_2, t, i)$ is given in Definition 9 and it is larger than or equal to $\epsilon_{mac}(k_2, t, i)$.

5.4 Extension to Server Compromise

The system is said to be secure against server compromise if the off-line exhaustive search for the password is the best attack when an adversary obtains a signature of the password of a user. Note that the signature of the password means all the necessary information for the server to verify the user, and it includes enough information to perform the off-line exhaustive search for the password.

If one wants to enhance our protocol to the server compromise, the following extension is available. The server stores $V_s := h^{pass_s}$ as the signature of the password for the user. In the case of authentication, the server generates a random number $r_3 \in (Z/qZ)^*$ in addition to r_2 and sends $y_3 := g^{r_3}$ with y_2. Both the client and the server calculate $km_c := \{(y_2 \cdot h^{-pass_c})^{r_1} || y_3^{pass_c}\}$ and $km_s := \{(y_1 \cdot h^{-pass_s})^{r_2} || V_s^{r_3}\}$, respectively, and then include y_3 in each MAC, such as $MAC_{km}(Tag||y_1||y_2||y_3)$.

5.5 Security Proof of Our Protocol

5.5.1 Replacement of h with g

Before we show the formal security proof of our protocol, we describe why two distinct generators, h and g, should be used (instead of one generator). It is because the following adversary $\mathcal{A}_I$ can narrow down the candidates for the keying material to at most N, the number of the possible passwords, with off-line attacks.

$\mathcal{A}_I$ runs the protocol with the target entity impersonating its partner. For simplicity, we assume $\mathcal{A}_I$ impersonates a client. $\mathcal{A}_I$ generates y_1 using randomly chosen r_1 and $pass_c$, and then sends it to the target. The keying material of the target is $km_s := (y_1 \cdot g^{-pass_s})^{r_2}$, and $\mathcal{A}_I$ can narrow down its candidates to at most N since

$$km_s = (y_2 \cdot g^{-pass_s})^{r_1 + pass_c - pass_s} \tag{5.6}$$

and $\mathcal{A}_I$ knows $pass_c$, r_1 and the candidates for $pass_s$, which is at most N.

If N is in the range of off-line exhaustive search, $\mathcal{A}_I$ can determine the correct one by seeing whether $v_2 = MAC_{km_s}(Tag_s||y_1||y_2)$ holds or not with off-line search.

On the other hand, in our protocol, adversaries have to find $a = \log_g h$ to narrow down the candidates for km_s since the following holds

$$km_s = (y_2 \cdot h^{-pass_s})^{r_1 + a(pass_c - pass_s)}. \quad (5.7)$$

5.5.2 Security Model and Formal Validation of Security

In order to consider a more advantageous situation for adversaries, we assume they have access to the following oracles, which were originally introduced by Bellare et al in [2], but a little bit modified for our protocol.

Execute oracle: It accepts two IDs of entities sharing the same password. Then it carries out a honest execution of the protocol between them, and outputs the corresponding transcript. This oracle ensures that adversaries are able to observe all the transcripts between any entities including the target ones.

Send oracle: It accepts an entity ID and a message that is a part of a transcript. It acts as the entity, and then outputs a completed transcript corresponding to them. This oracle ensures that adversaries are able to run a protocol with any entity impersonating its partner and obtain the corresponding transcripts.

Reveal oracle: It accepts both an entity ID and a session ID, and then reveals the corresponding session-key. (This oracle does not reveal the session-key of the challenge transcript.) Note that a session-key might be leaked out since it is used outside of the protocol in various applications that might deal it insecurely (e.g. by using it as a key of very weak encryption algorithms). Reveal oracle simulates such a situation.

Corrupt oracle: This oracle is used to see whether the protocol satisfies the forward secrecy, i.e. whether the disclosure of a long-lived secret (a password in our protocol) does not compromise the secrecy of the session-keys from earlier runs (even though that compromises the authenticity and thus the secrecy of new runs). It accepts two entity IDs and then reveals the corresponding password shared between them. This oracle can be used after the transcripts related with the target password are generated.

Test$_{sk}$ oracle: This oracle is used to see whether adversaries can obtain some information on the challenge session-key by giving a hint on it to them. It accepts an entity ID in the challenge session, and then flips a coin $b \in \{0, 1\}$. If $b = 0$, it returns the corresponding session-key. Otherwise it returns a random one except the correct session-key. This oracle can be used only once per challenge.

Test$_{km}$ oracle: Since a session-key is generated from a keying material, we prepare this oracle to see whether a strong secret can be generated in the secrecy amplification phase. This oracle accepts both an entity ID and an session ID, and then flips a coin $b \in \{0, 1\}$. If $b = 0$, it returns the corresponding keying material. Otherwise, it returns a random one except the correct keying material. Note that adversaries are not allowed to distinguish the obtained information from this oracle using $(Tag_c||y_1||y_2)$ and v_1 or $(Tag_s||y_1||y_2)$ and v_2 since it is given to see whether a strong secret can be generated in the secrecy amplification phase.

Using the above oracles, adversaries suppose to try to distinguish a session-key given by Test$_{sk}$ oracle.

At first, we define the followings:

Definition 5 (Advantage) *Let $Pr(Win)$ denote the probability that an algorithm $\mathcal{A}$ can distinguish whether a given key is the correct session-key or not. Then $Adv_{\mathcal{A}}^{ind_{sk}}$, the advantage of $\mathcal{A}$ distinguishing the session-key, is given by*

$$Adv_{\mathcal{A}}^{ind_{sk}} = 2Pr(Win) - 1. \tag{5.8}$$

Definition 6 (DDH Problem) *Given $g_b \in \mathcal{G}$ and $d = (d_1, d_2, d_3) = (g_b^{x_1}, g_b^{x_2}, g_b^{x_3})$ where x_3 is either $x_1 \cdot x_2$ or not with probability $1/2$, then decide whether $g_b^{x_3} = g_b^{x_1 \cdot x_2}$ or not.*

Definition 7 (Probability of Solving DDH Problem) *Let $\epsilon_{ddh}(k_1, t)$ denote the probability that the DDH problem of size $k_1 = \log_2 q$ is solved in a polynomial time t with the best known algorithm.*

The requirement for the MAC generation function in our protocol is $\epsilon_{mac}(k_2, t, i)$, given in the following Definition 8, can be negligibly small for practical security parameter k_2 and i (that is a polynomial of k_2). $\epsilon_{mac}(k_2, t, i)$ is upper bounded by $\epsilon_{mac'}(k_2, t, i)$, which is given in Definition 8 that is a more general definition.

Definition 8 (Selective UnForgeability of a MAC Against Partially Chosen Message Attack) *Let $\epsilon_{mac}(k_2, t, i)$ denote the probability that a k_2 bit length MAC of a given message can be forged in a polynomial time t with the best known algorithm that are allowed to ask at most i (which is a polynomial of k_2) queries to the following MAC generation oracle (which is available in our protocol by abusing entities or using Send, Execute and Reveal oracles). The MAC generation oracle here accepts a message m, entity $\in \{server, client\}$, target $\in \{v, sk\}$ and a bijective function $f()$ and then returns, for randomly chosen r_1 and r_2, $\mathrm{MAC}_{f(km)}(Tag_s||m||g^{r_2})$ if entity $=$ server and target $=$ v, $\mathrm{MAC}_{f(km)}(Tag_c||g^{r_1}||m)$ if entity $=$ client and target $= v$, $\mathrm{MAC}_{f(km)}(Tag_{sk}||m||g^{r_2})$*

if entity = server and target = sk or $\mathrm{MAC}_{f(km)}(Tag_{sk}||g^{r_1}||m)$ *if entity = client and taget = sk, respectively. A MAC is said to be SUF-PCMA (Selectively UnForgeable against Partially Chosen Message Attacks) if* $\epsilon_{mac}(k_2, t, i)$ *is negligibly small.*

Definition 9 (Existential UnForgeability of a MAC Against Chosen Message Attack) *Let* $\epsilon_{mac'}(k_2, t, i)$ *denote the probability that a new MAC-message pair for a* k_2 *bit length MAC can be generated in a polynomial time* t *with the best known algorithm that are allowed to ask at most* i *(which is a polynomial of* k_2*) queries to a MAC generation oracle, which accepts a message* m *and a bijective function* $f()$ *and then returns* $\mathrm{MAC}_{f(km)}(m)$*. A MAC is said to be EUF-CMA (Existential UnForgeable against Chosen Message Attacks) if* $\epsilon_{mac'}(k_2, t, i)$ *is negligibly small.*

Under the following assumption, Theorem 2 is true[3]. The intuitive interpretation of Theorem 2 is that if both N and $|\mathcal{G}|$ are large enough and both $\epsilon_{mac}(k_2, t, q_{se}+2q_{ex}+q_{re}+2)$ and $\epsilon_{ddh}(k_1, t)$ can be negligibly small for appropriate security parameters k_1 and k_2, the advantage for the active adversaries can be bounded by a negligibly small value.

Assumption 1 (Password) *Users' passwords are chosen uniformly at random from a set of cardinality* N*.*

Theorem 2 (Indistinguishability of sk**)** *Suppose the following adversary* $\mathcal{A}$*, which accepts a challenge transcript (that may be obtained by eavesdropping a protocol, impersonating a partner or intruding in the middle of the target entities), and then asks* q_{ex}*,* q_{se} *and* q_{re} *queries to the Execute, Send, Reveal oracles respectively, and finally is given* sk_x *by* $Test_{sk}$ *oracle where* sk_x *is either the target session-key or not with the probability 1/2. Then* $Adv_{\mathcal{A}}^{ind_{sk}}$*, the advantage of it to distinguish whether* sk_x *is the target session key or not in a polynomial time* t *is upper bounded by*

$$\begin{aligned} Adv_{\mathcal{A}}^{ind_{sk}} \leq\ & \varepsilon_{mac}(k_2, t, q_{se} + 2q_{ex} + q_{re} + 2) \\ & +2(q_{se} + q_{ex} + 1) \cdot \varepsilon_{ddh}(k_1, t) \\ & +\frac{2q_{se} + 1}{N} + \frac{2(q_{se} + q_{ex})}{|\mathcal{G}|} \end{aligned} \tag{5.9}$$

where both k_1 *and* k_2 *are the security parameters.*

Proof.

Recall that Win is an event that $\mathcal{A}$ distinguishes sk_x correctly. Win happens either after an event KmUnknown occurs or after its compliment event $\overline{\text{KmUnknown}}$ occurs

[3]Theorem 2 can be extended easily to the case where passwords are chosen non-uniformly since the uniformity assumption of the passwords is just for simplicity.

where $\overline{\text{KmUnknown}}$ is an event that $\mathcal{A}$ obtains some significant information on the keying material km in the secrecy amplification phase, and KmUnknown is an event that $\mathcal{A}$ does not obtains any significant information on the keying material km in the secrecy amplification phase. Thus $Pr(\text{Win})$ is upper bounded by

$$\begin{aligned} Pr(\text{Win}) &= Pr(\text{Win}|\text{KmUnknown})Pr(\text{KmUnknown}) \\ &\quad + Pr(\text{Win}|\overline{\text{KmUnknown}})Pr(\overline{\text{KmUnknown}}) \\ &\leq Pr(\text{Win}|\text{KmUnknown}) + Pr(\overline{\text{KmUnknown}}). \end{aligned} \tag{5.10}$$

We evaluate $Pr(\text{Win}|\text{KmUnknown})$ first. Even if km is unknown, the following two adversaries $\mathcal{A}_{replay}$ and $\mathcal{A}_{mac}$, can distinguish sk_x. $\mathcal{A}_{mac}$ tries to forge a MAC of $(Tag_{sk}||y_1||y_2)$, and then distinguish sk_x. $\mathcal{A}_{replay}$ tries to obtain at least one transcript coinciding with the challenge transcript using Send or Execure oracles, and then obtains the corresponding session-key, which is the same as the challenge session-key, using Reveal oracle.

Let $Pr(\text{Win}_{\mathcal{A}_{replay}})$ and $Pr(\text{Win}_{\mathcal{A}_{mac}})$ denote the probabilities of $\mathcal{A}_{replay}$ and $\mathcal{A}_{mac}$ being able to distinguish sk_x, respectively. $Pr(\text{Win}_{\mathcal{A}_{replay}})$ is upper bounded by

$$Pr(\text{Win}_{\mathcal{A}_{replay}}) \leq \frac{(q_{se} + q_{ex})}{|\mathcal{G}|} \tag{5.11}$$

since $\mathcal{A}_{replay}$ cannot control at least either r_1 or r_2 and can obtain at most $(q_{se} + q_{ex})$ transcripts. The upper bound of $Pr(\text{Win}_{\mathcal{A}_{mac}})$ is given by the following lemma.

Lemma 4 *Suppose the probability that an adversary $\mathcal{A}_{mac}$ can forge a k_2 bit length MAC of a given message in a polynomial time t using i message-MAC pairs without knowing its key is $\epsilon_{mac}(k_2, t, i)$. Then $Pr(\text{Win}_{\mathcal{A}_{mac}})$, the probability of $\mathcal{A}_{mac}$ distinguishing a given session-key without knowing its keying material is upper bounded by $1/2 + \epsilon_{mac}(k_2, t, i)/2$.*

Proof.

The situation where $\mathcal{A}_{mac}$ tries to distinguish a session-key can be divided into the following four cases according to whether a MAC forged by $\mathcal{A}_{mac}$ (of the given message $(Tag_{sk}||y_1||y_2)$) is valid or not, and whether a key given by Test_{sk} oracle is correct or not, i.e. $b = 0$ or $b = 1$.

Let MACValid denote an event that the forged MAC is valid. The best strategy for $\mathcal{A}_{mac}$ to maximize the winning probability to distinguish the given key from Test_{sk} oracle is to return $b = 0$ (with the probability 1) if the generated MAC coincides with the given key, and $b = 1$ (with the probability 1) otherwise since $\mathcal{A}_{mac}$ can only know whether the generated MAC and the given key coincide or not, and then the probabilities they coincide and they do not are given by

$$Pr(b=0,\mathrm{MACValid}) + Pr(b=1,\overline{\mathrm{MACValid}}) \cdot \frac{1}{2^{k_2}-1} \tag{5.12}$$

and

$$Pr(b=0,\overline{\mathrm{MACValid}}) + Pr(b=1,\mathrm{MACValid}) + Pr(b=1,\overline{\mathrm{ForgeMAC}}) \cdot \frac{2^{k_2}-2}{2^{k_2}-1} \tag{5.13}$$

respectively where $Pr(b=0,\mathrm{MACValid}) > Pr(b=1,\overline{\mathrm{MACValid}}) \cdot \frac{1}{2^{k_2-1}}$ and $Pr(b=0,\overline{\mathrm{MACValid}}) > Pr(b=1,\mathrm{MACValid}) + Pr(b=1,\overline{\mathrm{ForgeMAC}}) \cdot \frac{2^{k_2-2}}{2^{k_2-1}}$ hold as long as $\epsilon_{mac}(k_2,t,i) > \frac{1}{2^{k_2}}$.

This give the following probability

$$\begin{aligned} Pr(\mathrm{Win}_{\mathcal{A}_{mac}} &\mid b=0,\mathrm{MACValid}) = 1, & (5.14)\\ Pr(\mathrm{Win}_{\mathcal{A}_{mac}} &\mid b=1,\mathrm{MACValid}) = 1, & (5.15)\\ Pr(\mathrm{Win}_{\mathcal{A}_{mac}} &\mid b=0,\overline{\mathrm{MACValid}}) = 0, & (5.16)\\ Pr(\mathrm{Win}_{\mathcal{A}_{mac}} &\mid b=1,\overline{\mathrm{MACValid}}) = \frac{2^{Len(sk)}-2}{2^{Len(sk)}-1}, & (5.17) \end{aligned}$$

And thus $Pr(\mathrm{Win}_{\mathcal{A}_{mac}})$ is upper bounded by

$$\begin{aligned} &Pr(\mathrm{Win}_{\mathcal{A}_{mac}})\\ &= Pr(\mathrm{Win}_{\mathcal{A}_{mac}}|b=0,\mathrm{MACValid}) \cdot Pr(b=0) \cdot Pr(\mathrm{MACValid})\\ &\quad + Pr(\mathrm{Win}_{\mathcal{A}_{mac}}|b=1,\mathrm{MACValid}) \cdot Pr(b=1) \cdot Pr(\mathrm{MACValid})\\ &\quad + Pr(\mathrm{Win}_{\mathcal{A}_{mac}}|b=0,\overline{\mathrm{MACValid}}) \cdot Pr(b=0) \cdot Pr(\overline{\mathrm{MACValid}})\\ &\quad + Pr(\mathrm{Win}_{\mathcal{A}_{mac}}|b=1,\overline{\mathrm{MACValid}}) \cdot Pr(b=1) \cdot Pr(\overline{\mathrm{MACValid}})\\ &= Pr(\mathrm{MACValid})\\ &\quad + Pr(\mathrm{Win}_{\mathcal{A}_{mac}}|b=1,\overline{\mathrm{MACValid}}) \cdot Pr(b=1) \cdot Pr(\overline{\mathrm{MACValid}})\\ &\le \epsilon_{mac}(k_2,t,i) + \frac{2^{k_2}-2}{2^{k_2}-1} \cdot \frac{1}{2} \cdot \{1-\epsilon_{mac}(k_2,t,i)\}\\ &\le \frac{1}{2} + \frac{\epsilon_{mac}(k_2,t,i)}{2} \end{aligned} \tag{5.18}$$

□

$\mathcal{A}_{mac}$ can obtain at most $q_{se} + 2q_{ex} + q_{re}$ message-MAC pairs using Send, Execute, Reveal oracles, and at most 2 message-MAC pairs from a challenge transcript. Thus

$$i = q_{se} + 2q_{ex} + q_{re} + 2. \tag{5.19}$$

By substituting (5.19) for (5.18) and summing up (5.11) and (5.18), we can obtain

$$Pr(\text{Win}|\text{KmUnknown}) \leq \frac{(q_{se} + q_{ex})}{|\mathcal{G}|} + \frac{1}{2} + \frac{\epsilon_{mac}(k_2, t, q_{se} + 2q_{ex} + q_{re} + 2)}{2}. \tag{5.20}$$

Next we evaluate $Pr(\overline{\text{KmUnknown}})$, the possibility of $\mathcal{A}$ being able to obtain some information on the keying material km in the secrecy amplification phase. In the secrecy amplification phase, $\mathcal{A}$ can obtain g, h, y_1, y_2 (and pre-images of either y_1 or y_2 by impersonating the corresponding entity). The obtained data can be classified into the following four cases according to whether or not the passwords of the two entities coincide with each other, and whether or not the adversary knows the pre-image of either y_1 or y_2.

Case 1: Passwords of the target entity and its parter are different, i.e. $pass_c \neq pass_s$, and the adversary knows the pre-image of neither y_1 nor y_2.

Case 2: Passwords of the target entity and its parter are the same, i.e. $pass_c = pass_s$, and the adversary knows the pre-image of neither y_1 nor y_2.

Case 3: Passwords of the target entity and its parter are different, i.e. $pass_c \neq pass_s$, and the adversary knows the pre-image of either y_1 or y_2.

Case 4: Passwords of the target entity and its parter are the same, i.e. $pass_c = pass_s$, and the adversary knows the pre-image of either y_1 or y_2.

While Case 4 is the most advantageous for $\mathcal{A}$, it happens only when $\mathcal{A}$ inputs the correct password impersonating the parter of the target entity on-line. This probability is bounded by $(q_{se} + 1)/N$ since $\mathcal{A}$ can try at most $q_{se} + 1$ passwords on-line where q_{se} passwords are tried using Send oracle and 1 using the challenge session. The other cases happen with more high probabilities. For example, Case 1 and 2 happen when an adversary eavesdrops a session, or sends modified values of ever used y_1 or y_2, i.e. sends $y_1 \cdot g^{j_1} \cdot h^{j_2} \bmod p$ or $y_2 \cdot g^{j_1} \cdot h^{j_2} \bmod p$ for $j_1, j_2 \in Z/qZ$ to the target entity. Case 3 happens when an adversary generates y_1 (or y_2) from its pre-images and sends it to the target entity.

While Case 1 to 3 happen with high probability, distinguishing the keying material in these cases is as hard as or harder than solving DDH problem. Lemma 5 shows that distinguishing it in Case 2 is as hard as or harder than solving DDH problem.

Lemma 5 *Suppose there exists an algorithm $\mathcal{A}_1$, which accepts a challenge transcript g, h, y_1 and y_2 between the entities sharing the same password, and is given a hint km_x from*

$Test_{km}$ oracle where km_x is either equal to the keying material of the target entity, i.e. km_c or km_s, or not with the probability of $1/2$, and finally distinguishes whether km_x is the correct keying material or not in at most τ steps and with the advantage of ϵ. Then one can construct an algorithm $\mathcal{B}_1$ which runs in τ' steps and solves a given DDH problem with the advantage of ϵ' where

$$\begin{aligned} \epsilon' &= \epsilon, && (5.21) \\ \tau' &= \tau + Poly(k_1) && (5.22) \end{aligned}$$

and $Poly(k_1)$ is a polynomial of a security parameter $k_1 = \log_2 q$.

Proof.

$\mathcal{B}_1$ can be constructed as follows. At first $\mathcal{B}_1$ receives a DDH set g_b and $d = (d_1, d_2, d_3) = (g_b^{x_1}, g_b^{x_2}, g_b^{x_3})$. $\mathcal{B}_1$ chooses a random password $pass_s = pass_c$ and a random generator $h \in \mathcal{G}$, and then gives $g := g_b$, h, $y_1 := d_1 \cdot h^{pass_c}$, $y_2 := d_2 \cdot h^{pass_s}$ and $km_x := d_3$ to $\mathcal{A}_1$. If the answer of $\mathcal{A}_1$ is $km_x = km_c$ (which also means $km_x = km_s$), $\mathcal{B}_1$ returns $d_3 = g_b^{x_1 \cdot x_2}$. Otherwise it returns $d_3 \neq g_b^{x_1 \cdot x_2}$.

$\mathcal{B}_1$ can solve the DDH problem with the same advantage as ϵ since $d_3 = g_b^{x_1 \cdot x_2}$ holds with probability 1 if $km_x = km_s = km_c$. The number of steps required for $\mathcal{B}_1$ is mainly consumed in the calculation of h^{pass_c} and h^{pass_s} which ends in polynomial steps of $k_1 = \log_2 q$. Thus $\tau' = \tau + Poly(k_1)$.

□

Lemma 6 shows that distinguishing the keying material of the server (impersonating a client) in Case 3 is as hard as or harder than solving DDH problem. This also means distinguishing the client's keying material impersonating a server is as hard as or harder than solving DDH problem. (The corresponding proof can be obtained by replacing r_1 and $pass_c$ in the following proof with r_2 and $pass_s$ respectively, due to the symmetry of our protocol.)

Lemma 6 *Suppose there exists an algorithm $\mathcal{A}_2$, which accepts g, h, y_2, y_1, r_1, $pass_c$ and km_x where g, h, y_2 and y_1 are a challenge transcript between entities that does not share the same password, r_1 and $pass_c$ are the pre-image of y_1, and km_x is a hint given by $Test_{km}$ oracle, which is either km_s or not with the probability of $1/2$, and finally distinguishes whether $km_x = km_s$ or not in at most τ steps and with the advantage of ϵ. Then one can construct an algorithm $\mathcal{B}_2$ which runs in τ' steps and solves a given DDH problem with the advantage of ϵ' where*

$$\begin{aligned} \epsilon' &= \epsilon, && (5.23) \\ \tau' &= \tau + Poly(k_1) && (5.24) \end{aligned}$$

and $Poly(k_1)$ is a polynomial of a security parameter $k_1 = \log_2 q$.

$\mathcal{B}_2$ can be constructed as follows. At first $\mathcal{B}_2$ receives a DDH set, g_b and $d = (d_1, d_2, d_3) = (g_b^{x_1}, g_b^{x_2}, g_b^{x_3})$. It chooses a random number $r_1 \in (Z/qZ)^*$, two distinct passwords $pass_c$ and $pass_s$, and then gives $\mathcal{A}_2$ $g := g_b$, $h := d_2$, $y_2 := d_1 h^{pass_s}$, $y_1 := g^{r_1} h^{pass_c}$, r_1, $pass_c$ and $km_x = d_1^{r_1} \cdot d_3^{(pass_c - pass_s)}$. If the answer of $\mathcal{A}_2$ is $km_x = km_s$, $\mathcal{B}_2$ returns $d_3 = g_b^{x_1 \cdot x_2}$. Otherwise it returns $d_3 \neq g_b^{x_1 \cdot x_2}$.

$\mathcal{B}_2$ can solve the DDH problem with the same advantage as ϵ since

$$\begin{aligned} km_x &= d_1^{r_1} \cdot d_3^{(pass_c - pass_s)}, && (5.25)\\ km_s &= g^{x_1 \cdot r_1} \cdot h^{(pass_c - pass_s) x_1} \\ &= d_1^{r_1} \cdot g^{x_2 (pass_c - pass_s) x_1}, && (5.26) \end{aligned}$$

and $d_3 = g_b^{x_1 x_2}$ holds if $km_s = km_x$. The number of steps required for $\mathcal{B}_2$ is mainly consumed in the calculation of y_1, y_2 and km_x which ends in polynomial steps of $k_1 = \log_2 q$. Thus $\tau' = \tau + Poly(k_1)$.

□

Distinguishing the target keying material in Case 1 is as hard as or harder than doing that in Case 3 since the pre-images of y_1 and y_2 are not given to the adversaries in Case 1. The corresponding proof can be obtained simply by removing y_1 and $pass_c$ from the inputs of $\mathcal{B}_2$ in the proof of Lemma 6.

From the above discussion and Definition 7, the probability that one can obtain some information on the keying material from one transcript in Case 1 to 3 is upper bounded by $\epsilon_{ddh}(k_1, t)$. In total, $\mathcal{A}$ can obtain at most $q_{se} + q_{ex} + 1$ transcripts where $q_{se} + q_{ex}$ can be obtained using Send and Execute oracles, and 1 from a challenge transcript. Thus the probability of $\mathcal{A}$ being able to obtain some information on the challenge keying material in Case 1 to 3 is upper bounded by $(q_{se} + q_{ex} + 1) \cdot \epsilon_{ddh}(k_1, t)$. And then the probability of $\mathcal{A}$ being able to obtain it in the secrecy amplification phase is upper bounded by

$$\begin{aligned} & Pr(\overline{KmUnknown}) \\ \leq\ & \frac{q_{se} + 1}{N} + (q_{se} + q_{ex} + 1) \cdot \epsilon_{ddh}(k_1, t). && (5.27) \end{aligned}$$

By substituting (5.20) and (5.27) for (5.10), the upper bound of $Pr(\text{Win})$ is given by

$$\begin{aligned} Pr(\text{Win}) \leq\ & \frac{(q_{se} + q_{ex})}{|\mathcal{G}|} + \frac{1}{2} + \frac{q_{se} + 1}{N} \\ & + \frac{\epsilon_{mac}(k_2, t, q_{se} + 2q_{ex} + q_{re} + 2)}{2} \\ & + (q_{se} + q_{ex} + 1) \cdot \epsilon_{ddh}(k_1, t). && (5.28) \end{aligned}$$

(5.9) can be obtained by substituting (5.28) for (5.8).

□

5.6 Summary

We proposed a pretty-simple password-authenticated key-exchange protocol that is proven to be secure in the standard model (instead of the random oracle model) under the following three assumptions. (1) DDH (Decision Diffie-Hellman) problem is hard. (2) The entropy of the password is large enough to avoid on-line exhaustive search (but not necessarily off-line exhaustive searches). (3) MAC is selectively unforgeable against partially chosen message attacks, (which is weaker than existentially unforgeable against chosen message attacks).

Our protocol is almost as efficient as Diffie-Hellman key-exchange, and can be implemented easily with a small modification of it.

Chapter 6

Conclusion

In this dissertation, we conducted research on remote user authentication schemes that are not only secure against both real and network adversaries, but also relying on no special devices, such as bio-sensors and tamper resistant areas. The real world adversaries are assumed to obtain, with peeping, all the information displayed to and typed by the users, and then obtain, with theft, all their personal devices they have. The network adversaries are assumed to not only eavesdrop all the communication between terminals and servers, but also establish fake servers and let the users connect them. Even under the above scenario, we showed that it is possible to construct secure remote user authentications without using special devices.

The contributions of this dissertation are summarized as follows:

In Chapter 2, we focused on CRHI (Challenge-Response Human Identification) as a countermeasure against real world adversaries, and evaluated the exact resistance against peeping attacks. We proposed to improve it by controlling a history of challenges, and that could successfully increase the number of peeps it can resist.

In Chapter 3, we proposed a further improvement of the resistance against peeps by applying newly proposed VSS (Visual Secret Sharing) that can limit the visible space of the decoded image. We evaluated the visibility of the space, and showed that it can transmit messages to a user in certain position. This result tells us the combination of it and CRHI can deal with the real world adversaries.

In Chapter 4, we considered preventing attacks over network using PKCs (Public-Key Cryptosystems). Since primitive PKCs do not have the desirable securities, we proposed how to convert such primitive PKCs into ideal ones. Precisely, we proposed OAEP++ for deterministic OW-CPA primitives, α and γ for fully trapdoor OW-CPA primitives and β for partially trapdoor OW-PCA primitives. We also evaluated the primitive PKCs based on the decoding problems, and showed they can be ideally strong using our conversions.

The advantage of these PKCs are that no polynomial time algorithm has not been discovered yet to break the underlying problem of them even with the quantum computers, even though the integer factoring and the discrete logarithm would be broken in probabilistic polynomial time with them.

In Chapter 5, we focused PAKE (Password-Authenticated Key-Exchange) to remove the burden of public-key management and certificate-verification from users. We proposed a simple one that is almost as efficient as Diffie-Hellman key-exchange in communication and computation costs, and then proved its security under standard assumptions.

This research still leaves room for further work. One is on the transaction security after user authentication, and another is on the security against stronger attacks, such as terminal compromise or server compromise. The transaction security includes both the secrecy and the integrity of communication after the user authentication. One way to realize this is to share a (huge) code-book between a user and the server. The code-book may be generated by the server and sent to the user in a secure way. The user carries it and uses it to encrypt the input and to decrypt the output of the terminal. Note that these encryption and decryption procedures must be secure against replay attacks. If the terminals are trustworthy, users do not need to carry the code-book with them since it can be displayed securely using LVSVSS (Limiting Visible Space VSS). The point to study will include how to make the code-book compact, e.g. by optimizing the transaction.

Bibliography

[1] C. M. Adams and H. Meijer. "Security-related comments regarding McEliece's public-key cryptosystem". In *Proc. of CRYPTO '87, LNCS 293*, pages 224–228. Springer–Verlag, 1988.

[2] M. Bellare, D. Pointcheval, and P. Rogaway. "Authenticated key exchange secure against dictionary attack". In *Proc. of EUROCRYPT 2000: LNCS 1807*, pages 139–155, 2000.

[3] M. Bellare and P. Rogaway. "Random oracles are practical: A paradigm for designing efficient protocols". In *Proc. of the First ACM CCCS*, pages 62–73, 1993.

[4] M. Bellare and P. Rogaway. "Optimal asymmetric encryption". In *Proc. of EUROCRYPT '94, LNCS 950*, pages 92–111, 1995.

[5] S. Bellovin and M. Merritt. "Encrypted key exchange: Password-based protocols secure against dictionary attacks". In *Proc. of IEEE Symposium on Security and Privacy*, pages 72–84, 1992.

[6] T. Berson. "Failure of the McEliece public-key cryptosystem under message-resend and related-message attack". In *Proc. of CRYPTO '97, LNCS 1294*, pages 213–220. Springer–Verlag, 1997.

[7] D. Bleichenbacher and P. Nguyen. "Noisy polynomial interpolation and noisy chinese remaindering". In *Proc. of EUROCRYPT 2000: LNCS 1807*, pages 53–69, 2000.

[8] V. Boyko, P. MacKenzie, and S. Patel. "Provably secure password authenticated key exchange using Diffie-Hellman". In *Proc. of EUROCRYPT 2000: LNCS 1807*, pages 156–171, 2000.

[9] A. Canteaut and N. Sendrier. "Cryptoanalysis of the original McEliece cryptosystem". In *Proc. of ASIACRYPT '98*, pages 187–199, 1998.

[10] D. Catalano, R. Gennaro, N. H.-Graham, and P. Q. Nguyen. "Paillier's cryptosystem revisited". In *Proc. of ACM 8th Computer and Communication Security Conference*, 2001.

[11] T. Dierks and C. Allen. "The TLS protocol version 1.0". *RFC 2246*, 1999.

[12] W. Diffie and M. Hellman. "New directions in cryptography". *IEEE Trans. IT*, 22(6):644-654, 1976.

[13] W. Diffie, P.C. van Oorschot, and M. Wiener. "Authentication and authenticated key exchanges". *Designs Codes and Cryptography*, 2, 1992.

[14] D. Dolve, C. Dwork, and M. Naor. "Non-malleable cryptography". In *Proc. of the 23rd STOC.* ACM Press, 1991.

[15] T. ElGamal. "A public-key cryptosystem and a signature scheme based on discrete logarithms". In *Proc. of CRYPTO '84*, pages 10–18, 1985.

[16] A. Fiat and A. Shamir. "How to prove yourself". In *Proc. of CRYPTO '86, LNCS 263*, pages 186–194. Springer–Verlag, 1986.

[17] J. B. Fischer and J. Stern. "An efficient pseudo-random generator provably as secure as syndrome decoding". In *Proc. of EUROCRYPT '96, LNCS 1070*, pages 245–255. Springer–Verlag, 1996.

[18] A. O. Freier, P. Karlton, and P. C. Kocher. "The SSL protocol version 3.0". *http://wp.netscape.com/eng/ssl3/*, 1996.

[19] E. Fujisaki and T. Okamoto. "How to enhance the security of public-key encryption at minimum cost". In *Proc. of PKC'99, LNCS 1560*, pages 53–68, 1999.

[20] E. Fujisaki and T. Okamoto. "Secure integration of asymmetric and symmetric encryption schemes". In *Proc. of CRYPTO '99, LNCS 1666*, pages 535–554, 1999.

[21] E. Fujisaki, T. Okamoto, D. Pointcheval, and J. Stern. "RSA-OAEP is secure under the RSA assumption". In *Proc. of CRYPTO 2001, LNCS 2139*, pages 260–274, 2001.

[22] J. K. Gibson.

[23] O. Goldreich and Y. Lindell. "Session-key generation using human passwords only". In *Proc. of CRYPTO 2001*, pages 408–432, 2001.

[24] S. Goldwasser and S. Micali. "Probabilistic encryption". *Journal of Computer and System Sciences*, pages 270–299, 1984.

[25] S. Goldwasser, S. Micali, and C. Rackoff. "The knowledge complexity of interactive proof systems". In *Proc. of STOC '85*, pages 291–304, 1985.

[26] Lov K. Grover. "a fast quantum mechanical algorithm for database search". pages 212–219, 1996.

[27] S. Halevi and H. Krawczyk. "Public-key cryptography and password protocols". In *Proc. of ACM Conference on Computer and Commuinication s Security*, 1998.

[28] C. Hall, I. Goldberg, and B. Schneier. "Reaction attacks against several public-key cryptosystems". In *Proc. of the 2nd International Conference on Information and Communications Security (ICICS'99), LNCS 1726*, pages 2–12, 1999.

[29] N. Haller. "The S/KEY one-time password system". *RFC 1760*, 1995.

[30] N. Haller, C. Metz, P. Nesser, and M. Straw. "A one-time password system". *RFC 2289*, 1998.

[31] J. Hoffstein, J. Pipher, and J. H. Silverman. "NTRU: Aring-based public key cryptosystem". In *Proc. of Algorithmic Number Theory Symposium (ANTS-3)*, pages 267–288, 1998.

[32] H. Ijima and T. Matsumoto. "A simple scheme for challenge–response type human identification (in Japanese)". In *Proc. of Symposium on Cryptography and Information Security (SCIS94-13C)*, 1994.

[33] J. Jonsson. "An OAEP variant with a tight security proof draft 1.0". In *IACR ePrint archive 2002/034: http://eprint.iacr.org/2002/034*, 2002.

[34] J. Katz, R. Ostrovsky, and M. Yung. "Efficient password-authenticated key exchange using human-memorable passwords". In *Proc. of EUROCRYPT 2001: LNCS 2045*, pages 475–494, 2001.

[35] S. Kent and R. Atkinson. "Security architecture for the internet protocol". *RFC 2401*, 1998.

[36] K. Kobara and H. Imai. "Challenge control on challenge-response type human identification". In *Proc. of Japan–Korea Joint Workshop on Information Security and Cryptology (JW-ISC) '95*, pages 5–14, 1995.

[37] K. Kobara and H. Imai. "On the properties of the security against peeping attacks on challenge-response type direct human identification scheme using uniform mapping (in Japanese)". *IEICE Trans.(A)*, J79-A(8):1352–1359, 8 1996.

[38] K. Kobara and H. Imai. "Countermeasure against reaction attacks (in Japanese)". In *The 2000 Symposium on Cryptography and Information Security : A12*, January 2000.

[39] K. Kobara and H. Imai. "OAEP++ – another simple bug fix in OAEP –". In *Rump Session at Asiacrypt 2000*, 2000.

[40] K. Kobara and H. Imai. "Semantically secure McEliece public-key cryptosystems –conversions for McEliece PKC–". In *Proc. of PKC '01, LNCS 1992*, pages 19–35. Springer–Verlag, 2001.

[41] N. Koblitz. "Elliptic curve cryptosystems". *Math. Comp.*, 48:203–209, 1987.

[42] H. Krawczyk, M. Bellare, and R. Canetti. "HMAC: Keyed-hashing for message authentication". *RFC 2104*, 1997.

[43] T. Kwon. "Authentication and key agreement via memorable password". In *Proc. of NDSS 2001 Symposium Conference*, 2001.

[44] P. J. Lee and E. F. Brickell. "An observation on the security of McEliece's public-key cryptosystem". In *Proc. of EUROCRYPT '88, LNCS 330*, pages 275–280. Springer–Verlag, 1988.

[45] A. Lenstra and E. Verheul. "The XTR public key system". In *Proc. of CRYPTO 2000, LNCS 1880*, pages 1–19, 2000.

[46] P. Loidreau and N. Sendrier. "Some weak keys in McEliece public-key cryptosystem". In *Proc. of IEEE International Symposium on Information Theory, ISIT '98*, page 382, 1998.

[47] P. MacKenzie. "More efficient password-authenticated key exchange". In *Proc. of Topics in Cryptology – CT-RSA 2001 : LNCS 2020*, pages 361–377, 2001.

[48] P. MacKenzie. "On the security of the SPEKE password-authenticated key exchange protocol". In *IACR ePrint archive, http://eprint.iacr.org/2001/057/*, 2001.

[49] P. MacKenzie, S. Patel, and R. Swaminathan. "Password-authenticated key exchange based on RSA". In *Proc. of ASIACRYPT 2000*, pages 599–613. Springer–Verlag, 2000.

[50] T. Matsumoto. "An interactive human identification scheme with exact resistance to question-and-answer observing attacks –preliminary announcement – (in Japanese)". *Technical Report of IEICE*, ISEC94–39:15–20, December 1994.

[51] T. Matsumoto. "Interactive human identification schemes with exact resistance to question-answer attacks and their applications to human-computer cryptography (in Japanese)". *Technical Report of IEICE*, ISEC94–55:33–43, March 1995.

[52] T. Matsumoto. "Gummy and conductive silicone rubber fingers: Importance of vulnerability analysis". In *Proc. of ASIACRYPT 2002 : LNCS 2501*, pages 574–575. Springer-Verlag, 2002.

[53] T. Matsumoto and H. Imai. "Human identification through insecure channel". In *Proc. of EUROCRYPT '91, LNCS 547*, pages 409–421. Springer–Verlag, 1991.

[54] T. Matsumoto and R. Mizutani. "On interactive human identification schemes". In *Proc. of Japan–Korea Joint Workshop on Information Security and Cryptology (JW-ISC) '95*, pages 1–4, 1995.

[55] R. J. McEliece. "A public-key cryptosystem based on algebraic coding theory". In *Deep Space Network Progress Report*, 1978.

[56] A. J. Menezes, P. C. Oorschot, and S. A. Vanstone. "Mceliece public-key encryption". In *"Handbook of Applied Cryptography"*, page 299. CRC Press, 1997.

[57] A. J. Menezes, P. C. Oorschot, and S. A. Vanstone. "Simultaneous multiple exponentiation". In *"Handbook of Applied Cryptography"*, pages 617–619. CRC Press, 1997.

[58] V. S. Miller. "Use of elliptic curves in cryptography". In *Proc. of CRYPTO'85*, pages 417–426. Springer–Verlag, 1986.

[59] M. Naor and B. Pinkas. "Oblivious transfer and polynomial evaluation". In *Proc. 30th ACM Symp. on Theory of Computing*, pages 245–254, 1999.

[60] M. Naor and A. Shamir. "Visual cryptograpy". In *Proc. of EUROCRYPT '94, LNCS 950*, pages 1–12. Springer–Verlag, 1994.

[61] M. Naor and M. Yung. "Universal one-way hash functions and their cryptographic applications". In *Proc. of STOC '98*, pages 33–43, 1998.

[62] H. Niederreiter. “Knapsack-type cryptosystem based on algebraic coding theory”. *Problems of Control and Information Theory*, 15(2):157–166, 1986.

[63] T. Okamoto and D. Pointcheval. “REACT: Rapid enhanced-security asymmetric cryptosystem transform”. In *Proc. of RSA Conference '01*, 2001.

[64] T. Okamoto and S. Uchiyama. “A new public key cryptosystem as secure as factoring”. In *Proc. of EUROCRYPT '98, LNCS 1403*, pages 129–146, 1999.

[65] P. Paillier. “Public-key cryptosystems based on discrete logarithms residues”. In *Proc. of EUROCRYPT '99, LNCS 1592*, pages 223–238. Springer–Verlag, 1999.

[66] D. Pointcheval. “Chosen-ciphertext security for any one-way cryptosystem”. In *Proc. of PKC 2000, LNCS 1751*, pages 129–146. Springer–Verlag, 2000.

[67] M.O. Rabin. “Digitalized signatures and public-key functions as intractable as factorization”. In *MIT Laboratory for Computer Science Technical Report*, volume MIT/LCS/TR-212, 1979.

[68] R. Rivest, A. Shamir, and L. Adleman. “A method for obtaining digital signature and public-key cryptosystems”. *IEEE Trans. IT*, 22(6):644–654, 1976.

[69] A. Shamir. “How to share a secret”. *Communications of the ACM*, 22(11):612–613, 1979.

[70] A. Shamir. “An efficient identification scheme based on permuted kernels”. In *Proc. of CRYPTO '89, LNCS 435*, pages 606–609. Springer–Verlag, 1990.

[71] P.W. Shor. “Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer”. *SIAM Journal on Computing*, 26(5):1484–1509, 1997.

[72] V. Shoup. “OAEP reconsidered”. In *Proc. of CRYPTO 2001*, pages 239–259, 2001.

[73] W. Simpson. “PPP challenge handshake authentication protocol (CHAP)”. *RFC 1994*, 1996.

[74] J. Stern. “A method for finding codewords of small weight”. In *Proc. of Coding Theory and Applications , LNCS 388*, pages 106–113. Springer–Verlag, 1989.

[75] J. Stern. “A new identification scheme based on syndrome decoding”. In *Proc. of CRYPTO '93, LNCS 773*, pages 13–21. Springer–Verlag, 1994.

[76] J. Stern. “Designing identification scheme with keys of short size”. In *Proc. of CRYPTO '94, LNCS 839*, pages 164–173. Springer–Verlag, 1994.

[77] H. M. Sun. “Further cryptanalysis of the McEliece public-key cryptosystem”. *IEEE Trans. on communication letters*, 4(1):18–19, 2000.

[78] Wen-Guey Tzeng. “Efficient 1-out-n oblivious transfer schemes”. In *Proc. of PKC 2002, LNCS 2274*, pages 159–171. Springer–Verlag, 2002.

[79] A. Vardy. “The intractability of computing the minimum distance of a code”. *IEEE Trans. on IT*, 43(6):1757–1766, 1997.

[80] T. Ylonen. “SSH protocol architecture”. *I-D draft-ietf-architecture-11.txt*, 2001.

Published Works

- **Regular Papers**

  1. K. Kobara and H. Imai. "Security against peeping attacks of challenge-response type direct human identification scheme using uniform mapping" *Electronics and Communications in Japan (Part III)*, 82(1):79–86, January 1999. Japanese version appeared in *IEICE Trans.*, J79-A(8):1352–1359, 8 1996.
  2. K. Kobara and H. Imai. "Limiting the visible space visual secret sharing schemes and their application to human identification". In *Advances in Cryptology - ASIACRYPT'96, LNCS 1163*, pages 185–195. Springer–Verlag, 1996.
  3. K. Kobara and H. Imai. "The capacity of narrow-band subliminal channels in the case of successive carrier transmission". *Electronics and Communications in Japan (Part III)*, 84(1):1–10, January 2001. Japanese version appeared in *IEICE Trans.*, J82-A(10):1585–1592, 10 1999.
  4. K. Kobara and H. Imai. "An error-control method for narrow-band subliminal channels: How to embed more bits in a carrier ". *Electronics and Communications in Japan (Part III)*, 84(8):1–10, August 2001. Japanese version appeared in *IEICE Trans.*, J83-A(9):1089–1098, 9 2000.
  5. K. Kobara and H. Imai. "Semantically secure McEliece public-key cryptosystem". *IEICE Trans.*, E85-A(1):74–83, January 2002.
  6. K. Kobara and H. Imai. "New chosen-plaintext attacks on the one-wayness of the modified McEliece PKC proposed at Asiacrypt 2000". In *Proc. of PKC '02, LNCS 2274*, pages 237–251. Springer–Verlag, 2002.
  7. K. Kobara and H. Imai. "Pretty-simple password-authenticated key-exchange protocol proven to be secure in the standard model". *IEICE Trans.*, E85-A(10):2229–2237, October 2002.
  8. K. Kobara and H. Imai. "On the chosen-ciphertext security of enhanced Niederreiter public-key cryptosystems". *IEICE Trans. under submission.*

- **International Conferences**

1. K. Kobara and H. Imai. "Challenge control on challenge-response type human identification". In *Proc. of Japan–Korea Joint Workshop on Information Security and Cryptology (JW-ISC) '95*, pages 5–14, 1995.
2. K. Kobara and H. Imai. "Limiting the visible space applications in visual secret sharing". In *Proc. of 1996 International Symposium on Information Theory and Its Applications*, pages 71–74, 1996.
3. K. Kobara and H. Imai. "The capacity of a channel with a one-way function". In *Proc. of Japan–Korea Joint Workshop on Information Security and Cryptology (JW-ISC) '97*, pages 173–179, 1997.
4. K. Kobara and H. Imai. "Self-synchronized message randomization methods for subliminal channels". In *Proc. of International Conference on Information and Communications Security (ICICS'97) : LNCS 1334*, pages 325–334. Springer–Verlag, 1997.
5. K. Kobara and H. Imai. "How to increase information transmission rate of narrowband subliminal channels". In *Proc. of 1998 International Symposium on Information Theory and Its Applications*, pages 138–141, 1998.
6. K. Kobara and H. Imai. "A successive carrier-transmission model for narrowband subliminal channels". In *The 1st International Conference on Information Security and Cryptology*, pages 179–178. KIISC, December 1998.
7. K. Kobara and H. Imai. "On the channel capacity of narrow-band subliminal channels". In *Proc. of the 2nd International Conference on Information and Communications Security (ICICS'99) : LNCS 1726*, pages 309–323, 1999.
8. K. Kobara and H. Imai. "Countermeasures against all the known attacks to the McEliece PKC". In *Proc. of 2000 International Symposium on Information Theory and Its Applications*, pages 661–664, November 2000.
9. K. Kobara and H. Imai. "Copyright protection techniques for the future network society". In *Proc. of Workshop on Information Security Applications (WISA 2000)*, pages 51–52, 2000.
10. K. Kobara and H. Imai. "Copyright protection techniques for the future network society". In *Proc. of Workshop for Protection of the Multimedia Contents (WPMC 2000)*, pages 69–87, 2000.
11. K. Kobara and H. Imai. "Semantically secure McEliece public-key cryptosystems –conversions for McEliece PKC–". In *Proc. of PKC '01, LNCS 1992*, pages 19–35. Springer–Verlag, 2001.

12. K. Kobara and H. Imai. "OAEP++ – another very simple way to fix the bug in OAEP –". In *Proc. of 2002 International Symposium on Information Theory and Its Applications: S6-4-5*, pages 563–566, 2002.
13. K. Kobara, T. Inoue, O. Yoshikawa, M. Yasukawa, Y. Ishida, and H. Imai. "Secure and perfectly distributed data backup system". In *Proc. of Hong Kong International Computer Conference 2002 (HKICC2002)*, 2002.
14. K. Kobara and H. Imai. "PAKE vs. password-based authentications in wireless standards". In *Proc. of The 3rd International Workshop on ITS Telecommunications*, pages 135–138, 2002.

- **Awards**

  1. The SCIS'96 Paper Award for Young Researchers, 1996.1
  2. The ISITA2002 Paper Award for Young Researchers, 2002.10