Anonymity and Privacy in Public Key Cryptography

A DISSERTATION
SUBMITTED TO THE FACULTY OF THE GRADUATE SCHOOL
OF THE UNIVERSITY OF MINNESOTA
BY

Vishal Saraswat

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF
Doctor of Philosophy

Andrew Odlyzko

May, 2012

Dedication

Dedicated to my teachers, especially the first and the most influential teacher – my mother, Savitri Saraswat.

Acknowledgements

I owe my gratitude to many people who have contributed towards the completion of this thesis with their encouragement and support.

I am truly indebted to my advisor, Professor Andrew Odlyzko, for his patient guidance. I am thankful to Professors Scot Adams, Nicholas Hopper and Fadil Santosa for their participation in my dissertation committee. I am grateful to Dr. Giovanni Di Crescenzo with whom I co-authored the paper [17] on which Chapter 2 is based and to Dr. Aaram Yun with whom I co-authored the paper [28] on which Chapter 3 is based.

I thank Professor Yongdae Kim who introduced me to Cryptography, and Professors David Frank and Paul Garrett who guided me through my initial period at the University of Minnesota. I am thankful to Professors M.S. Raghunathan and Ravi Rao at the Tata Institute of Fundamental Research, Mumbai, and Professors Kajal Baksi, Tarun Kumar Bandyopadhyay and Partha Pratim Ghosh at St. Xavier's College, Kolkata, for their counseling and inspiration. I am grateful to my teachers Bobby Joseph, Kunwar Rai, Raj Narayan Singh and Rowland John Torcato whose mentoring had profound impact on my mathematical aptitude.

Special thanks to my friends, especially Akash Anand, Gouri Shankar Banerjee, Paromita Chakraborty, Deepshikha Jaiswal, Kriti Mehra, Balapa Padmapriya, Rekha Santhanam, Gopi Saraswat and Madhumita Sen, for their encouragement and support.

Each member of my family has made many sacrifices to help me achieve my goals. I am sincerely grateful to my grandparents Ram Niranjan and Ram Devi, my parents Subodh Kumar and Savitri Devi, my elder uncle Gouri Shankar and his wife Santosh, my younger uncle Bhavani Shankar and his wife Panna, my aunts Swati and Sulochana, my sisters Princy, Radhika and Riddhi, and last but not the least, my brother Vikash, his wife Khushboo and their daughter Devianshi.

Finally, I thank my beloved wife, Sahana Subbarao, for her sincere support and patience, and our son, Himanshu Dhruv Saraswat, for bringing so much joy and happiness into our lives.

Abstract

The Internet has become an essential part of our daily lives. However, various agents collect information about us giving rise to a plethora of privacy concerns. This thesis aims to guard the privacy of users by providing them anonymity in their online communication.

In the first part of this thesis, Chapter 2 based on our paper [17], we work on anonymity in identity based public-key cryptosystems (IBE). We provide the first ever instance and to date the only known non-trivial instance of anonymizing a non-anonymous IBE. We do so by constructing a public key encryption with keyword search (PEKS) which is equivalent to anonymous identity-based encryption (aIBE) as proved in [1]. Our PEKS is the first ever scheme not based on bilinear forms, and is based on the hardness of the quadratic residuosity problem. We construct our PEKS scheme using a non-trivial transformation of the Cocks IBE scheme. PEKS in itself is a very useful tool. It enables delegation of searching capabilities on encrypted data to a third party, who does not hold the entire secret key, but only an appropriate token which allows searching operations while preserving data privacy.

In the second part of this thesis, Chapter 3 based on our paper [28], we work on anonymity in digital signature schemes. We provide means of providing anonymity to the signer of a message by generically transforming any given digital signature scheme to an anonymous signature scheme. We present a new formalism of anonymous signature, where instead of the message as in previously presented schemes [33, 18], a part of the signature is withheld to maintain anonymity. We introduce the notion of unpretendability to guarantee infeasibility for someone other than the correct signer to pretend authorship of the message and signature. Our definition retains applicability for all previous applications of the anonymous signature, provides stronger security, and is conceptually simpler.

Keywords: Public-Key Encryption, Searchable Public-Key Encryption, Anonymity, Privacy, Quadratic Residuosity, Jacobi Symbol, Digital Signature, Anonymous Signature, Commitment Scheme, Unpretendability

Contents

Chapter 1

Introduction

The Internet has become an essential part of our daily lives. However, various agents collect information about us giving rise to a plethora of privacy concerns. This thesis aims to guard the privacy of users by providing them anonymity in their online communication. This thesis is divided into two parts: in Chapter 2, which is based on the joint work [17] with Dr. Giovanni Di Crescenzo, we work on public-key encryptions with keyword search which imply anonymous identity based public-key cryptosystems, and in Chapter 3, which based on the joint work [28] with Dr. Aaram Yun, we work on anonymity in digital signature schemes.

Public-key encryption is an essential tool for keeping online communications safe and secure. Anonymous encryption is well-known to be an attractive solution to the problem of fully-private communication in which it is required that the ciphertexts are sender-anonymous and receiver-anonymous. Anonymous encryption provides protection against traffic analysis by using a bulletin board setup. When a message meant for a specific member of a set of users is broadcast on the board, only the intended recipient is able to decipher the message while an adversary is unable to determine the identity of the intended recipient.

Anonymous encryption is necessary in anonymous credential systems [14]. Such a system enables users to control the dissemination of information about themselves by making it infeasible to correlate transactions carried out by the same user. Authenticated key exchange protocols such as SKEME [23] use anonymous encryption to protect the identity of roaming users from eavesdroppers. In SKEME, roaming users communicate with a base station for authentication and distribution of a session key based on the parties' public keys. Another application of anonymous encryption is for bid secrecy and verifiability in auction protocols [27]. The approach of [27] is to express each bid as an encryption of a known message, with the key to encrypt it corresponding to the value of the bid. Thus, the encryption key needs to be hidden. While in the previous two applications, the key-privacy property was needed to protect identities, in the last application, anonymous encryption is exploited to satisfy a secrecy requirement.

In the first part of this thesis, Chapter 2, we work on anonymity in identity based public-key cryptosystems (IBE). We provide the first ever instance and to date the only known non-trivial instance of anonymizing a non-anonymous IBE. We do so by constructing a public key encryption with keyword search (PEKS) which is equivalent to anonymous identity-based encryption (aIBE) as proved in [1]. Our PEKS is the first ever scheme not based on bilinear forms, and is based on the hardness of the quadratic residuosity problem. We construct our PEKS scheme using a non-trivial transformation of the Cocks IBE scheme.

A classical research area in cryptography is that of designing candidates for cryptographic primitives under different intractability assumptions in order to guarantee that not all cryptographic primitives depend on the supposed hardness of a single computational problem. So far, all presented PEKS schemes were based on bilinear forms and finding a PEKS that is not based on bilinear forms has been an open problem since the notion of PEKS was first introduced in [9]. Therefore our construction also achieves this conventional desideratum about all cryptographic primitives.

PEKS in itself is a very useful tool. It enables delegation of searching capabilities on encrypted data to a third party, who does not hold the entire secret key, but only an appropriate token which allows searching operations while preserving data privacy. Suppose a user Alice requires that when her email server receives an email, it should take actions depending on specific keywords associated with the email. For example, all words on the subject line as well as the sender's email address could be used as keywords. For these actions to be taken, the gateway needs to be able to read these keywords. However, if Alice also requires that these keywords be hidden from an adversary then the sender Bob would need to encrypt both the contents of the email and the keywords. But then the email gateway too cannot see the keywords and hence cannot make routing decisions. With PEKS one can enable Alice to give the gateway the ability to test whether a certain keyword is in the email without allowing the gateway to learn anything else about the email. More generally, Alice should be able to specify a few keywords that the email gateway can search for, but learn nothing else about the incoming email.

In the second part of this thesis, Chapter 3, we work on anonymity in digital signature schemes. A digital signature is a digital code that can be attached to an electronic message (email, spreadsheet, text file, etc.) to uniquely identify the sender and ensure authenticity and integrity of the document. Like a written signature, a digital signature guarantees that the person who claims to have sent a message is the one who sent it. Moreover, a digital signature also guarantees the message received is the one that was sent and has not been altered in any way since that person created it.

An anonymous signature is a signature scheme where the signature of a message does not reveal the identity of the signer. As in the case of anonymous encryption, anonymous signatures are useful in key exchange protocols [12, 23], anonymous transaction systems [14, 21], auction systems [24], and anonymous paper reviewing [33].

The notion of anonymous signature was formalized in 2006 [33] even though the notion of anonymous encryption was defined much earlier in 2001 [4]. Since a digital signature is publicly verifiable, an adversary attacking the anonymity of a digital signature can simply verify the message-signature pair with respect to a candidate's public key to verify whether they signed the message or not. Therefore, as long as the adversary obtains both the message and the signature, it seems that anonymity is impossible.

Previous attempts to provide anonymity involved either hiding the message or adding a "hidden" randomness in the message. The process of ensuring "enough" randomness in the message made those schemes very expensive, sometimes too expensive. Moreover, this added randomness made those signature schemes useless in many of the intended applications. Finally, the previous formalism does not give a rigorous guarantee of infeasibility for someone other than the correct signer to come later and pretend that the signature is theirs.

We present a new formalism of anonymous signature, where instead of the message as in previously presented schemes, a part of the signature is withheld to maintain anonymity. We introduce the notion of *unpretendability* to guarantee infeasibility for someone other than the correct signer to pretend authorship of the message and signature. Our definition retains applicability for all previous applications of the anonymous signature, provides stronger security, and is conceptually simpler. Finally, we provide a generic algorithm to transform any given digital signature scheme to an anonymous signature scheme which retains all the properties of the original digital signature scheme.

Chapter 2

Public Key Encryption with Keyword Search based on Jacobi Symbols

2.1 Introduction

Public-key encryption schemes with searchable keywords are useful to delegate searching capabilities on encrypted data to a third party, who does not hold the entire secret key, but only an appropriate token which allows searching operations but preserves data privacy. Such notion was previously proved to imply identity-based public-key encryption [9] and to be equivalent to anonymous (or key-private) identity-based encryption [9, 1], which are useful for fully-private communication.

Motivation. PEKS allows a sender to compute an encrypted message, so that the receiver can allow a third party to search keywords in the encrypted message without (additional) loss of privacy of the content of the message. The following motivating example for PEKS is taken almost verbatim from [9]. Suppose, a user Alice wishes to read her email on a number of devices: desktop, laptop and pager. Alice's email gateway is supposed to route email to the appropriate device based on the keywords in the email. For example, when Bob sends email with the keyword "urgent", the email is routed to Alice's pager, and when Bob sends email with the keyword "lunch", the email is routed to Alice's desktop for reading later. One expects each email to contain a small number of keywords. For example, all words on the subject line as well as the sender's email address could be used as keywords. Now, suppose Bob sends encrypted email to Alice using Alice's public key in which both the contents of the email and the keywords are encrypted. In this case the email gateway cannot see the keywords and hence cannot make routing decisions. With PEKS one can enable Alice to give the gateway the ability to test whether "urgent" is a keyword in the email, but the gateway should learn nothing else about the email. More generally, Alice should be able to specify a few keywords that the email gateway can search for, but learn nothing else about the incoming emails.

Previous work. In its non-interactive variant, constructions for this primitive were shown to be at least as hard to obtain as constructions for identity-based encryption (as proved in [9]). Moreover, the existence of PEKS was proved to follow from the existence of "anonymous" or "key-private" identity-based encryption (this was noted in [9] and formally proved in [1]), namely, encryption where the identity of the recipient remains unknown. Anonymous encryption is well-known to be an attractive solution to the problem of fully-private communication which provides sender-anonymity, receiver-anonymity and protection against traffic analysis, see, for example, discussions in [3, 13]. It is a natural goal then to try to convert the existing identity-based public-key cryptosystems into their anonymous variant so that a PEKS is automatically obtained. In fact, the anonymity or key-privacy property for a public-key encryption scheme (whether it is identity-based or not), is in itself a property of independent interest, as already discussed in [3], where this property was defined and investigated for conventional (that is, not identity-based) public-key encryption schemes. So far, however, all published PEKS schemes have been transformations of identity-based cryptosystems based on bilinear forms. Even the authors of [9] noted the difficulty of coming up with other examples of PEKS schemes, by observing that the only identity-based cryptosystem not based on bilinear forms (namely, the Cocks identity-based encryption scheme [16]) does not seem to have a direct transformation into an anonymous variant and thus into a PEKS scheme. Further work on PEKS (for example, [32, 26, 20, 1, 13]) did not contribute towards this goal, but only studied schemes and variations based on bilinear forms.

The construction of a new identity-based encryption scheme based on quadratic residuosity [11] and having short ciphertexts was obtained very recently; after seeing our work [6]. This scheme is also anonymous, like the scheme proposed by us in [17], but is based on very different techniques. Although their scheme is quite elegant, encryption and decryption operations are estimated [5] to be significantly less efficient than in the Cocks scheme. Instead, when used as an anonymous identity-based encryption scheme, our scheme is only less efficient than the original (and not anonymous) the Cocks scheme by a small constant factor. Following up on our construction, a similar construction [2] was obtained and is more space-efficient than our scheme by a small constant factor.

Our results. In this chapter, we construct the first PEKS scheme which is not based on bilinear forms but is based on a new assumption that can be seen as a variant of the well-known hardness of deciding quadratic residues modulo a large composite integer. Our scheme is obtained as a non-trivial transformation of the Cocks scheme. By the known equivalence of PEKS scheme and anonymous identity-based encryption, our scheme immediately gives the first anonymous identity-based encryption scheme which is not based on bilinear forms, a problem left open in [9]. Our scheme essentially preserves the time efficiency of the (not anonymous) identity-based encryption of the Cocks scheme, which was claimed in the original paper [16] to be satisfactory in a hybrid encryption mode (that is, when used to encrypt first a short session key and then using this key to produce a symmetric encryption of a large message). We do note however that the decryption time of the Cocks scheme (and thus of our scheme too) is less efficient than the known schemes based on bilinear forms.

Organization of the chapter. In what follows, we start by reviewing in Section 2.2 the formal definitions related to the notion of interest in this chapter: public-key encryption with keyword search. In Section 2.4, we present our public-key cryptosystem with keyword search, and in Section 2.5, we prove its properties.

2.2 Definitions and Preliminaries

We recall the general notion of identity-based public-key cryptosystems (IBE) (as defined in [10, 16]) in Subsection 2.2.1. We recall the known notion and formal definition of PEKS (as defined in [9, 1]) in Subsection 2.2.2. Jacobi symbols, $\left(\frac{\cdot}{\cdot}\right)$, and their calculations will be crucial for our construction and we recall it in Subsection 2.2.3. Finally in Subsection 2.2.4, we recall the Cocks identity-based public-key cryptosystem [16]) on which our construction is based.

Notations and Conventions. We denote the set of positive integers by $\mathbb{Z}^+$. For any positive integer $n > 1$, $\mathbb{Z}_n$ denotes the set of integers modulo n and $\mathbb{Z}_n^*$ denotes the set of invertible elements in $\mathbb{Z}_n$. We use $s \leftarrow S$ to denote a random variable s that randomly chooses an element from the set S uniformly and independently. We denote by $z \leftarrow \mathcal{A}(x, y, \dots)$ the output z of an algorithm $\mathcal{A}$ with input $(x, y, \dots)$, If $\mathcal{A}$ is a randomized algorithm, we use the same notation to define a random variable z which is the output of $\mathcal{A}$. If S is a set, (string, vector, matrix respectively) then we use $|S|$ to denote its size (length, vector-length, determinant respectively). Also, we say that a function $f : \mathbb{Z}^+ \to [0, 1]$ is *negligible in n* if $f(n) < 1/P(n)$ for any polynomial P and sufficiently large n.

We denote by m the security parameter, by k the length of the identities, and by l the length of the messages, where k and l are (independent) polynomials in m. We use $\mathcal{M}$ to denote the message space and $\mathcal{I}$ to denote the identity space.

2.2.1 Identity-based Public-key Cryptosystem

Definition 2.2.1. An *identity-based public-key cryptosystem* (IBE) can be defined as a 4-tuple of polynomial time algorithms $\mathsf{IBE} = (\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Encrypt}, \mathsf{Decrypt})$ with the following semantics:

Setup is used by the trusted authority TA to generate public parameters PK and a master secret key SK;

KeyGen is used by the trusted authority TA to generate a private key t_{id} given a party's id;

Encrypt is used by a sender who wants to encrypt a message to a receiving party and only uses the receiver's id and the public parameters PK;

Decrypt is used by a receiver to decrypt a ciphertext and only uses the private key t_{id} and the public parameters PK.

Some Identity Based Encryption Schemes of Interest

The Boneh-Franklin identity-based cryptosystem [10] denoted as

$$\mathsf{BF\text{-}IBE} = (\mathsf{BF\text{-}Setup}, \mathsf{BF\text{-}KeyGen}, \mathsf{BF\text{-}Encrypt}, \mathsf{BF\text{-}Decrypt})\,.$$

The Boneh-Boyen identity-based cryptosystem [7] denoted as

$$\mathsf{BB\text{-}IBE} = (\mathsf{BB\text{-}Setup}, \mathsf{BB\text{-}KeyGen}, \mathsf{BB\text{-}Encrypt}, \mathsf{BB\text{-}Decrypt})\,.$$

The Waters identity-based cryptosystem [31] denoted as

$$\mathsf{W\text{-}IBE} = (\mathsf{W\text{-}Setup}, \mathsf{W\text{-}KeyGen}, \mathsf{W\text{-}Encrypt}, \mathsf{W\text{-}Decrypt})\,.$$

The Cocks identity-based cryptosystem [16] denoted as

$$\mathsf{CC\text{-}IBE} = (\mathsf{CC\text{-}Setup}, \mathsf{CC\text{-}KeyGen}, \mathsf{CC\text{-}Encrypt}, \mathsf{CC\text{-}Decrypt})\,.$$

The DiCrescenzo-Saraswat anonymized Cocks identity-based cryptosystem [17] denoted as

$$\mathsf{CS\text{-}IBE} = (\mathsf{CS\text{-}Setup}, \mathsf{CS\text{-}KeyGen}, \mathsf{CS\text{-}Encrypt}, \mathsf{CS\text{-}Decrypt})\,.$$

Security Definitions

Consistency. Informally, for an identity-based encryption scheme to be consistent we require that the decryption of a ciphertext C must yield the same message M for which the encryption algorithm had computed the ciphertext.

Definition 2.2.2. We say that an IBE is (computationally) *consistent* if for all security parameters $m \in \mathbb{Z}^+$, all identities $id \in \mathcal{I}$ and all messages $M \in \mathcal{M}$, we have $Pr[\mathsf{Decrypt}(\mathsf{KeyGen}(SK, id), \mathsf{Encrypt}(PK, id, M)) \neq M]$ is negligible in m, where the probability is taken over the choice of $(PK, SK) \leftarrow \mathsf{Setup}(1^m)$.

Adversary-based Consistency. For *adversary-based consistency*, one would like to ensure that a ciphertext $C = \mathsf{Encrypt}(PK, id, M)$ should not be decryptable using the private key of an identity $id' \neq id$ to any valid message $M' \in \mathcal{M}$. Informally, for $M \leftarrow \mathcal{M}$, even an adversary that has access to the public parameters PK and is able to obtain private keys t_{id_i} for polynomial number of identities id_i of its choice should not be able to produce two different identities $id_0 \neq id_1$ such that $\mathsf{Decrypt}(\mathsf{KeyGen}(SK, id_1), \mathsf{Encrypt}(PK, id_0, M))$ does not return failure, $\perp$. Formally, we define *consistency against an active attacker* using the following game between a challenger $\mathcal{C}$ and an attacker $\mathcal{A}$.

IBE **Adversary-Based Consistency Game.**

1. The challenger runs the $\mathsf{Setup}(1^m)$ algorithm to generate PK and SK. It gives PK to the attacker.
2. A message $M \leftarrow \mathcal{M}$ is uniformly chosen.
3. The attacker returns two identities $id_0, id_1 \in \mathcal{I}$.
4. Encryption $C = \mathsf{Encrypt}(PK, id_0, M)$ and private key $t_{id_1} = \mathsf{KeyGen}(SK, id_1)$ are computed.
5. The attacker wins the game if $id_0 \neq id_1$ and $\mathsf{Decrypt}(t_{id_1}, C) \neq \perp$.

Definition 2.2.3. We say that an identity-based encryption scheme IBE satisfies (computational) *adversary-based consistency* if for any attacker $\mathcal{A}$ running in time polynomial in m, $\mathcal{A}$'s advantage (in breaking the consistency of the identity-based encryption scheme IBE)

$$Pr\left[\mathsf{Decrypt}(\mathsf{KeyGen}(SK, id_1), \mathsf{Encrypt}(PK, id_0, M)) \neq \perp\right]$$

is negligible in m.

Security. For an identity-based encryption scheme to be (semantically) *secure*, a ciphertext $\mathsf{Encrypt}(PK, id, M)$ must not reveal any information about M unless the private key t_{id} is available. Informally, even an attacker that has access to the public parameters PK and is able to obtain private keys t_{id_i} for polynomial number of identities id_i of its choice should not be able to distinguish between encryptions of messages

M_0 and M_1 of its choice for an identity id of its choice for which he did not obtain the private key. Formally, we define the security of an identity-based encryption, IBE, using the following game between a challenger $\mathcal{C}$ and an attacker $\mathcal{A}$.

IBE Security Game.

1. The challenger runs the $\mathsf{Setup}(1^m)$ algorithm to generate PK and MK. It gives PK to the attacker.
2. The attacker can adaptively ask the challenger for the private key t_{id} for any identity $id \in \mathcal{I}$ of its choice. Additionally, the attacker can also query for decryptions of ciphertexts (id, C) with respect to any identity id.
3. At some point, the attacker $\mathcal{A}$ sends the challenger two messages M_0, M_1 and an identity id^* on which it wishes to be challenged. The only restriction is that the attacker did not previously ask for the private key t_{id^*}. The challenger picks a random bit $b \leftarrow \{0,1\}$ and gives the attacker the challenge ciphertext $C^* = \mathsf{Encrypt}(PK, id^*, M_b)$.
4. The attacker can continue to query for private keys t_{id} for any identity id of its choice as long as $id \neq id^*$ and for queries for decryptions of ciphertexts (id, C) with respect to any identity id as long as $(id, C) \neq (id^*, C^*)$.
5. Eventually, the attacker $\mathcal{A}$ outputs a bit $b' \in \{0,1\}$.

The attacker wins the game if its output differs significantly depending on whether he was given the challenge ciphertext corresponding to M_0 or M_1. This is formalized as follows.

Definition 2.2.4. We say that an identity-based encryption scheme IBE is (t, q, ϵ)-*secure* against an adaptive chosen ciphertext attack if for any attacker $\mathcal{A}$ running in time t and making at most q private key queries, $\mathcal{A}$'s advantage (in breaking the security of the identity-based encryption scheme IBE)

$$\mathrm{Adv}_{\mathcal{A}}(m) = \Big| \mathrm{Prob}[\,\mathcal{A}^{ind-cca-0} = 1\,] - \mathrm{Prob}[\,\mathcal{A}^{ind-cca-1} = 1\,] \Big|$$

is at most ϵ where, for $b \in \{0,1\}$, $\mathcal{A}^{ind-cca-b} = 1$ denotes the event that $\mathcal{A}$ returns 1 given that $C = \mathsf{Encrypt}(PK, id^*, M_b)$.

Anonymity. An identity-based encryption scheme is *anonymous* if a ciphertext $\mathsf{Encrypt}(PK, id, M)$ does not reveal any information about the identity id of the recipient. Informally, even an attacker that has access to the public parameters PK and is able to obtain private keys t_{id_i} for polynomial number of identities id_i of its choice should not be able to distinguish between ciphertexts C_0 and C_1 obtained by encrypting a message M of its choice for identities id_0 and id_1 of its choice for which he did not obtain the private keys. Formally, we define the anonymity of an identity-based encryption, IBE, using the following game between a challenger $\mathcal{C}$ and an attacker $\mathcal{A}$.

IBE **Anonymity game.**

1. The challenger runs the $\mathsf{Setup}(1^m)$ algorithm to generate PK and MK. It gives PK to the attacker.
2. The attacker can adaptively ask the challenger for the private key t_{id} for any identity $id \in \mathcal{I}$ of its choice.
3. At some point, the attacker $\mathcal{A}$ sends the challenger two identities id_0, id_1 and a message m^* on which it wishes to be challenged. The only restriction is that the attacker must not have previously received the private keys t_{id_0} or t_{id_1}. The challenger picks a random bit $b \leftarrow \{0,1\}$ and gives the attacker the challenge ciphertext $C = \mathsf{Encrypt}(PK, id_b, M^*)$.
4. The attacker can continue to ask for private keys t_{id} for any identity id of its choice as long as $id \neq id_0, id_1$.
5. Eventually, the attacker $\mathcal{A}$ outputs a bit $b' \in \{0,1\}$.

The attacker wins the game if its output differs significantly depending on whether he was given the challenge ciphertext corresponding to id_0 or id_1. This is formalized as follows.

Definition 2.2.5. We say that an identity-based encryption scheme IBE is (t, q, ϵ)-*anonymous* against an adaptive chosen ciphertext attack if for any attacker $\mathcal{A}$ running in time t and making at most q private key queries, $\mathcal{A}$'s advantage (in breaking the anonymity of the identity-based encryption scheme IBE)

$$\mathrm{Adv}_{\mathcal{A}}(m) = \big|\, \mathrm{Prob}[\, \mathcal{A}^{ano-cca-0} = 1\,] - \mathrm{Prob}[\, \mathcal{A}^{ano-cca-1} = 1\,] \,\big|$$

is at most ϵ where, for $b \in \{0,1\}$, $\mathcal{A}^{ano-cca-b} = 1$ denotes the event that $\mathcal{A}$ returns 1 given that $C = \mathsf{Encrypt}(PK, id_b, M^*)$.

Definition 2.2.6. We say that an identity-based encryption scheme IBE is ϵ-*anonymous* against an adaptive chosen ciphertext attack if for all attackers $\mathcal{A}$ running in time polynomial in m and making at most polynomial in m private key queries, $\mathcal{A}$'s advantage (in breaking the anonymity of the identity-based encryption scheme IBE)

$$\mathrm{Adv}_{\mathcal{A}}(m) = \left| \,\mathrm{Prob}[\,\mathcal{A}^{ano-cca-0} = 1\,] - \mathrm{Prob}[\,\mathcal{A}^{ano-cca-1} = 1\,]\, \right|$$

is at most ϵ where, for $b \in \{0,1\}$, $\mathcal{A}^{ano-cca-b} = 1$ denotes the event that $\mathcal{A}$ returns 1 given that $C = \mathsf{Encrypt}(PK, id_b, M^*)$.

2.2.2 Public-Key Encryption with Keyword Search

Informally speaking, in a PEKS scheme a sender would like to send a message in encrypted form to a receiver so that the receiver can allow a third party to search keywords in the encrypted message without losing (any additional) privacy on the message's content. According to [9], a non-interactive implementation of this task can be performed as follows. The sender encrypts her message using a conventional public key cryptosystem, and then appends to the resulting ciphertext a *Public-key Encryption with Keyword Search* (PEKS) of each keyword. Specifically, to encrypt a message M with searchable keywords $W_1, \ldots, W_m$, the sender computes and sends to the receiver

$$E_{A_{pub}}(M) \;\|\; \mathsf{ksEnc}(A_{pub}, W_1) \;\|\; \cdots \;\|\; \mathsf{ksEnc}(A_{pub}, W_m), \tag{2.1}$$

where A_{pub} is the receiver's public key and E is the encryption algorithm of the conventional public-key cryptosystem. Based on this encryption, the receiver can give the third party a certain trapdoor T_W which enables the third party to test whether one of the keywords associated with the message is equal to the word W of the receiver's choice. Specifically, given $\mathsf{ksEnc}(A_{pub}, W')$ and T_W, the third party can test whether $W = W'$; if $W \neq W'$ the third party learns nothing more about W'. Note that sender and receiver do not communicate in this entire process as the sender generates the searchable ciphertext for W' given just the receiver's public key (hence, the term "*public-key* encryption with keyword search" is used here).

More formally, we consider a setting with three parties: a sender, a receiver, and a third party (representing the email gateway in the application example given in the introduction). In this setting, a public-key encryption with keyword search is defined as follows.

Definition 2.2.7. A (non-interactive) public-key encryption scheme with keyword search (PEKS) consists of the following polynomial time randomized algorithms:

1. KeyGen(1^m): on input security parameter 1^m in unary, it returns a pair (A_{pub}, A_{priv}) of public and private keys.

2. ksEnc(A_{pub}, W): on input a public key A_{pub} and a keyword W, it returns a ciphertext, also called the *searchable encryption* of W.

3. Trapdoor(A_{priv}, W): on input Alice's private key and a keyword W, it returns a trapdoor T_W.

4. Test(A_{pub}, S, T_W): on input Alice's public key, a searchable encryption $S =$ ksEnc(A_{pub}, W'), and a trapdoor $T_W =$ Trapdoor(A_{priv}, W), it returns 'yes' or 'no'.

Given the above definition, an execution of a PEKS scheme goes as follows. First, the receiver runs the KeyGen algorithm to generate her public/private key pair. Then, she uses the Trapdoor algorithm to generate trapdoors T_W for any keywords W which she wants the third party to search for. The third party uses the given trapdoors as input to the Test algorithm to determine whether a given message encrypted by any sender using algorithm ksEnc contains one of the keywords W specified by the receiver.

We now define the main properties which a PEKS scheme must satisfy: (two variants of) *consistency* and *security*. The following basic definition will be useful: we say that a given function $f : N \rightarrow [0, 1]$ is *negligible in n* if $f(n) < 1/p(n)$ for any polynomial p and sufficiently large n.

Consistency. Next, we consider definitions of consistency for a PEKS (following definitions in [9, 1]). We consider two variants: right-keyword consistency and adversary-based consistency for a PEKS in the random oracle model.

Informally, in right-keyword consistency, we require the success of the search of any word W for which the encryption algorithm had computed a searchable encryption.

Definition 2.2.8. We say that a PEKS is *right-keyword consistent* if it holds that for any word W, the probability that $\mathsf{Test}(A_{pub}, C, T_W) \neq$ 'yes' is negligible in m, where A_{pub} was generated using the KeyGen algorithm and input 1^m, C was computed as $\mathsf{ksEnc}(A_{pub}, W)$ and T_W was computed as $\mathsf{Trapdoor}(A_{priv}, W)$.

Informally, in adversary-based consistency, one would like to ensure that even an adversary that has access to the public parameters PK and to a (uniformly distributed) random oracle cannot come up with two different keywords such that the testing algorithm returns 'yes' on input a trapdoor for one word and a public-key encryption with keyword search of the other. Formally, we define consistency against an attacker $\mathcal{A}$ using the following game between a challenger and an attacker. Here, we denote by m the security parameter, given in unary as input to both players, and by k the length of the keywords, where we assume that $k = \Theta(m^c)$, for some constant $c > 0$ (this assumption is seen to be wlog using simple padding).

PEKS **Adversary-Based Consistency game:**

1. The challenger runs the $\mathsf{KeyGen}(1^m)$ algorithm to generate A_{pub} and A_{priv}. It gives A_{pub} to the attacker.
2. The attacker returns two keywords $W_0, W_1 \in \{0,1\}^k$.
3. Encryption $C = \mathsf{ksEnc}(A_{pub}, W_0)$ and trapdoor $T_{W_1} = \mathsf{Trapdoor}(A_{priv}, W_1)$ are computed.
4. The attacker wins the game if $W_0 \neq W_1$ and $\mathsf{Test}(A_{pub}, C, T_{W_1})$ returns 'yes'.

We define $\mathcal{A}$'s advantage $\mathrm{Adv}_{\mathcal{A}}(m, k)$ in breaking the consistency of PEKS as the probability that the attacker wins the above game.

Definition 2.2.9. We say that a PEKS satisfies (computational) *adversary-based consistency* if for any attacker $\mathcal{A}$ running in time polynomial in m, we have that the function $\mathrm{Adv}_{\mathcal{A}}(m)$ is negligible in m.

Security. Finally, we recall the definition of security for a PEKS (in the sense of semantic-security). Here, one would like to ensure that an $\mathsf{ksEnc}(A_{pub}, W)$ does not reveal any information about W unless T_W is available. This is done by considering an attacker who is able to obtain trapdoors T_W for any W of his choice, and require that, even under such attack, the attacker should not be able to distinguish an encryption of a keyword W_0 from an encryption of a keyword W_1 for which he did not obtain the trapdoor. Formally, we define security against an active attacker $\mathcal{A}$ using the following game between a challenger and the attacker. Here, we denote by m the security parameter, given in unary as input to both players, and by k the length of the keywords, where we assume that $k = \Theta(m^c)$, for some constant $c > 0$ (this assumption is seen to be wlog using simple padding).

PEKS Security game:

1. The challenger runs the $\mathsf{KeyGen}(1^m)$ algorithm to generate A_{pub} and A_{priv}. It gives A_{pub} to the attacker.
2. The attacker can adaptively ask the challenger for the trapdoor T_W for any keyword $W \in \{0,1\}^k$ of his choice.
3. At some point, the attacker $\mathcal{A}$ sends the challenger two keywords W_0, W_1 on which it wishes to be challenged. The only restriction is that the attacker did not previously ask for the trapdoors T_{W_0} or T_{W_1}. The challenger picks a random $b \in \{0,1\}$ and gives the attacker $C = \mathsf{ksEnc}(A_{pub}, W_b)$. We refer to C as the challenge ciphertext.
4. The attacker can continue to ask for trapdoors T_W for any keyword W of his choice as long as $W \neq W_0, W_1$.
5. Eventually, the attacker $\mathcal{A}$ outputs $d \in \{0,1\}$.

Here, the attacker wins the game if its output differs significantly depending on whether he was given the challenge ciphertext corresponding to W_0 or W_1. This is formalized as follows. First, for $b \in \{0,1\}$, let $\mathcal{A}^b = 1$ denote the event that $\mathcal{A}$ returns 1 given that $C = \mathsf{ksEnc}(A_{pub}, W_b)$. Then, define $\mathcal{A}$'s advantage in breaking the PEKS scheme as

$$\mathrm{Adv}_{\mathcal{A}}(m) = \left|\,\mathrm{Prob}[\,\mathcal{A}^0 = 1\,] - \mathrm{Prob}[\,\mathcal{A}^1 = 1\,]\,\right|$$

Definition 2.2.10. We say that a PEKS is semantically secure against an adaptive chosen-keyword attack if for any attacker $\mathcal{A}$ running in time polynomial in m, we have that the function $\mathrm{Adv}_{\mathcal{A}}(m)$ is negligible in m.

Remark 2.2.11. We defined right-keyword consistency as done in [9, 1] (although the name was first used in [1]). The (computational version of the) adversary-based consistency was defined as a relaxed version of what is called just consistency in [1]; the relaxation consisting in only restricting the adversary to return keywords which have a known upper-bounded length. Although guaranteeing a slightly weaker property, this is essentially not a limitation in practical scenarios where a (small) upper bound on the length of keywords is known to all parties. We also note that such a relaxation is always done, for instance, in the definition of conventional public-key cryptosystems.

2.2.3 Jacobi Symbols

Definitions

Definition 2.2.12 (Quadratic Residue)**.** For a positive integer $n > 1$, we say that $a \in \mathbb{Z}_n^*$ is a *quadratic residue* of n if there exists $x \in \mathbb{Z}_n$ such that

$$x^2 \equiv a \mod n\,.$$

Otherwise, a is called a *quadratic nonresdue* of n.

Definition 2.2.13 (Legendre Symbol)**.** Let p be an odd prime and a be an integer. The *Legendre symbol* of a and p, denoted $\left(\frac{a}{p}\right)$, is defined as:

$$\left(\frac{a}{p}\right) = \begin{cases} 0 & \text{if } \gcd(a,p) \neq 1; \\ +1 & \text{if } a \text{ is a quadratic residue} \mod p; \\ -1 & \text{if } a \text{ is a non-quadratic residue} \mod p\,. \end{cases}$$

Definition 2.2.14 (Jacobi Symbol)**.** Let n be an odd positive integer with prime factorization ${p_1}^{e_1} \cdots {p_k}^{e_k}$. Let a be an integer. The *Jacobi symbol* $\left(\frac{a}{n}\right)$ of a and n is defined to be

$$\left(\frac{a}{n}\right) = \prod_{i=1}^{k} \left(\frac{a}{p_i}\right)^{e_i}$$

where $\left(\frac{a}{p_i}\right)$ is the Legendre symbol of a and p_i.

Properties of the Jacobi symbol

There are a number of useful properties of the Jacobi symbol which can be used to speed up calculations. They include:

1. If n is prime, the Jacobi symbol is the Legendre symbol.
2. $\left(\frac{a}{n}\right) \in \{0, 1, -1\}$.
3. $\left(\frac{a}{n}\right) = 0$ iff $\gcd(a, n) \neq 1$.
4. If $a \equiv b \mod n$, then $\left(\frac{a}{n}\right) = \left(\frac{b}{n}\right)$.
5. $\left(\frac{ab}{n}\right) = \left(\frac{a}{n}\right)\left(\frac{b}{n}\right)$; $\left(\frac{\cdot}{\cdot}\right)$ is a completely multiplicative function in its top argument.
6. $\left(\frac{a}{mn}\right) = \left(\frac{a}{m}\right)\left(\frac{a}{n}\right)$; note that this implies $\left(\frac{a}{n^2}\right)$ is 0 or 1 for any a and any n.
7. $\left(\frac{1}{n}\right) = 1$
8. For any odd integer n, $\left(\frac{-1}{n}\right) = (-1)^{(n-1)/2} = \begin{cases} +1 & \text{if } n \equiv 1 \mod 4; \\ -1 & \text{if } n \equiv 3 \mod 4\,. \end{cases}$
9. For any odd integer n, $\left(\frac{2}{n}\right) = (-1)^{(n^2-1)/8} = \begin{cases} +1 & \text{if } n \equiv 1 \text{ or } 7 \mod 8; \\ -1 & \text{if } n \equiv 3 \text{ or } 5 \mod 8\,. \end{cases}$
10. For odd integers m and n, $\left(\frac{m}{n}\right) = \left(\frac{n}{m}\right)(-1)^{(m-1)(n-1)/4}$.

The last property is known as reciprocity and is similar to the *law of quadratic reciprocity* for Legendre symbols:

Theorem 2.2.15 (Law of Quadratic Reciprocity)**.** Let p and q be two distinct odd primes. Then

$$\left(\frac{q}{p}\right)\left(\frac{p}{q}\right) = (-1)^{(p-1)(q-1)/4}$$

or, formulating differently,

$$\left(\frac{p}{q}\right) = \begin{cases} -\left(\frac{q}{p}\right) & \text{if both } p \text{ and } q \text{ are congruent to 3 modulo 4;} \\ +\left(\frac{q}{p}\right) & \text{if one of } p \text{ or } q \text{ is congruent to 1 modulo 4.} \end{cases}$$

There are two statements about quadratic residues with respect to the Legendre symbol which cannot be made with the Jacobi symbol.

First, $\left(\frac{a}{n}\right) = 1$ does not imply a is a quadratic residue of n. Since the Jacobi symbol $\left(\frac{a}{n}\right)$ is a product of Legendre symbols, if an even number of the Legendre symbols in the product evaluate to -1 then $\left(\frac{a}{n}\right) = 1$ but a is not a quadratic residue of n.

Second, there is no analogue of Euler's congruence

$$\left(\frac{a}{p}\right) \equiv a^{(p-1)/2} \mod p\,,\; p \text{ an odd prime,}$$

for Jacobi symbols. In fact, that congruence is false at least half the time for Jacobi symbols with a composite denominator. (This is the basis for the Solovay-Strassen probabilistic primality test.)

Calculating the Jacobi symbol

To calculate the Jacobi symbol $\left(\frac{a}{n}\right)$ for integers a and n, where n is an odd positive integer, first check if $\gcd(a, n) = 1$ (otherwise the Jacobi symbol is 0). Then choose $0 < b < n$ such that $a \equiv b \mod n$ so that $\left(\frac{a}{n}\right) = \left(\frac{b}{n}\right)$. Then write $b = 2^s c$ with odd c so that

$$\left(\frac{a}{n}\right) = \left(\frac{b}{n}\right) = \left(\frac{2}{n}\right)^s \left(\frac{c}{n}\right).$$

Now apply the quadratic reciprocity law to get

$$\left(\frac{c}{n}\right) = (-1)^{(c-1)(n-1)/4} \left(\frac{n}{c}\right)$$

so that

$$\left(\frac{a}{n}\right) = \left(\frac{b}{n}\right) = \left(\frac{2}{n}\right)^s \left(\frac{c}{n}\right) = \left(\frac{2}{n}\right)^s (-1)^{(c-1)(n-1)/4} \left(\frac{n}{c}\right).$$

The factor $\left(\frac{2}{n}\right)$ can be computed using the Property 9 of Jacobi symbols:

$$\left(\frac{2}{n}\right) = (-1)^{(n^2-1)/8} = \begin{cases} +1 & \text{if } n \equiv 1 \text{ or } 7 \mod 8; \\ -1 & \text{if } n \equiv 3 \text{ or } 5 \mod 8\,. \end{cases}$$

The factor $(-1)^{(c-1)(n-1)/4}$ is equal to -1 if $c, n \equiv 3 \mod 4$ and 1 otherwise.

At this point we can start the whole procedure over again for $\left(\frac{n}{c}\right)$. Eventually we can apply the equation $\left(\frac{-1}{n}\right) = (-1)^{(n-1)/2}$, which is equal to 1 if $n \equiv 1 \mod 4$ and -1 otherwise.

Example 2.2.1. We try to calculate $\left(\frac{107}{23}\right)$. Since $\gcd(107, 23) = 1$ and $107 \equiv 15 \mod 23$ we find

$$\left(\frac{107}{23}\right) = \left(\frac{15}{23}\right) .$$

Since both 15 and 23 are congruent 3 modulo 4 we have

$$\left(\frac{15}{23}\right) = -\left(\frac{23}{15}\right) = -\left(\frac{8}{15}\right) = -\left(\frac{2^3}{15}\right) = -\left(\frac{2}{15}\right) .$$

This can be evaluated using the Property 9 of Jacobi symbols and we find

$$\left(\frac{107}{23}\right) = -1 .$$

2.2.4 Cocks Identity-Based Public-Key Cryptosystem

We recall the construction of the Cocks identity-based public-key cryptosystem [16] which is based on Jacobi symbols and quadratic residues modulo a large composite integer. We denote it as $\mathsf{CC\text{-}IBE} = (\mathsf{CC\text{-}Setup}, \mathsf{CC\text{-}KeyGen}, \mathsf{CC\text{-}Encrypt}, \mathsf{CC\text{-}Decrypt})$:

$\mathsf{CC\text{-}Setup}$**:** Given a security parameter 1^m in unary, for $m \in \mathbb{Z}^+$, this algorithm (run by the TA) does the following:

1. randomly choose two primes p, q of length $m/2$, and such that both p and q are congruent to $3 \mod 4$ and set $n = pq$ (such integers n are also called Blum-Williams integers);
2. generate the description of a cryptographic hash function $H : \{0,1\}^* \to \mathbb{Z}_n^{+1}$ (which is assumed to behave as a random oracle in the analysis) where $\mathbb{Z}_n^{+1}$ is the set of positive integers x which are $< n$, coprime with n, and are such that the Jacobi symbol $\left(\frac{x}{n}\right) = +1$. (Note that the Jacobi symbol can be calculated without knowledge of the factorization of n; see, for example, Section 2.2.3.)

The public parameters of the system are $PK = (n, H)$ and the master secret key is $SK = (p, q)$.

CC-KeyGen**:** On input Alice's identity $ID \in \{0,1\}^*$, this algorithm computes $a = H(ID)$ and $r = a^{\frac{n+5-(p+q)}{8}} \mod n$, and returns r.

This algorithm is run by the TA after receiving Alice's identity, and the value r is meant to be received by Alice, who sets her secret key equal to $SK_A = r$. Note that by choice of p, q, the value r is a square root of either a or $-a$ depending on whether a is a square or not; if a is a square then $r^2 = a \mod n$; otherwise $-a$ must be a square and then $r^2 = -a \mod n$.

CC-Encrypt**:** If a user Bob wants to send a transport key K to Alice, he encrypts K using this algorithm on input Alice's identity ID, as follows:

1. Compute $a = H(ID)$ and encode K as values $(K_1, \ldots, K_m)$, where K_i is an encoding as $+1$ or -1 of the i-th bit k_i of the transport key K (e.g., $K_i = 2k_i - 1$).
2. For each $i = 1, \ldots, m$, randomly choose $t_i, t_{m+i} \in \mathbb{Z}_n^*$ such that $\left(\frac{t_i}{n}\right) = \left(\frac{t_{m+i}}{n}\right) = K_i$.
3. For $i = 1, \ldots, m$, compute $s_i = (t_i + a/t_i) \mod n$;
   for $i = m+1, \ldots, 2m$, compute $s_i = (t_i - a/t_i) \mod n$
4. Send $s = (s_1, \ldots, s_m, s_{m+1}, \ldots, s_{2m})$ to Alice.

CC-Decrypt**:** This algorithm is run by Alice that recovers the transport key K as follows:

1. Set $j = 0$ if $r^2 = a \mod n$ or $j = m$ if $r^2 = -a \mod n$.
2. For $i = 1, \ldots, m$, compute $\bar{K}_{j+i} = \left(\frac{s_{j+i}+2r}{n}\right)$ and the decoding $\bar{k}_{j+i} \in \{0,1\}$ of $\bar{K}_{j+i}$ (e.g., $\bar{k}_{j+i} = (\bar{K}_{j+i} + 1)/2$).
3. Return the transport key $(\bar{k}_{j+1}, \ldots, \bar{k}_{j+m})$.

Consistency of CC-IBE We now note that the above algorithms decrypt correctly. Specifically, if $r^2 = a \mod n$ then for $i = 1, \ldots, m$, it holds that

$$s_i + 2r = t_i + a/t_i + 2r = t_i \cdot (1 + r^2/t_i^2 + 2r/t_i) = t_i \cdot (1 + r/t_i)^2 \mod n\,.$$

Then it holds that

$$\bar{K}_i = \left(\frac{s_i + 2r}{n}\right) = \left(\frac{t_i}{n}\right) = K_i, \tag{2.2}$$

as long as as long as $s_i + 2r \mod n$ is in $\mathbb{Z}_n^*$. Note that there are only $p+q-1$ elements in $\mathbb{Z}_n \setminus \mathbb{Z}_n^*$ and thus the probability that $s_i + 2r \mod n$ is not in $\mathbb{Z}_n^*$ is $1/p + 1/q - 1/pq \leq$

$2^{-m/4+1}$, which is negligible in m, since p, q are chosen to be of length $m/2$.

An almost identical analysis applies in the case $r^2 = -a \mod n$, where for $i = m+1, \ldots, 2m$, it holds that

$$s_i + 2r = t_i - a/t_i + 2r = t_i \cdot (1 + r^2/t_i^2 - 2r/t_i) = t_i \cdot (1 - r/t_i)^2 \mod n.$$

and therefore, except with negligible probability, it holds that

$$\bar{K}_i = \left(\frac{s_i + 2r}{n}\right) = \left(\frac{t_i}{n}\right) = K_i,$$

so that $\bar{K} = K$.

With respect to the security property, from [16] it follows that this scheme is secure against a chosen-identity attack (in the random oracle model) under the assumption that it is hard to decide quadratic residuosity modulo Blum-Williams integers.

2.3 An Intractability Assumption

In this section, we define the Quadratic Indistinguishability Problem (QIP). Our cryptosystem is based on the intractability assumption of the QIP. We prove that the QIP is at least as hard as the Quadratic Residuousity Problem (QRP) by reducing the QRP to the QIP. Technically speaking, we might solve the following subproblem: We evaluate and discuss the statistical and computational difference between the two distributions below and prove that they are "identical".

2.3.1 Quadratic Indistinguishability Problem

Given a security parameter m, let $\mathrm{BW}(1^m)$ denote the set of m-bit primes p such that $p = 3 \bmod 4$. For a positive integer $n > 1$, let $\mathbb{Z}_n^*$ denote the set of positive integers which are less than and coprime to n. Let $QR(n)$ denote the set of quadratic residues modulo n and let $\mathbb{Z}_n^{+1}$ (resp. $\mathbb{Z}_n^{-1}$) be the elements of $\mathbb{Z}_n^*$ which have Jacobi symbol equal to $+1$ (resp. -1). Let $\mathrm{CS}(n, \alpha)$ denote the set of integers s in $\mathbb{Z}_n^*$ which satisfy the condition α.

Quadratic Residuosity Problem (QRP). The QRP *problem* consists of efficiently distinguishing the following two distributions:

$$\begin{aligned} E_0(1^m) &= \{p, q \leftarrow \mathrm{BW}(1^m); n \leftarrow p \cdot q; \\ &\qquad s \leftarrow \mathbb{Z}_n^{-1} \cup QR(n) \;:\; (n, s)\} \\ &= \{p, q \leftarrow \mathrm{BW}(1^m); n \leftarrow p \cdot q; \\ &\qquad s \leftarrow \mathrm{CS}(n, s \in \mathbb{Z}_n^{-1} \cup QR(n)) \;:\; (n, s)\} \\ E_1(1^m) &= \{p, q \leftarrow \mathrm{BW}(1^m); n \leftarrow p \cdot q; \\ &\qquad s \leftarrow \mathbb{Z}_n^* \;:\; (n, s)\} \\ &= \{p, q \leftarrow \mathrm{BW}(1^m); n \leftarrow p \cdot q; \\ &\qquad s \leftarrow \mathrm{CS}(n, s \in \mathbb{Z}_n^*) \;:\; (n, s)\}\,. \end{aligned}$$

We say that algorithm $\mathcal{A}$ has *advantage* ϵ in solving QRP if we have that:

$$\begin{aligned} \Big| \Pr[(n, s) \leftarrow E_0(1^m) : \mathcal{A}(n, s) = 1] \\ - \Pr[(n, s) \leftarrow E_1(1^m) : \mathcal{A}(n, s) = 1] \Big| = \epsilon. \end{aligned} \tag{2.3}$$

We say that QRP is *intractable* if all polynomial time (in m) algorithms have a negligible (in m) advantage in solving QRP.

Quadratic Indistinguishability Problem (QIP). The QIP *problem* consists of efficiently distinguishing the following two distributions:

$$\begin{aligned} D_0(1^m) &= \{p, q \leftarrow \mathrm{BW}(1^m); n \leftarrow p \cdot q; h \leftarrow \mathbb{Z}_n^{+1}; \\ &\qquad s \leftarrow \mathrm{CS}(n, s^2 - 4h \in \mathbb{Z}_n^{-1} \cup QR(n)) \;:\; (n, h, s)\} \\ D_1(1^m) &= \{p, q \leftarrow \mathrm{BW}(1^m); n \leftarrow p \cdot q; h \leftarrow \mathbb{Z}_n^{+1}; \\ &\qquad s \leftarrow \mathbb{Z}_n^* \;:\; (n, h, s)\} \\ &= \{p, q \leftarrow \mathrm{BW}(1^m); n \leftarrow p \cdot q; h \leftarrow \mathbb{Z}_n^{+1}; \\ &\qquad s \leftarrow \mathrm{CS}(n, s \in \mathbb{Z}_n^*) \;:\; (n, h, s)\}\,. \end{aligned}$$

We say that algorithm $\mathcal{A}$ has *advantage* ϵ in solving QIP if we have that:

$$\Big| \Pr[(n,h,s) \leftarrow D_0(1^m) : \mathcal{A}(n,h,s) = 1] \\ - \Pr[(n,h,s) \leftarrow D_1(1^m) : \mathcal{A}(n,h,s) = 1]\Big| = \epsilon. \quad (2.4)$$

We say that QIP is *intractable* if all polynomial time (in m) algorithms have a negligible (in m) advantage in solving QIP.

Before proving the equivalence of QRP and QIP, we prove the equivalence of QIP and QIP$_0$.

QIP$_0$ **Problem.** The QIP$_0$ *problem* consists of efficiently distinguishing the following two distributions:

$$\begin{aligned} D_{0,0}(1^m) = \{&p, q \leftarrow \mathrm{BW}(1^m); n \leftarrow p \cdot q; h \leftarrow \mathbb{Z}_n^{+1}; \\ &s \leftarrow \mathrm{CS}(n, s^2 - 4h \in \mathbb{Z}_n^{-1} \cup QR(n)) \; : \; (n,h,s)\} \\ D_{0,1}(1^m) = \{&p, q \leftarrow \mathrm{BW}(1^m); n \leftarrow p \cdot q; h \leftarrow \mathbb{Z}_n^{+1}; \\ &s \leftarrow \mathrm{CS}(n, s^2 - 4h \in \mathbb{Z}_n^{*}) \; : \; (n,h,s)\} \,. \end{aligned}$$

Recall that given a finite set C and any two subsets A and B, we have

$$|A \cup B| \leq |C| \quad \text{and} \quad |A \cap B| \geq |A| + |B| - |C|$$

so that

$$|A \Delta B| = |A \cup B| - |A \cap B| \leq |C| - (|A| + |B| - |C|) = 2|C| - |A| - |B| \,. \qquad (2.5)$$

Since addition by a non-zero element is a 1-1 map on $\mathbb{Z}_n$, we have

$$|4h + \mathbb{Z}_n^*| = |\mathbb{Z}_n^*| = (p-1)(q-1)$$

so that, by Equation 2.5,

$$|4h + \mathbb{Z}_n^* \Delta \mathbb{Z}_n^*| \leq 2pq - (p-1)(q-1) - (p-1)(q-1) = 2(p+q-1)\,.$$

Thus the advantage of an adversary distinguishing the two distributions

$$\begin{aligned} D_1'(1^m) &= \{p, q \leftarrow \mathrm{BW}(1^m); n \leftarrow p \cdot q; h \leftarrow \mathbb{Z}_n^{+1}; \\ &\qquad s \leftarrow \mathrm{CS}(n, s \in \mathbb{Z}_n^*) \,:\, (n, h, s)\} \\ D_{0,1}'(1^m) &= \{p, q \leftarrow \mathrm{BW}(1^m); n \leftarrow p \cdot q; h \leftarrow \mathbb{Z}_n^{+1}; \\ &\qquad s \leftarrow \mathrm{CS}(n, s \in 4h + \mathbb{Z}_n^*) \,:\, (n, h, s)\} \end{aligned} \tag{2.6}$$

is at most

$$\frac{2(p+q-1)}{(p-1)(q-1)}\,.$$

In particular, the advantage of an adversary distinguishing the two distributions

$$\begin{aligned} D_1''(1^m) &= \{p, q \leftarrow \mathrm{BW}(1^m); n \leftarrow p \cdot q; h \leftarrow \mathbb{Z}_n^{+1}; \\ &\qquad s \leftarrow \mathrm{CS}(n, s^2 \in \mathbb{Z}_n^*) \,:\, (n, h, s)\} \\ D_{0,1}''(1^m) &= \{p, q \leftarrow \mathrm{BW}(1^m); n \leftarrow p \cdot q; h \leftarrow \mathbb{Z}_n^{+1}; \\ &\qquad s \leftarrow \mathrm{CS}(n, s^2 \in 4h + \mathbb{Z}_n^*) \,:\, (n, h, s)\} \end{aligned} \tag{2.7}$$

is at most

$$4 \cdot \frac{2(p+q-1)}{(p-1)(q-1)/4} = \frac{32(p+q-1)}{(p-1)(q-1)}\,.$$

Now,

$$s^2 - 4h \in \mathbb{Z}_n^* \iff s^2 \in 4h + \mathbb{Z}_n^* \quad \text{and} \quad s \in \mathbb{Z}_n^* \iff s^2 \in \mathbb{Z}_n^*$$

so that the advantage of an adversary distinguishing the two distributions

$$\begin{aligned} D_1(1^m) &= \{p, q \leftarrow \mathrm{BW}(1^m); n \leftarrow p \cdot q; h \leftarrow \mathbb{Z}_n^{+1}; \\ &\qquad s \leftarrow \mathrm{CS}(n, s \in \mathbb{Z}_n^*) \,:\, (n, h, s)\} \\ D_{0,1}(1^m) &= \{p, q \leftarrow \mathrm{BW}(1^m); n \leftarrow p \cdot q; h \leftarrow \mathbb{Z}_n^{+1}; \\ &\qquad s \leftarrow \mathrm{CS}(n, s^2 - 4h \in \mathbb{Z}_n^*) \,:\, (n, h, s)\} \end{aligned} \tag{2.8}$$

is at most

$$\frac{32(p+q-1)}{(p-1)(q-1)}\,.$$

Hence $D_{0,1}(1^m)$ and $D_1(1^m)$ are indistinguishable and the problems QIP and QIP_0 are equivalent (since $D_{0,0}(1^m)$ and $D_0(1^m)$ are the same distribution).

So, to prove the equivalence of QRP and QIP, we only need to prove the equivalence of QRP and QIP_0. Given a distinguisher $\mathcal{A}$ for QIP_0, which takes as input (n, h, s) where n is a Blum-Williams integer, $h \leftarrow \mathbb{Z}_n^{+1}$ and $s \in \mathbb{Z}_n^*$, and outputs a bit b indicating (n, h, s) is from the distribution $D_{0,b}(1^m)$, we simulate a distinguisher $\mathcal{B}$ for QRP, which takes as input (n, s) where n is a Blum-Williams integer and $s \in \mathbb{Z}_n^*$, and outputs a bit b indicating (n, s) is from the distribution $E_b(1^m)$ as follows

1. $\mathcal{B}$ randomly chooses $\sigma \leftarrow \mathbb{Z}_n^*$ and sets $h = \frac{\sigma^2 - s}{4}$.

2. If $h \notin \mathbb{Z}_n^{+1}$, $\mathcal{B}$ discards this σ and goes back to the previous step.

3. $\mathcal{B}$ invokes $\mathcal{A}$ and inputs (n, h, σ) to it.

4. $\mathcal{B}$ outputs $b \leftarrow \mathcal{A}(n, h, \sigma)$.

Since $s = \sigma^2 - 4h$, $(n, h, \sigma) \leftarrow D_{0,b}(1^m)$ implies that

if $b = 0$, $\sigma \leftarrow \mathrm{CS}(n, s \in \mathbb{Z}_n^{-1} \cup QR(n))$;

else if $b = 1$, $\sigma \leftarrow \mathrm{CS}(n, s \in \mathbb{Z}_n^*)$.

Thus $\mathcal{B}$ is correct whenever $\mathcal{A}$ is. Hence QIP_0 is equivalent to QRP.

2.3.2 Two Related Problems

We define two more problems, very closely related to the QIP, which we will use in the proof of security in the Section 2.5.

QIP_1 **Problem.** We define the QIP_1 *problem* as the problem of efficiently distinguishing the following two distributions:

$$\begin{aligned} D_{1,0}(1^m) = \{&p, q \leftarrow \mathrm{BW}(1^m); n \leftarrow p \cdot q; d \leftarrow \{0,1\}; h_0, h_1 \leftarrow \mathbb{Z}_n^{+1}; \\ &s \leftarrow \mathrm{CS}(n, s^2 - 4h_d \in \mathbb{Z}_n^{-1} \cup QR(n)) \,:\, (n, h_0, h_1, s)\} \\ D_{1,1}(1^m) = \{&p, q \leftarrow \mathrm{BW}(1^m); n \leftarrow p \cdot q; h_0, h_1 \leftarrow \mathbb{Z}_n^{+1}; s \leftarrow \mathbb{Z}_n^* \,:\, (n, h_0, h_1, s)\} \end{aligned}$$

We say that algorithm $\mathcal{A}$ has *advantage* ϵ in solving QIP_1 if we have that:

$$\Big| \Pr[(n, h_0, h_1, s) \leftarrow D_{1,0}(1^m) : \mathcal{A}(n, h_0, h_1, s) = 1] \\ - \Pr[(n, h_0, h_1, s) \leftarrow D_{1,1}(1^m) : \mathcal{A}(n, h_0, h_1, s) = 1]\Big| = \epsilon. \quad (2.9)$$

We say that QIP_1 is *intractable* if all polynomial time (in m) algorithms have a negligible (in m) advantage in solving QIP_1.

By a simple simulation argument, we can prove the following theorem:

Theorem 2.3.1. The QIP_1 problem is intractable if and only if the QIP problem is so.

To prove the equivalence of QIP and QIP_1, given D_0 and D_1, we choose randomly $h_1 \leftarrow \mathbb{Z}_n^{+1}$ and create $D_{1,0}$ and $D_{1,1}$. If we can distinguish between these two with probability ϵ, then with prob $\epsilon/2$ we can distinguish between the given two distribution.

QIP_2 **Problem.** We define the QIP_2 *problem* as the problem of efficiently distinguishing the following two distributions:

$$\begin{aligned} D_{2,0}(1^m) = \{&p, q \leftarrow \mathrm{BW}(1^m); n \leftarrow p \cdot q; h_0, h_1 \leftarrow \mathbb{Z}_n^{+1}; \\ &s \leftarrow \mathrm{CS}(n, s^2 - 4h_0 \in \mathbb{Z}_n^{-1} \cup QR(n)) \, : \, (n, h_0, h_1, s)\} \\ D_{2,1}(1^m) = \{&p, q \leftarrow \mathrm{BW}(1^m); n \leftarrow p \cdot q; h_0, h_1 \leftarrow \mathbb{Z}_n^{+1}; \\ &s \leftarrow \mathrm{CS}(n, s^2 - 4h_1 \in \mathbb{Z}_n^{-1} \cup QR(n)) \, : \, (n, h_0, h_1, s)\} \end{aligned}$$

We say that algorithm $\mathcal{A}$ has *advantage* ϵ in solving QIP_2 if we have that:

$$\Big| \Pr[(n, h_0, h_1, s) \leftarrow D_{2,0}(1^m) : \mathcal{A}(n, h_0, h_1, s) = 1] \\ - \Pr[(n, h_0, h_1, s) \leftarrow D_{2,1}(1^m) : \mathcal{A}(n, h_0, h_1, s) = 1]\Big| = \epsilon. \quad (2.10)$$

We say that QIP_2 is *intractable* if all polynomial time (in m) algorithms have a negligible (in m) advantage in solving QIP_2.

By a simple hybrid argument, we can prove the following theorem:

Theorem 2.3.2. The QIP_2 problem is intractable if and only if the QIP problem is so.

To prove the equivalence of QIP and QIP_2, given D_0 and D_1, we choose randomly $h' \leftarrow \mathbb{Z}_n^{+1}$ and create D_1'. Then we treat D_1 and D_1' as $D_{2,0}$ and $D_{2,1}$. If we can distinguish between these two then we contradict the indistinguishability between D_0 and D_1 and between D_0 and D_1'.

Thus we have proved that all the five problems: QRP, QIP, QIP_0, QIP_1, and QIP_2, are "equivalent".

2.4 Our Construction

In this section we present our main construction: a public-key encryption with keyword search under the intractability assumption of the Quadratic Residuousity Problem (QRP). We first formally state our main result. Then, in Subsection 2.4.1 we give an informal discussion where we sketch a preliminary (but flawed) construction, explain why it does not work, and how we fix it. Finally, in Subsection 2.4.2, we formally describe our public-key cryptosystem with keyword search.

Our result. In the rest of the chapter we prove the following

Theorem 2.4.1. Assume that the quadratic residuousity problem (QRP) is intractable. Then there exists (constructively) a public-key encryption scheme with keyword search.

2.4.1 An Informal Discussion

A first (not yet anonymous) construction. The first approach is a very natural one — we simply apply the Cocks identity-based encryption scheme in place of Boneh-Franklin scheme as done in the PEKS scheme presented in [9]. Given a security parameter 1^m, for $m \in \mathbb{Z}^+$, the user Alice uses algorithm CC-Setup to generate two sufficiently large primes p, q such that both p and q are congruent to $3 \mod 4$ and a cryptographic hash function H (assumed to behave like a random oracle in the analysis); she then outputs the public parameters $A_{pub} = (n, H)$ and keeps secret the master secret key $A_{priv} = (p, q)$. Alice treats each keyword W as an identity and, using algorithm CC-KeyGen, computes a square root g of $h = H(W)$ or $-h$ depending on which one is a square modulo n, and supplies the email server with $T_W = g$ as the trapdoor for W. A user Bob wishing to send an encrypted email to Alice with the keyword W uses algorithm CC-Encrypt to encrypt the string 1^k, where $k = |W|$, using W as the necessary input identity, thus obtaining $\mathsf{ksEnc}(A_{pub}, W) = s = (s_1, \ldots, s_k, s_{k+1}, \ldots, s_{2k})$. The server then decrypts s as in algorithm CC-Decrypt and outputs 'yes' if the decryption returns 1^k, else outputs 'no'.

The problem with the first approach. The above scheme does not satisfy anonymity because the Cocks identity-based encryption scheme is not *public-key private* or *anonymous*, in the sense of Bellare et al. [3], and thus the ciphertext returned by algorithm PEKS may reveal more information than desired about the searchable keyword. This fact was already briefly mentioned in [9], but it is useful to analyze it in greater detail here to understand how we will obtain our main construction. Specifically, to clarify this fact, we note that for $i = 1, \ldots, k$, it holds that

$$s_i^2 - 4h = (t_i + h/t_i)^2 - 4h = (t_i - h/t_i)^2 \mod n$$

and therefore, except with negligible probability, $\left(\frac{s_i^2 - 4h}{n}\right) = +1$ and, analogously, for $i = k+1, \ldots, 2k$, it holds that

$$s_i^2 + 4h = (t_i - h/t_i)^2 + 4h = (t_i + h/t_i)^2 \mod n$$

and therefore, except with negligible probability, $\left(\frac{s_i^2 + 4h}{n}\right) = +1$.

On the other hand, for any other keyword $W' \neq W$, if $h' = H(W')$, the quantities $s_i^2 - 4h'$ and $s_i^2 + 4h'$ are not necessarily squares and their Jacobi symbols $\left(\frac{s_i^2 \pm 4h'}{n}\right)$ may be -1. In fact, when h' is randomly chosen in $\mathbb{Z}_n^*$, for each $i \in \{1, \ldots, 2k\}$, it holds that $\left(\frac{s_i^2 \pm 4h'}{n}\right) = -1$ exactly half the time $s_i^2 \pm 4h'$ is in $\mathbb{Z}_n^*$. Thus, an outsider can easily find out whether a keyword W is in the message or not with some non-negligible probability, which is not desirable.

Fixing the problems and ideas behind our construction. At a very high level, we still would like to use the approach in [9]; which, very roughly speaking, might be abstracted as follows: a public-key encryption with keyword search is 'carefully computed' as the output of an identity-based encryption algorithm on input (a function of) the keyword W as the identity and a plaintext sent in the clear. the computation of the searchable ciphertext is such that (with high probability) the plaintext sent in the clear is the actual decryption of this ciphertext if and only if the trapdoor associated with the same keyword W is used for decryption.

However, the main difficulty in implementing this approach with (a modification of) the Cocks scheme CC-IBE is in obtaining a modification which additionally satisfies the 'public-key privacy' or 'anonymity' property. We solve this problem by modifying the distribution of the ciphertext in CC-IBE, so that its modified distribution is 'properly randomized', and, when used in the context of a ciphertext associated with our PEKS scheme, does not reveal which keyword is being used. The randomization of the ciphertext not only has to guarantee that the ciphertext does not reveal the identity used (or, in other words, the integer $h = H(ID)$), but also has to guarantee that the distribution remains the same when it is matched with another identity (for example, another integer $h' = H(ID')$). In this randomization process, we have to take care of two main technical obstacles, one related to the distribution of the integers s_i with respect to the Jacobi symbols $\left(\frac{s_i^2 \pm 4h}{n}\right)$; and another related to efficiently guaranteeing that all values s_i are constructed using uniformly and independently distributed hashes (or, functions of the keyword) playing as the identity. We achieve this through two levels of randomization. First, the ciphertext contains $4k$ integers s_i in $\mathbb{Z}_n^*$ such that the Jacobi symbols of the related expressions $s_i^2 - 4h$ are uniformly distributed in $\{-1, +1\}$ whenever $s_i^2 \pm 4h$ is in $\mathbb{Z}_n^*$. Second, to make sure that these Jacobi symbols are also independently distributed, we do not use a single value h, but use a uniformly and independently distributed h_i for each index i.

2.4.2 Formal Description

We denote our PEKS scheme as CS-PEKS = (CS-KeyGen, CS-ksEnc, CS-Trapdoor, CS-Test). CS-PEKS uses a cryptographic hash function $H : \{0,1\}^k \rightarrow \mathbb{Z}_n^{+1}$ (which is assumed in the analysis to behave as a random oracle). We denote by m the security parameter and by k the length of keywords. We assume wlog that $k = \Theta(m^c)$ for some constant $c > 0$ (concrete values for m, k can be $m = 1024$ and $k = 160$). CS-PEKS can be described as follows:

CS-KeyGen(1^m)**:** On input security parameter 1^m in unary, for $m \in \mathbb{Z}^+$, do the following:

1. randomly choose two primes p, q of length $m/2$, and such that both p and q are congruent to $3 \mod 4$ and set $n = pq$;
2. Set $A_{pub} = (n, 1^k)$ and $A_{priv} = (p, q)$, and output: (A_{pub}, A_{priv}).

CS-ksEnc(A_{pub}, W)**:** Let $A_{pub} = (n, 1^k)$, $W \in \{0,1\}^k$, and do the following:

```
1. For each $i = 1, \ldots, 4k$,
     compute $h_i = H(W|i)$;
     randomly and independently choose $u_i \in \mathbb{Z}_n^*$;
     if $\left(\frac{u_i^2 - 4h_i}{n}\right) = +1$ then randomly and independently choose $t_i \in \mathbb{Z}_n^{+1}$
       and set $s_i = (t_i + h_i/t_i) \mod n$.
     if $\left(\frac{u_i^2 - 4h_i}{n}\right) \in \{-1, 0\}$ then set $s_i = u_i$.
2. Output $\vec{s} = (s_1, \ldots, s_{4k})$.
```

CS-Trapdoor(A_{priv}, W)**:** Let $A_{priv} = (p, q)$, $W \in \{0,1\}^k$, and do the following:

```
1. For $i = 1, \ldots, 4k$;
     compute $h_i = H(W|i)$;
     use $p, q$ to randomly choose $g_i \in \mathbb{Z}_n^*$ (if any) such that $g_i^2 = h_i \mod n$;
     if $h_i$ has no square root modulo $n$, then set $g_i = \perp$;
2. return: $(g_1, \ldots, g_{4k})$.
```

CS-Test(A_{pub}, s, T_W)**:** Let $T_W = (g_1, \ldots, g_{4k})$ and $\vec{s} = (s_1, \ldots, s_{4k})$, and do the following:

```
1. For $i = 1, \ldots, 4k$,
     if $g_i = \perp$ then set $\bar{t}_i = \perp$;
     if $g_i^2 = h_i \mod n$ then
       if $\left(\frac{s_i^2 - 4h_i}{n}\right) = +1$ then set $\bar{t}_i = \left(\frac{s_i + 2g_i}{n}\right)$;
         otherwise set $\bar{t}_i = \perp$;
2. output 'yes' if $\bar{t}_i \in \{+1, \perp\}$ for all $i = 1, \ldots, 4k$; otherwise output 'no'.
```

Remark 2.4.2 (Ciphertext distribution and scheme parameters). We note that the distribution of the ciphertext $\vec{s} = (s_1, \ldots, s_{4k})$ returned by algorithm $\mathsf{CS\text{-}ksEnc}$ has only negligible statistical distance from the distribution where each element s_i is uniformly distributed among the integers such that $s_i^2 - 4h_i \in QR(n)$ with probability 1/2, or $s_i^2 - 4h_i \in \mathbb{Z}_n^{-1}$ with probability 1/2.

We note that it is essential to choose our scheme's parameter $k = \Theta(m^c)$, for some $c > 0$, to guarantee that the consistency properties of $\mathsf{CS\text{-}PEKS}$ are satisfied in an asymptotic sense. Good practical choices for parameters m, k include setting $m = 1024$ and $k = 160$.

2.5 Properties of Our Construction

In Subsections 2.5.1 and 2.5.2 we prove the consistency and security properties of our public-key encryption scheme with keyword search.

2.5.1 Proof of Consistency

We prove the right-keyword consistency and the adversary-based consistency of $\mathsf{CS\text{-}PEKS}$ in this subsection.

Right-keyword consistency. For $i = 1, \ldots, 4k$, whenever $\left(\frac{s_i^2 - 4h_i}{n}\right) = +1$, it always holds that $h_i = g_i^2 \mod n$ and it never holds that $\bar{t}_i = \left(\frac{s_i + 2g_i}{n}\right) = -1$. The latter fact is proved by observing that, for $i = 1, \ldots, 4k$, it holds that

$$s_i + 2g_i = t_i + h_i/t_i + 2g_i = t_i \cdot (1 + g_i^2/{t_i}^2 + 2g_i/t_i) = t_i \cdot (1 + g_i/t_i)^2 \mod n,$$

and thus, except with negligible probability,

$$\bar{t}_i = \left(\frac{s_i + 2g_i}{n}\right) = \left(\frac{t_i}{n}\right) = +1.$$

Now, the above equalities do not hold only when $s_i + 2g_i \mod n$ is not in $\mathbb{Z}_n^*$, in which case it still holds that $\left(\frac{s_i + 2g_i}{n}\right) = 0 \neq -1$. As a consequence of these two facts, the right-keyword consistency property holds with probability 1.

Adversary-based consistency. Let $\mathcal{A}$ be an attacker running in polynomial time and having access to a random oracle H, that tries to violate the computational consistency property of CS-PEKS. By (W_0, W_1) we denote the pair of distinct k-bit keywords that $\mathcal{A}$ returns in the CS-PEKS adversary-based consistency game of Section 2.2. Also, by $q_1, \ldots, q_\ell$ we denote the queries made by $\mathcal{A}$ to H, and we define the following two events:

$$\begin{aligned} \text{Bad} \quad &= \quad \text{`either there exist } i,j \in \{1,\ldots,4k\} \text{ such that } H(W_0|i) = H(W_1|j) \\ &\qquad \text{or there exist } i,j \in \{1,\ldots,\ell\} \text{ such that } H(q_i) = H(q_j)\text{'} \\ \text{Success} \quad &= \quad \text{`}W_0 \neq W_1 \text{ and } \mathsf{CS\text{-}Test}(A_{pub}, s, T_{W_1}) = 1\text{'}, \\ &\qquad \text{where } s = \mathsf{CS\text{-}ksEnc}(A_{pub}, W_0) \text{ and } T_{W_1} = \mathsf{CS\text{-}Trapdoor}(A_{priv}, W_1). \end{aligned}$$

Then we have that $\text{Adv}_{\mathcal{A}}(1^k)$ is equal, by definition, to Prob[*Success*], and this probability can be rewritten as

$$\text{Prob}[\,Success\,] \leq \text{Prob}[\,Bad\,] + \text{Prob}[\,Success|\neg Bad\,].$$

To compute an upper bound of Prob[*Success*], we first note that we can easily bound Prob[*Bad*] as at most $(4k+1)^2(\ell+1)^2/2^m$, which is negligible in m as k, ℓ are bounded by a polynomial in m. Then we note that Prob[$Success|\neg Bad$] can be upper-bounded as the probability that there exists a word W_1 for which all values $\bar{t}_i$ computed by algorithm CS-Test on input (A_{pub}, s, T_{W_1}), where $s = \mathsf{CS\text{-}ksEnc}(A_{pub}, W_0)$, are either $+1$ or $\perp$. This happens if, for $i = 1, \ldots, 4k$, none of the values $h'_i := H(W_1|i)$ simultaneously satisfies the following two conditions:

1. $\left(\frac{s_i^2 - 4h'_i}{n}\right) = +1$;
2. $\left(\frac{s_i + 2g'_i}{n}\right) = -1$, where g'_i is the random value computed when running CS-Trapdoor on input (A_{priv}, W_1) such that $(g'_i)^2 = h'_i \mod n$.

Since H is a random oracle, and since we condition on $\neg Bad$, the probability that values g'_i, h'_i simultaneously satisfy the above two conditions is at least $1/4 - \delta$ for some δ negligible in m, even, conditioned on the values g'_j, h'_j, for all $j < i$. Therefore, the probability that for a single word W_1 none of the g'_i, h'_i satisfies the above two conditions is at most $(3/4+\delta)^{4k}$. Finally, a simple union bound implies that the probability that there exists a k-bit W_1 for which none of the g'_i, h'_i satisfies the above two conditions is at most $2^k \cdot (3/4+\delta)^{4k} < (2/3)^k$, which is negligible in m, as k was assumed to be $\Theta(m^c)$, for some $c > 0$.

2.5.2 Proof of Security

Let $\mathcal{A}$ be a polynomial-time algorithm that attacks CS-PEKS and succeeds in breaking with advantage ϵ, and while doing that, it makes at most $q_H > 0$ queries to the random oracle H and at most $q_T > 0$ trapdoor queries. We will show that ϵ is negligible in m or otherwise $\mathcal{A}$ can be used to construct an algorithm $\mathcal{B}$ that violates the intractability of the QRP. More precisely, we will attempt to violate the intractability of one among two problems QIP_1 and QIP_2 which, along with the QIP, are computationally equivalent to the QRP (proved in Section 2.3).

We prove this by defining a sequence of games, which we call 'CS-PEKS Security Game t', for $t = 0, \ldots, 4k$, which are all variations of the PEKS Security Game defined in Section 2.2.

CS-PEKS Security Game t:

1. Algorithm $\mathcal{B}$ takes as input (n, h_0, h_1, s), where n is a Blum-Williams integer, and $h_0, h_1 \in \mathbb{Z}_n^{+1}$ and $s \in \mathbb{Z}_n^*$.
2. First of all $\mathcal{B}$ runs the CS-KeyGen(1^m) algorithm to generate $A_{pub} = (n, 1^k)$ and $A_{priv} = (p, q)$; afterwards, it gives A_{pub} to the attacker $\mathcal{A}$.
3. $\mathcal{A}$ can adaptively ask for outputs from the random oracle H to any inputs of its choice. To respond to H-queries, algorithm $\mathcal{B}$ maintains a list of tuples $\langle W_i, j, h_{i,j}, g_{i,j}, d(i,j), c(i,j)\rangle$ called the H-list. The list is initially empty. When $\mathcal{A}$ queries the random oracle H at a point $(W_i|j)$, for $W_i \in \{0,1\}^k$ and $j \in \{1, \ldots, 4k\}$, algorithm $\mathcal{B}$ responds as follows.
   If tuple $\langle W_i, j, h_{i,j}, g_{i,j}, d(i,j), c(i,j)\rangle$ appears on the H-list then algorithm $\mathcal{B}$ responds with $H(W_i|j) = h_{i,j} \in \mathbb{Z}_n^{+1}$.
   Otherwise, $\mathcal{B}$ uniformly chooses $d(i,j) \in \{0,1\}$, $r_{i,j} \in \mathbb{Z}_n^*$, and randomly choose $c(i,j) \in \{0,1\}$ such that $c(i,j) = 0$ with probability $1/(q_T+1)$ and $c(i,j) = 1$ with probability $1 - 1/(q_T+1)$.

If $c(i,j) = 1$ then $\mathcal{B}$ computes $h_{i,j} = (-1)^{d(i,j)} \cdot r_{i,j}^2 \mod n$; sets $g_{i,j} = \perp$ if $d(i,j) = 1$, or $g_{i,j} = r_{i,j}$ if $d(i,j) = 0$; adds $\langle W_i, j, h_{i,j}, g_{i,j}, d(i,j), c(i,j)\rangle$ to the H-list and responds with $h_{i,j}$ to the H-query $(W_i|j)$.

If $c(i,j) = 0$, then $\mathcal{B}$ sets $d = d(i,j)$, computes $h_{i,j} = h_d \cdot r_{i,j}^2 \mod n$, sets $g_{i,j} = r_{i,j}$, adds $\langle W_i, j, h_{i,j}, g_{i,j}, d(i,j), c(i,j)\rangle$ to the H-list and responds with $h_{i,j}$ to the H-query $(W_i|j)$.

4. $\mathcal{A}$ can adaptively ask for the trapdoor T_W for any keyword $W \in \{0,1\}^k$ of his choice, to which $\mathcal{B}$ responds as follows.

   If tuple $\langle W_i, j, h_{i,j}, g_{i,j}, d(i,j), c(i,j)\rangle$ already appears on the H-list, for some $j \in \{1, \ldots, 4k\}$ and $W_i = W$, then $\mathcal{B}$ responds with $(g_{i,1}, \ldots, g_{i,4k})$ to the trapdoor query W if $c(i,j) = 1$ or reports failures and halts if $c(i,j) = 0$.

   Otherwise $\mathcal{B}$ randomly chooses $d(i,j) \in \{0,1\}$ and $r_{i,j} \in \mathbb{Z}_n^*$; computes $h_{i,j} = (-1)^{d(i,j)} \cdot r_{i,j}^2 \mod n$; sets $g_{i,j} = \perp$ if $d(i,j) = 1$ or $g_{i,j} = r_{i,j}$ otherwise, responds with $(g_{i,1}, \ldots, g_{i,4k})$ to the trapdoor query W and inserts $\langle W_i, j, h_{i,j}, g_{i,j}, d(i,j), c(i,j)\rangle$ in H-list.

5. The attacker $\mathcal{A}$ sends the two keywords W_0, W_1 on which it wishes to be challenged (for which it did not previously ask for trapdoors T_{W_0} or T_{W_1}).

   If the two tuples

   $$\langle W_u, j, h_{u,j}, g_{u,j}, d(u,j), c(u,j)\rangle \text{ and } \langle W_v, j, h_{v,j}, g_{v,j}, d(v,j), c(v,j)\rangle$$

   satisfying $W_u = W_0$, $W_v = W_1$, $j = t$, and $((c(u,j) = 0) \vee (c(v,j) = 0))$, are not in H-list, then $\mathcal{B}$ reports failures and halts.

   Otherwise $\mathcal{B}$ computes $(s_1, \ldots, s_{4k})$ as follows:

   - $s_1, \ldots, s_{t-1}$ are computed as from algorithm $\mathsf{CS\text{-}ksEnc}$ on input A_{pub}, W_0;
   - s_t is set equal to $s \cdot r_{i,t} \mod n$;
   - $s_{t+1}, \ldots, s_{4k}$ are computed as from algorithm $\mathsf{CS\text{-}ksEnc}$ on input A_{pub}, W_1.

6. Given challenge $(s_1, \ldots, s_{4k})$, $\mathcal{A}$ can continue to ask for random oracle H's outputs for any input of its choice, and for trapdoors T_W for any keyword W of his choice as long as $W \neq W_0, W_1$; these are answered as in items 3 and 4, respectively.
7. $\mathcal{A}$ outputs $out \in \{0,1\}$.

By using a standard hybrid argument on our assumption that $\mathcal{A}$ breaks the security of CS-PEKS with probability ϵ, we obtain that there exists $t \in \{1, \ldots, 4k\}$ such that

$$|\,\text{Prob}[\mathcal{A}_t = 1\,] - \text{Prob}[\mathcal{A}_{t+1} = 1\,]\,| \geq \epsilon/4k, \tag{2.11}$$

where by $\mathcal{A}_t = 1$ we denote the event that $\mathcal{A}$ returns 1 in the real attack game given that the challenge ciphertext $\vec{s}$ had been computed as follows: $s_1, \ldots, s_t$ are computed as in algorithm CS-ksEnc on input A_{pub}, W_0; and $s_{t+1}, \ldots, s_{4k}$ are computed as in algorithm CS-ksEnc on input A_{pub}, W_1. Similarly, as in [9], we can obtain that with probability at least $\epsilon/4k$, $\mathcal{A}$ queries at least one of the two H-queries $(W_0|t)$, $(W_1|t)$.

The proof continues by considering two cases according to whether only one of the two queries is made or both of them are made. In the first case, we show that $\mathcal{B}$ violates the intractability of the QIP_1 problem, and in the second case, we show that it violates the intractability of the QIP_2 problem.

Case (a). We now consider the case when only one of the two H-queries, say, $(W_0|t)$, is made by algorithm $\mathcal{A}$.

We continue the proof by noting that bit $c(i, t)$ associated with the query $(W_0|t)$, where the i-th queried keyword is W_0, satisfies $c(i, t) = 0$ with probability $1/(q_T + 1)$. Assuming that $c(i, t) = 0$, we evaluate the distribution of ciphertext $\vec{s}$ in CS-PEKS Security Game t, for $t = 1, \ldots, 4k$.

First, we let $d = d(i, t)$ and observe that when $(n, h_0, h_1, s) \in D_{1,0}(1^m)$, the ciphertext $\vec{s}$ in CS-PEKS Security Game t appears to $\mathcal{A}$ to be distributed exactly as if $s_1, \ldots, s_t$ were computed as in algorithm CS-ksEnc on input A_{pub}, W_0, and $s_{t+1}, \ldots, s_{4k}$ were computed as in algorithm CS-ksEnc on input A_{pub}, W_1. This can be seen by observing that we assumed that $c(i, t) = 0$ and thus $H(W_0|t) = h_d \cdot r_{i,t}^2$; then, it holds that s_t is randomly distributed among the integers such that $s_t^2 - 4H(W_0|t) \in \mathbb{Z}_n^{-1} \cup QR(n))$ as it satisfies $s_t^2 - 4H(W_0|t) = (sr_{i,t})^2 - 4h_d r_{i,t}^2 = r_{i,t}^2(s^2 - 4h_d)$, where $s^2 - 4h_d$ is also randomly distributed among the integers in $\mathbb{Z}_n^{-1} \cup QR(n))$ as $(n, h_0, h_1, s) \in D_{1,0}(1^m)$. Therefore, the probability that $\mathcal{A}$ returns 1 in CS-PEKS Security Game t when $(n, h_0, h_1, s) \in D_{1,0}(1^m)$ is the same as the probability that $\mathcal{A}_t = 1$.

We now consider the case when $(n, h_0, h_1, s) \in D_{1,1}(1^m)$, the ciphertext $\vec{s}$ in CS-PEKS Security Game t appears to $\mathcal{A}$ to be distributed exactly as if $s_1, \ldots, s_{t-1}$ were computed as in algorithm CS-ksEnc on input A_{pub}, W_0, and $s_t, \ldots, s_{4k}$ were computed as in algorithm CS-ksEnc on input A_{pub}, W_1. This can be seen by observing that s_t is uniformly distributed in $\mathbb{Z}_n^*$ by definition of $D_{1,1}$, and that if s_t were computed as in algorithm CS-ksEnc on input A_{pub}, W_1, it would appear to $\mathcal{A}$ to have the same distribution, as we assumed that $(W_1|t)$ was not queried by $\mathcal{A}$. Therefore, the probability that $\mathcal{A}$ returns 1 in CS-PEKS Security Game t when $(n, h_0, h_1, s) \in D_{1,1}(1^m)$ is the same as the probability that $\mathcal{A}_{t-1} = 1$.

This implies that the probability that $\mathcal{B}$ distinguishes $D_{1,0}(1^m)$ from $D_{1,1}(1^m)$ is the probability $1/(e \cdot q_T)$ that $\mathcal{B}$ does not halt in CS-PEKS Security Game t, times the probability $\epsilon/(4k \cdot (q_T + 1))$ that $\mathcal{A}$ makes only one H-queries among $(H_0|t), (H_1|t)$ and it holds that the associated bit $c_{\cdot,t} = 0$.

Since ϵ is assumed to be not negligible, so is the quantity $\epsilon/(e \cdot 4k \cdot (q_T + 1))$, and therefore $\mathcal{B}$ violates the intractability of the QIP_1 problem.

Case (b). We now consider the case when both H-queries $(W_0|t)$, $(W_1|t)$ are made by algorithm $\mathcal{A}$.

We continue the proof by noting that bits $c(i, t), c(j, t)$ associated with the two queries, where the i-th queried keyword is W_i and the j-th queried keyword is W_1, satisfy $c(i, t) = c(j, t) = 0$ with probability at least $1/(q_T + 1)^2$. Under this setting, we evaluate the distribution of ciphertext $\vec{s}$ in CS-PEKS Security Game t, for $t = 1, \ldots, 4k$.

First, we observe that when $(n, h_0, h_1, s) \in D_{2,0}(1^m)$, the ciphertext $\vec{s}$ in CS-PEKS Security Game t appears to $\mathcal{A}$ to be distributed exactly as if $s_1, \ldots, s_t$ were computed as in algorithm CS-ksEnc on input A_{pub}, W_0, and $s_{t+1}, \ldots, s_{4k}$ were computed as in algorithm CS-ksEnc on input A_{pub}, W_1. This can be seen by observing that we assumed that $c(i, t) = 0$ and thus $H(W_0|t) = h_0 \cdot r_{i,t}^2$; then, it holds that s_t is randomly distributed among the integers such that $s_t^2 - 4H(W_0|t) \in \mathbb{Z}_n^{-1} \cup QR(n))$ as it satisfies $s_t^2 - 4H(W_0|t) = (sr_{i,t})^2 - 4h_0 r_{i,t}^2 = r_{i,t}^2(s^2 - 4h_0)$, where $s^2 - 4h_0$ is also randomly distributed among the integers in $\mathbb{Z}_n^{-1} \cup QR(n))$ as $s \in D_{2,0}(1^m)$. Therefore, the probability that $\mathcal{A}$ returns 1 in CS-PEKS Security Game t when $(n, h_0, h_1, s) \in D_{2,0}(1^m)$ is the same as the probability that $\mathcal{A}_t = 1$.

Analogously, when $(n, h_0, h_1, s) \in D_{2,1}(1^m)$, the ciphertext $\vec{s}$ in CS-PEKS Security Game t appears to $\mathcal{A}$ to be distributed exactly as if $s_1, \ldots, s_{t-1}$ were computed as in algorithm $\mathsf{CS\text{-}ksEnc}$ on input A_{pub}, W_0, and $s_t, \ldots, s_{4k}$ were computed as in algorithm $\mathsf{CS\text{-}ksEnc}$ on input A_{pub}, W_1. This can be seen as before by again observing that we assumed that $c(j,t) = 0$ and thus $H(W_1|t) = h_1 \cdot r_{j,t}^2$; then, it holds that s_t is randomly distributed among the integers such that $s_t^2 - 4H(W_1|t) \in \mathbb{Z}_n^{-1} \cup QR(n))$ as it satisfies $s_t^2 - 4H(W_1|t) = (sr_{j,t})^2 - 4h_1 r_{j,t}^2 = r_{j,t}^2(s^2 - 4h_1)$, where $s^2 - 4h_1$ is also randomly distributed among the integers in $\mathbb{Z}_n^{-1} \cup QR(n))$ as $s \in D_{2,1}(1^m)$. Therefore, the probability that $\mathcal{A}$ returns 1 in $\mathsf{CS\text{-}PEKS}$ Security Game t when $(n, h_0, h_1, s) \in D_{2,1}(1^m)$ is the same as the probability that $\mathcal{A}_{t-1} = 1$.

This implies that $\mathcal{B}$ distinguishes $D_{2,0}(1^m)$ from $D_{2,1}(1^m)$ is the probability $1/(e \cdot q_T)$ that $\mathcal{B}$ does not halt in $\mathsf{CS\text{-}PEKS}$ Security Game t, times the probability $\epsilon/(4k \cdot (q_T + 1)^2)$ that $\mathcal{A}$ makes both H-queries $(H_0|t), (H_1|t)$ and it holds that $c_{i,t} = c_{j,t} = 0$.

Since ϵ is assumed to be not negligible, then so is the quantity $\epsilon/(e \cdot 4k \cdot q_T(q_T + 1)^2)$, and therefore $\mathcal{B}$ violates the intractability of the QIP_2 problem.

Acknowledgements

Most of the work in Chapter 2 was done jointly with Dr. Giovanni Di Crescenzo, Telcordia Technologies, Piscataway NJ, while visiting the Center for Discrete Mathematics and Theoretical Computer Science (DIMACS), Rutgers University, Piscataway NJ, during July 2006 – August 2006 on a research fellowship from Minnesota Center for Industrial Mathematics. The work resulted in a paper [17] published in IndoCrypt 2007.

Chapter 3

Anonymous Signatures Schemes

3.1 Introduction

An anonymous signature is a signature scheme where the signature σ of a message m does not reveal the identity of the signer. Yang et al. [33] discussed the usefulness of anonymous signatures in many applications where anonymity is needed, including key exchange protocols, auction systems, and anonymous paper reviewing.

The notion of the anonymous signature was formalized much later than that of the anonymous encryption. Bellare et al. [4] had already defined in Asiacrypt 2001 key-privacy, or anonymity of an encryption scheme, as indistinguishability of ciphertexts encrypted by different public keys, that is, an eavesdropper cannot obtain any information about the recipient (corresponding to the public key) from the ciphertext. However, one problem in introducing the idea of anonymity to digital signatures is that a signature is publicly verifiable. So, if there are only a few candidate signers, an adversary attacking the anonymity of the digital signature can simply try verification of the message-signature pair with respect to all candidate public keys to break anonymity. Therefore, as long as the adversary obtains both the message and the signature, it seems that anonymity is impossible.

Yang et al. resolved the paradox by guaranteeing the anonymity only when the adversary obtains the signature and not the message, or when there is some randomness in the message not revealed to the adversary. In fact, there are numerous applications in which not revealing the complete message is justifiable; for example, in the key transport example given by Yang et al., Bob already knows what Alice's message should be from previous communication, so Alice may send only the anonymous signature without the message, and can authenticates Alice while protecting Alice's anonymity from eavesdroppers. In the case of an auction, a bidder may append some random string r to a message m, which is his bid, and sign it. After the auction ends, only the winner has to reveal the randomness r and thus his identity, and the other participants remain anonymous.

This idea of hidden randomness in the message is used by Fischlin [18] to propose an elegant generic transformation for anonymous signatures from ordinary signatures, by applying the idea of randomness extractor to extract the hidden randomness and use it for anonymizing the signature. Fischlin's formulation of anonymous signatures is slightly different, but essentially captures the same idea as that of Yang et al. Also in [34], Zhang and Imai suggested the notion of 'strong anonymous signatures', where they considered the case when there is not much uncertainty in the message.

3.1.1 Limits of the previous formalism

We revisit the formal definition of anonymous signature and show that previous formalisms of anonymous signature are not completely satisfactory in that they fail to capture the intuition fully and actually are inconsistent with some of the suggested applications. Also, we argue that a slightly different formalism captures the intuition better, retains the applicability, models the application scenarios with greater consistency, enables simpler constructions, and gives better security guarantee.

As explained above, in the current formalism, signer anonymity is based on hidden residual randomness of the message. As long as there is enough such randomness, the signer maintains anonymity, but of course the signature cannot be verified. Eventually the randomness in message is revealed explicitly or implicitly, and whoever has the complete message-signature pair can verify the signature.

In order to model this, Yang et al. and Fischlin formalize that each signer, having public key pk, has certain message distribution $\mathcal{M}(pk)$. Then, two key pairs (pk_0, sk_0), (pk_1, sk_1) are chosen and pk_0 and pk_1 are given to the adversary. Also, a message m is chosen from $\mathcal{M}(pk_b)$ with respect to a random bit $b \in \{0, 1\}$, and the signature $\sigma = \mathrm{Sig}(sk_b, m)$ is computed and given to the adversary. If the adversary cannot guess the random bit b with probability significantly greater than 1/2, then the signature scheme is considered anonymous.

But this formalism is not satisfactory in some aspects. First, this is in fact *inconsistent* to the suggested application of anonymous auction or anonymous paper review. In these cases, if m is the original intended message, then the signer adds some random string r to form appended message $m\|r$, and releases the message m, together with the signature σ of the appended message $m\|r$. From the point of view of an eavesdropper, different original messages m give different message distributions of the whole appended messages $m\|r$; the message distribution cannot be a function of the public key pk only, and in fact also depends on the partially revealed portion (m) of the message.

Second, this definition does not formally give a guarantee of infeasibility for someone other than the correct signer to come later and pretend that the signature is his. We call this property *unpretendability*. For an ordinary signature for which complete message-signature pair is released at once, this problem may be less crucial; the pair is publicly verifiable and the authorship can be attributed to the signer. But for an anonymous signature, where only a part of the message-signature pair is initially released, there is a theoretical possibility that someone other than the signer may come and claim the authorship of the message and signature. For example, in the anonymous paper review example, an author, A, of a paper $paper_A$ picks a random string r, computes $\sigma \leftarrow \mathrm{Sig}(sk_A, paper_A\|r)$, initially releases $(paper_A, \sigma)$, and only later reveals r when the paper is accepted. Now, if the anonymous signature is not unpretendable, then another author, B, may be able to compute r' satisfying $\mathrm{Vf}(pk_B, paper_A\|r', \sigma) = \mathsf{true}$ and use such an r' to claim authorship of $paper_A$.

Hence, we argue that this unpretendability should be an essential feature of an anonymous signature in the absence of which anonymous signature is in fact not applicable for many of originally proposed applications.

Note that we are not claiming that any of the actual schemes proposed in previous papers fail to satisfy unpretendability. But, this notion still needs to be formally defined and guaranteed for each anonymous scheme. In fact, in Section 3.3, we will give an example of an unforgeable signature scheme which provides complete anonymity but is not unpretendable. This means that unpretendability does not follow directly from unforgeability and/or anonymity, and warrants a separate definition.

Third, we feel that the idea of a signature of an unknown message is somewhat counter-intuitive. Intuitively, a signature is a proof of authorship for a given document. If we do not know the document in question, or if we are not sure whether the document ends with 'Therefore you should . . . ,' or 'Therefore you should not . . . ,' then the meaning of a signature for such uncertain document is at least debatable.

3.1.2 Our formalism

Discarding hidden randomness in the message. For these reasons, we propose a new definition of anonymous signatures as follows: first, instead of relying on the hidden residual randomness of the message, we introduce hidden randomness to the signature. Second, we not only formalize the notion of anonymity but also give explicit formalization of unpretendability.

In traditional digital signatures, signature generation is considered as a randomized algorithm. Therefore this strategy of explicit randomness is applicable no matter how much entropy (or lack thereof) the distribution of the message has.

This enables us to disregard the randomness in the message altogether, and use the available randomness directly to anonymize the public key. In fact, even when there is enough entropy in the message distribution, often the randomness is not diffused in the whole message but well-separated from the rest of the message and controllable by the signer. For example, in the bidding example where the bidder appends some random string r to the message m and then signs the appended message $m\|r$, certainly the distribution of this appended message has enough entropy which can be extracted back, but we believe this is artificial; the original message was m, and intuitively, the signer is not really interested in protecting the integrity of r, which is not part of his message m which he *really* wanted to sign. Hence, it is more natural to regard this r as a part of the signature, instead of regarding this as a part of the message which needs to be signed and protected.

Surfacing the verification token. Therefore, in our formalism, we split a digital signature $\tilde{\sigma}$ into two parts, $\tilde{\sigma} = (\sigma, \tau)$. We call τ a *verification token*, or in short, a token. Then σ, the rest of $\tilde{\sigma}$, is now just called the *signature*. The signature σ and the token τ are computed by the signature generation algorithm which takes the signer's secret key and the message m as inputs, and when m, σ, and τ are presented, anyone can verify the validity of the signature using the public key of the signer. But as long as τ is hidden, the adversary cannot break the anonymity of the signer just from the message m and the signature σ. Meanwhile, anyone to whom the token τ (along with the identity of the signer) is revealed may verify the signature.

Note that our formalism is just a specialization of the traditional formalism of digital signature, and not something incompatible; (σ, τ) together serve as a signature which is publicly verifiable and unforgeable according to the usual definition. We only enforce our signature to have this special format, and to have anonymity and unpretendability in addition to the unforgeability.

In short, we surfaced the hidden randomness of the anonymous signature explicitly as the verification token, and moved it from the message to the signature itself. Also we identified and formalized the unpretendability as another property that an anonymous signature should have.

Enhanced notion of security. Separating the randomness extraction from the anonymous signature not only results in a conceptually cleaner formalism but also enables us to guarantee better notion of security. In previous formalisms the verification token was 'diffused' in the message itself due to which an adversary attacking the anonymity of the signature could not choose the challenge message by himself, and a random challenge message had to be therefore chosen out of some message distribution. But in our formalism, there is no problem for the adversary to adaptively choose the challenge message by himself, and indeed we give this stronger notion of anonymity, which all of our schemes meet.

Our contribution. In this chapter, we give a new formalism for an anonymous signature following the outline given in the introduction. Also, we present some examples of efficient anonymous signature schemes. We first give a generic construction out of any ordinary unforgeable signature scheme and a commitment scheme. Also, we show that the short signature scheme by Boneh and Boyen [8] can be naturally regarded as such a secure anonymous signature scheme according toour formalism, with essentially no modification.

3.2 Related work

The notion of anonymous signature was first formalized by Yang et al. in [33], and explored further by Fischlin in [18]. Our work revisits this notion, and provides an alternative formalism.

Zhang and Imai [34] proposed a very similar approach as ours. Their idea is to define 'strong anonymous signature', which maintains anonymity even when there is not much uncertainty in the message distribution. Though their definition of strong anonymity is essentially the same as our anonymity, they do not discuss unpretendability, which we argue is central to the notion of anonymous signatures.

There are pre-existing security notions closely related to unpretendability; Menezes and Smart [25] studied security against the key substitution attack for signature schemes, where an adversary produces a public key (and the corresponding secret key, in their formulation) to claim the ownership of a message-signature pair generated by someone else. Also Hu et al. [22] introduced key replacement attack, which is similar notion in context of certificateless signatures.

Galbraith and Mao [19] introduced the notion of anonymity to undeniable and confirmer signatures. Our definition of anonymity of an anonymous signature is similar to theirs, and also the fact that the signer has to provide the verification token later to let others verify the signature looks similar to the case of undeniable signatures. But an anonymous signature is not an undeniable signature; anyone who obtained the token of the signature can in fact let others verify the signature, without the involvement of the signer. In general, an anonymous signature is much simpler than an anonymous undeniable signature.

Also, there are notions of anonymity in group and ring signatures, but these are associated with anonymity within the group or ring in question. On the other hand, the anonymous signature in our formalism or in previous formalism is essentially a conventional signature scheme with some additional properties.

3.3 Definitions

3.3.1 Notations and conventions

We denote by $v \leftarrow A(x, y, z, \ldots)$ the operation of running a randomized or deterministic algorithm $A(x, y, z, \ldots)$ and storing the output to the variable v. If X is a set, then $v \overset{\mathrm{R}}{\leftarrow} X$ denotes the operation of choosing an element v of X according to the uniform random distribution on X. Unless stated otherwise, all algorithms in this chapter are probabilistic polynomial-time algorithms.

Remark 3.3.1. We define only the advantage of an adversary in a security experiment, and would not explicitly define the security notion itself. Informally, a signature scheme Σ is unforgeable if for any efficient adversary $\mathcal{A}$ in an experiment $\mathbf{Expt}_{\Sigma}^{\mathsf{expt}}(\mathcal{A})$, the advantage of the adversary $\mathbf{Adv}_{\Sigma,\mathcal{A}}^{\mathsf{expt}}$ is negligible. But definitions of 'efficient' or 'negligible' are not explicitly defined here and will depend on particular applications.

3.3.2 Anonymous signature

We define an *anonymous signature* Σ as a quadruple of algorithms $\Sigma = (\mathrm{Par}, \mathrm{Gen}, \mathrm{Sig}, \mathrm{Vf})$, where the parameter generation algorithm Par() outputs a common parameter $P \leftarrow \mathrm{Par}(1^k)$ using security parameter k, the key generation algorithm Gen() outputs a key pair $(pk, sk) \leftarrow \mathrm{Gen}(P)$ given the common parameter P as input, signature generation algorithm Sig() outputs a pair of a signature and a verification token $\tilde{\sigma} = (\sigma, \tau) \leftarrow \mathrm{Sig}(sk, m)$ with respect to the secret key sk and a message $m \in \{0,1\}^*$, and the deterministic, signature verification algorithm $\mathrm{Vf}(pk, m, \sigma, \tau)$ outputs true or false.

The common parameter P contains the security parameter k itself, and additionally contain other public information common to all users of the system, for example description of cryptographic groups used in the signature scheme.

For consistency, we require the following:

$$\mathrm{Vf}(pk, m, \mathrm{Sig}(sk, m)) = \mathsf{true},$$

for $P \leftarrow \mathrm{Par}(1^k)$, $(pk, sk) \leftarrow \mathrm{Gen}(P)$, and for any $m \in \{0,1\}^*$.

3.3.3 Unforgeability

We say that $\Sigma = (\mathrm{Par}, \mathrm{Gen}, \mathrm{Sig}, \mathrm{Vf})$ is *unforgeable*, if for any adversary $\mathcal{A}$, the advantage

$$\mathbf{Adv}^{\textsf{uf-cma}}_{\Sigma,\mathcal{A}}(k) \overset{\text{def}}{=} \mathbf{Pr}\left[\mathbf{Expt}^{\textsf{uf-cma}}_{\Sigma,\mathcal{A}}(k) = \textsf{true}\right]$$

is negligible in the experiment $\mathbf{Expt}^{\textsf{uf-cma}}_{\Sigma,\mathcal{A}}$ defined as follows:

```
Experiment $\mathbf{Expt}^{\textsf{uf-cma}}_{\Sigma,\mathcal{A}}(k)$
    $P \leftarrow \mathrm{Par}(1^k)$
    $(pk, sk) \leftarrow \mathrm{Gen}(P)$
    $(m^*, \sigma^*, \tau^*) \leftarrow \mathcal{A}^{\mathrm{Sig}(sk,\cdot)}(pk)$
    **return** $\mathrm{Vf}(pk, m^*, \sigma^*, \tau^*)$
```

where the adversary $\mathcal{A}$ has access to the signing oracle $\mathrm{Sig}(sk, \cdot)$ with respect to the secret key sk with the requirement that $\mathcal{A}$ is not allowed to query the signing oracle with m^*.

Similarly, we say that Σ is *strongly unforgeable*, if for any adversary $\mathcal{A}$, the advantage

$$\mathbf{Adv}^{\textsf{suf-cma}}_{\Sigma,\mathcal{A}}(k) \overset{\text{def}}{=} \mathbf{Pr}\left[\mathbf{Expt}^{\textsf{suf-cma}}_{\Sigma,\mathcal{A}}(k) = \textsf{true}\right]$$

is negligible in the experiment $\mathbf{Expt}^{\textsf{suf-cma}}_{\Sigma,\mathcal{A}}$, which is identical to $\mathbf{Expt}^{\textsf{uf-cma}}_{\Sigma,\mathcal{A}}$, except that we require $\mathcal{A}$ not to have received (σ^*, τ^*) as an answer to any query of form m^* to the signing oracle.

3.3.4 Anonymity

Consider an adversary which is a pair of algorithms $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$. Let st be the state information which $\mathcal{A}_1$ passes to $\mathcal{A}_2$. We say that $\Sigma = (\mathrm{Par}, \mathrm{Gen}, \mathrm{Sig}, \mathrm{Vf})$ is *anonymous*, if for any such $\mathcal{A}$, the advantage

$$\mathbf{Adv}^{\textsf{anon}}_{\Sigma,\mathcal{A}}(k) \overset{\text{def}}{=} \left|\mathbf{Pr}[\mathbf{Expt}^{\textsf{anon-1}}_{\Sigma,\mathcal{A}}(k) = 1] - \mathbf{Pr}[\mathbf{Expt}^{\textsf{anon-0}}_{\Sigma,\mathcal{A}}(k) = 1]\right|$$

is negligible, where experiments $\mathbf{Expt}^{\textsf{anon-}b}_{\Sigma,\mathcal{A}}$ $(b = 0, 1)$ are defined as follows:

```
Experiment $\mathbf{Expt}^{\mathsf{anon}\text{-}b}_{\Sigma,\mathcal{A}}(k)$
    $P \leftarrow \mathrm{Par}(1^k)$
    $(pk_0, sk_0) \leftarrow \mathrm{Gen}(P)$; $(pk_1, sk_1) \leftarrow \mathrm{Gen}(P)$
    $(m^*, st) \leftarrow \mathcal{A}_1^{\mathrm{Sig}(sk_0,\cdot),\mathrm{Sig}(sk_1,\cdot)}(pk_0, pk_1)$
    $(\sigma^*, \tau^*) \leftarrow \mathrm{Sig}(sk_b, m^*)$
    $b' \leftarrow \mathcal{A}_2^{\mathrm{Sig}(sk_0,\cdot),\mathrm{Sig}(sk_1,\cdot)}(\sigma^*, st)$
    return $b'$
```

We call Σ anonymous with respect to *full key exposure*, when the advantage of any adversary is still negligible even if the adversary also gets the secret keys sk_0, sk_1 as additional input. We denote by $\mathbf{Adv}^{\mathsf{anon\text{-}fke}}_{\Sigma,\mathcal{A}}(k)$ the advantage of an adversary in the anonymity experiment with full key exposure.

3.3.5 Unpretendability

We say that $\Sigma = (\mathrm{Par}, \mathrm{Gen}, \mathrm{Sig}, \mathrm{Vf})$ is *unpretendable*, if for any adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$, the advantage

$$\mathbf{Adv}^{\mathsf{up}}_{\Sigma,\mathcal{A}}(k) \overset{\text{def}}{=} \mathbf{Pr}\left[\mathbf{Expt}^{\mathsf{up}}_{\Sigma,\mathcal{A}}(k) = \mathsf{true}\right]$$

is negligible in the experiment $\mathbf{Expt}^{\mathsf{up}}_{\Sigma,\mathcal{A}}$ defined as follows:

```
Experiment $\mathbf{Expt}^{\mathsf{up}}_{\Sigma,\mathcal{A}}(k)$
    $P \leftarrow \mathrm{Par}(1^k)$
    $(pk^*, sk^*) \leftarrow \mathrm{Gen}(P)$
    $(m^*, st) \leftarrow \mathcal{A}_1^{\mathrm{Sig}(sk^*,\cdot)}(pk^*)$
    $(\sigma^*, \tau^*) \leftarrow \mathrm{Sig}(sk^*, m^*)$
    $(\tau, pk) \leftarrow \mathcal{A}_2^{\mathrm{Sig}(sk^*,\cdot)}(\sigma^*, \tau^*, st)$
    return $\mathrm{Vf}(pk, m^*, \sigma^*, \tau) \wedge (pk \neq pk^*)$
```

Intuitively, the adversary is trying to claim the authorship of (m^*, σ^*) which is signed by the target secret key sk^*. The adversary tries to produce an appropriate τ so that the signature is verified with his own public key pk, which could be freshly chosen, and the definition guarantees that the probability of success for this attempt is negligible.

Also, we define a weaker version of unpretendability: we say that $\Sigma = (\mathrm{Par}, \mathrm{Gen}, \mathrm{Sig}, \mathrm{Vf})$ is *weakly unpretendable*, if for any adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2, \mathcal{A}_3)$, the advantage

$$\mathbf{Adv}^{\mathsf{wup}}_{\Sigma,\mathcal{A}}(k) \overset{\text{def}}{=} \mathbf{Pr}\left[\mathbf{Expt}^{\mathsf{wup}}_{\Sigma,\mathcal{A}}(k) = \mathsf{true}\right]$$

is negligible in the experiment $\mathbf{Expt}^{\mathsf{wup}}_{\Sigma,\mathcal{A}}$ defined as follows:

```
Experiment $\mathbf{Expt}^{\mathsf{wup}}_{\Sigma,\mathcal{A}}(k)$
    $P \leftarrow \mathrm{Par}(1^k)$
    $(pk, st) \leftarrow \mathcal{A}_1(P)$
    $(pk^*, sk^*) \leftarrow \mathrm{Gen}(P)$
    $(m^*, st') \leftarrow \mathcal{A}_2^{\mathsf{Sig}(sk^*,\cdot)}(pk^*, st)$
    $(\sigma^*, \tau^*) \leftarrow \mathrm{Sig}(sk^*, m^*)$
    $\tau \leftarrow \mathcal{A}_3^{\mathsf{Sig}(sk^*,\cdot)}(\sigma^*, \tau^*, st')$
    **return** $\mathrm{Vf}(pk, m^*, \sigma^*, \tau)$
```

The difference between unpretendability and weak unpretendability is that in unpretendability, the adversary is allowed to choose his public key adaptively which is not allowed in the case of weak unpretendability. The notion of weak unpretendability is applicable for example in situations where there is a trustable PKI under which every party registers his public key to his identity, possibly timestamped and with proof of secret key possession; in such cases, the adversary cannot adaptively choose his public key after seeing the signature, and claim the ownership under the fresh key/identity. Many applications like anonymous paper review or anonymous auction could fall into this category, but this depends on how the public keys are managed. The unpretendability is stronger in that the adversary cannot claim the ownership of the signature even when he is allowed to freshly create a new public key.

As in the case of anonymity, we say that Σ is (weakly) unpretendable with respect to full key exposure if the advantage of any adversary is negligible even if the adversary gets the target secret key sk^* as additional input. We denote the advantage of an adversary in the (weak) unpretendability experiment with full key exposure by $(\mathbf{Adv}^{\mathsf{wup\text{-}fke}}_{\Sigma,\mathcal{A}}(k))$ $\mathbf{Adv}^{\mathsf{up\text{-}fke}}_{\Sigma,\mathcal{A}}(k)$.

3.3.6 Security of an anonymous signature

Suppose that $\Sigma = (\mathrm{Par}, \mathrm{Gen}, \mathrm{Sig}, \mathrm{Vf})$ is an anonymous signature scheme. We say that Σ is a *secure* anonymous signature, if Σ is unforgeable, anonymous, and at least weakly unpretendable.

We emphasize that the unpretendability is a crucial property that an anonymous signature should have. We have already demonstrated that if an anonymous signature is not unpretendable, then it cannot be used for some of the suggested applications like anonymous paper review. Here, we show an example of an anonymous signature which is unforgeable, anonymous, but not weakly unpretendable.

Suppose $\Sigma = (\mathrm{Par}, \mathrm{Gen}, \mathrm{Sig}, \mathrm{Vf})$ is an ordinary unforgeable signature scheme. We then construct an anonymous signature scheme $\Sigma' = (\mathrm{Par}', \mathrm{Gen}', \mathrm{Sig}', \mathrm{Vf}')$ as follows: $\mathrm{Par}'(1^k)$ is the same as $\mathrm{Par}(1^k)$, $\mathrm{Gen}'(P)$ is the same as $\mathrm{Gen}(P)$. $\mathrm{Sig}'(sk, m)$ is defined as

$$\mathrm{Sig}'(sk, m) = (\sigma, \tau) \stackrel{\text{def}}{=} (\mathrm{Sig}(sk, m) \oplus \tau, \tau)$$

where the verification token τ is a bitstring of the same bit-length as the signature $\mathrm{Sig}(sk, m)$ and is chosen uniform randomly. Finally, $\mathrm{Vf}'(sk, m, \sigma, \tau)$ is defined as

$$\mathrm{Vf}'(pk, m, \sigma, \tau) \stackrel{\text{def}}{=} \mathrm{Vf}(pk, m, \sigma \oplus \tau).$$

It is clear that the anonymous signature Σ' is both unforgeable and anonymous; since the signature $\mathrm{Sig}(sk, m)$ is masked with random bitstring τ in $\mathrm{Sig}'(sk, m)$, the adversary has no information about the signature. Only later when τ is revealed, the signature σ is revealed and signature can be verified. This is equivalent to deferring the signing to the last minute when the token τ has to be revealed. The scheme is therefore unforgeable, and unless τ is revealed, the signer anonymity is guaranteed.

But, it is trivial to break unpretendability of this scheme; if $(m^*, \sigma^* = \mathrm{Sig}(sk^*, m^*) \oplus \tau^*)$ is given, the adversary may compute $\mathrm{Sig}(sk, m^*)$ using his own secret key sk, and compute τ as

$$\tau \stackrel{\text{def}}{=} \mathrm{Sig}(sk, m^*) \oplus \sigma^*.$$

Then,

$$\mathrm{Vf}'(pk, m^*, \sigma^*, \tau) = \mathrm{Vf}(pk, m^*, \sigma^* \oplus \tau) = \mathrm{Vf}(pk, m^*, \mathrm{Sig}(sk, m^*)) = \mathsf{true}.$$

3.3.7 Commitment schemes

A *commitment scheme* Γ consists of a pair of algorithms $(\mathrm{Com}, \mathrm{CVf})$ satisfying the following:

Correctness: For any bitstring $s \in \{0,1\}^*$,

$$\mathbf{Pr}[(\mathsf{com}, \mathsf{dec}) \leftarrow \mathrm{Com}(s) : \mathrm{CVf}(\mathsf{com}, \mathsf{dec}, s) = \mathsf{true}] = 1\,.$$

Hiding: For any adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$, (the adversary is a pair of algorithms), the *hiding* advantage with respect to Γ is defined as

$$\mathbf{Adv}^{\mathsf{hide}}_{\Gamma,\mathcal{A}}(k) \overset{\text{def}}{=} \left|\mathbf{Pr}[\mathbf{Expt}^{\mathsf{hide}\text{-}1}_{\Gamma,\mathcal{A}}(k) = 1] - \mathbf{Pr}[\mathbf{Expt}^{\mathsf{hide}\text{-}0}_{\Gamma,\mathcal{A}}(k) = 1]\right|$$

is negligible where experiments $\mathbf{Expt}^{\mathsf{hide}\text{-}b}_{\Gamma,\mathcal{A}}$ $(b = 0, 1)$ are defined as follows:

```
Experiment $\mathbf{Expt}^{\mathsf{hide}\text{-}b}_{\Gamma,\mathcal{A}}(k)$
    $(s_0, s_1, st) \leftarrow \mathcal{A}_1(1^k)$
    $(\mathsf{com}, \mathsf{dec}) \leftarrow \mathrm{Com}(s_b)$
    $b' \leftarrow \mathcal{A}_2(\mathsf{com}, st)$
    return $b'$
```

Also, we require the adversary $\mathcal{A}$ to output s_0, s_1 of the same bitlength: $|s_0| = |s_1|$.

Binding: For any adversary $\mathcal{A}$, the *binding* advantage with respect to Γ is defined as

$$\mathbf{Adv}^{\mathsf{bind}}_{\Gamma,\mathcal{A}}(k) \overset{\text{def}}{=} \mathbf{Pr}\left[\mathbf{Expt}^{\mathsf{bind}}_{\Gamma,\mathcal{A}}(k) = \mathsf{true}\right]$$

is negligible in the experiment $\mathbf{Expt}^{\mathsf{bind}}_{\Gamma,\mathcal{A}}$ defined as follows:

```
Experiment $\mathbf{Expt}^{\mathsf{bind}}_{\Gamma,\mathcal{A}}(k)$
    $(\mathsf{com}, \mathsf{dec}, s, \mathsf{dec}', s') \leftarrow \mathcal{A}(1^k)$
    $p \leftarrow \mathrm{CVf}(\mathsf{com}, \mathsf{dec}, s)$
    $p' \leftarrow \mathrm{CVf}(\mathsf{com}, \mathsf{dec}', s')$
    return $p \wedge p' \wedge (s \neq s')$
```

3.4 Secure anonymous signature schemes

In this section, first we show how to construct an anonymous signature scheme generically from any ordinary unforgeable signature scheme. Then, we show that the short signature scheme of Boneh and Boyen [8] can be naturally considered as a secure anonymous signature according to our formalism, with essentially no modification. To be precise, it is a weakly unpretendable anonymous signature.

3.4.1 Generic construction

Here we present a generic construction of an anonymous signature scheme using an ordinary signature scheme and a commitment scheme. It is required that for any security parameter k, the signature scheme is unforgeable, and the public key size and the signature size are constant for all users.

Let $\Sigma = (\mathrm{Par}, \mathrm{Gen}, \mathrm{Sig}, \mathrm{Vf})$ be a signature scheme. When k is the security parameter, let $l_p(k)$ and $l_s(k)$ be the bit length of a public key pk and the bit length of a signature respectively. Let $(\mathrm{Com}, \mathrm{CVf})$ be a commitment scheme. We construct an anonymous signature $\Sigma' = (\mathrm{Par}', \mathrm{Gen}', \mathrm{Sig}', \mathrm{Vf}')$ using these as follows:

```
**function** $\mathrm{Par}'(1^k)$
    $P \leftarrow \mathrm{Par}(1^k)$
    **return** $P$
**function** $\mathrm{Gen}'(P)$
    $(pk, sk) \leftarrow \mathrm{Gen}(P)$
    $pk' \leftarrow pk$
    $sk' \leftarrow sk \| pk$
    **return** $(pk', sk')$
**function** $\mathrm{Sig}'(sk', m)$
    Parse $sk'$ as $sk \| pk$
    $\sigma' \leftarrow \mathrm{Sig}(sk, m)$
    $(\mathsf{com}, \mathsf{dec}) \leftarrow \mathrm{Com}(pk \| \sigma')$
    $\sigma \leftarrow \mathsf{com}$; $\tau \leftarrow \mathsf{dec} \| \sigma'$
    **return** $(\sigma, \tau)$
**function** $\mathrm{Vf}'(pk', m, \sigma, \tau)$
    Parse $\tau$ as $\tau_1 \| \tau_2$
    **return** $\mathrm{CVf}(\sigma, \tau_1, pk' \| \tau_2) \wedge \mathrm{Vf}(pk', m, \tau_2)$
```

3.4.2 Security of the generic construction

Theorem 3.4.1. Given an ordinary signature scheme Σ, consider the scheme Σ' defined in the previous subsection. If Σ is unforgeable, then Σ' is a secure unforgeable anonymous signature. Moreover, Σ' is both anonymous and unpretendable with respect to full key exposure.

Proof. First, we prove the unforgeability of Σ'.

Suppose that $\mathcal{A}$ is an adversary attacking the unforgeability of Σ'. Then using $\mathcal{A}$, we construct an adversary $\mathcal{B}$ which attacks the unforgeability of Σ, and satisfying

$$\mathbf{Adv}^{\mathsf{uf\text{-}cma}}_{\Sigma',\mathcal{A}}(k) \leq \mathbf{Adv}^{\mathsf{uf\text{-}cma}}_{\Sigma,\mathcal{B}}(k)\,.$$

The adversary $\mathcal{B}$ is given a public key pk of Σ, and the corresponding signing oracle $\mathrm{Sig}(sk,\cdot)$. $\mathcal{B}$ sets $pk' = pk$, and gives it to $\mathcal{A}$ and answers the signing query of $\mathcal{A}$ as follows: for signing query of m, $\mathcal{B}$ calls its own signing oracle with query m to obtain σ', computes $(\mathsf{com},\mathsf{dec}) \leftarrow \mathrm{Com}(pk,\sigma')$ and returns $(\sigma = \mathsf{com}, \tau = \mathsf{dec}\|\sigma')$ to $\mathcal{A}$. Note that this simulation of the unforgeability experiment for $\mathcal{A}$ by $\mathcal{B}$ is perfectly done according to the description of Σ'.

Suppose that $\mathcal{A}$ halts with output (m^*,σ^*,τ^*). Then $\mathcal{B}$ parses τ^* as $\tau_1\|\tau_2$, and halts with output (m^*,τ_2).

Whenever the output (m^*,σ^*,τ^*) of $\mathcal{A}$ is a successful forgery for Σ', then $\mathcal{B}$ outputs a successful forgery (m^*,τ_2) for Σ since from the definition of Vf', $\mathrm{Vf}'(pk',m^*,\sigma^*,\tau^*) = \mathsf{true}$ holds only if $\mathrm{Vf}(pk,m^*,\tau_2) = \mathsf{true}$ holds. This proves the claimed inequality.

Next, we show that Σ' satisfies anonymity with respect to full key exposure. Suppose that $\mathcal{A} = (\mathcal{A}_1,\mathcal{A}_2)$ is an adversary attacking anonymity of Σ'. Using $\mathcal{A}$, we construct $\mathcal{B}$ attacking the hiding property of the commitment scheme Γ, satisfying

$$\mathbf{Adv}^{\mathsf{anon\text{-}fke}}_{\Sigma',\mathcal{A}}(k) \leq \mathbf{Adv}^{\mathsf{hide}}_{\Gamma,\mathcal{B}}(k)\,.$$

Consider the experiment $\mathbf{Expt}^{\mathsf{hide}\text{-}b}_{\Gamma,\mathcal{B}}$ with respect to this adversary $\mathcal{B}$. Given the security parameter k, $\mathcal{B}$ generates common parameters of Σ' and two key pairs (pk'_0, sk'_0) and (pk'_1, sk'_1). $\mathcal{B}$ then runs $\mathcal{A}_1(pk'_0, pk'_1, sk'_0, sk'_1)$ to obtain an output (m^*, st) and gives $s_0 = pk_0\|\mathrm{Sig}(sk_0,m^*)$ and $s_1 = pk_1\|\mathrm{Sig}(sk_1,m^*)$ to the challenger. The challenger computes $(\mathsf{com}, \mathsf{dec}) \leftarrow \mathrm{Com}(s_b)$, and gives $\sigma^* = \mathsf{com}$ to $\mathcal{B}$. $\mathcal{B}$ now runs $\mathcal{A}_2(\sigma^*, st)$ to obtain an output b' and then halts with output b'.

Note that this simulation of the full-key exposure anonymity experiment for $\mathcal{A}$ by $\mathcal{B}$ is perfect, and the output of $\mathcal{B}$ is the same as the output of $\mathcal{A}$. Hence, $\mathbf{Pr}[\mathbf{Expt}^{\mathsf{hide}\text{-}b}_{\Gamma,\mathcal{B}}(k)] = \mathbf{Pr}[\mathbf{Expt}^{\mathsf{anon}\text{-}\mathsf{fke}\text{-}b}_{\Sigma',\mathcal{A}}(k)]$, for $b = 0, 1$. Therefore,

$$\begin{aligned}\mathbf{Adv}^{\mathsf{anon}\text{-}\mathsf{fke}}_{\Sigma',\mathcal{A}}(k) &= \left|\mathbf{Pr}[\mathbf{Expt}^{\mathsf{anon}\text{-}\mathsf{fke}\text{-}1}_{\Sigma',\mathcal{A}}(k) = 1] - \mathbf{Pr}[\mathbf{Expt}^{\mathsf{anon}\text{-}\mathsf{fke}\text{-}0}_{\Sigma',\mathcal{A}}(k) = 1]\right| \\ &= \left|\mathbf{Pr}[\mathbf{Expt}^{\mathsf{hide}\text{-}1}_{\Gamma,\mathcal{B}}(k) = 1] - \mathbf{Pr}[\mathbf{Expt}^{\mathsf{hide}\text{-}0}_{\Gamma,\mathcal{B}}(k) = 1]\right| \\ &= \mathbf{Adv}^{\mathsf{hide}}_{\Gamma,\mathcal{B}}(k).\end{aligned}$$

Finally, we show that Σ' satisfies unpretendability with respect to full key exposure. Suppose that $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ is an adversary attacking unpretendability of Σ'. Using $\mathcal{A}$, we construct an adversary $\mathcal{B}$ attacking the binding property of the commitment scheme Γ, satisfying

$$\mathbf{Adv}^{\mathsf{up}\text{-}\mathsf{fke}}_{\Sigma',\mathcal{A}}(k) \leq \mathbf{Adv}^{\mathsf{bind}}_{\Gamma,\mathcal{B}}(k)\,.$$

Given the security parameter k, $\mathcal{B}$ generates common parameters and a key pair (pk'^*, sk'^*), and runs $\mathcal{A}_1(pk'^*, sk'^*)$ to obtain an output (m^*, st). $\mathcal{B}$ then computes $(\sigma^*, \tau^*) \leftarrow \mathrm{Sig}'(sk'^*, m^*)$, and runs $\mathcal{A}_2(\sigma^*, \tau^*, st)$ to obtain an output (τ, pk'). Then $\mathcal{B}$ parses τ as $\tau_1 \| \tau_2$ and τ^* as $\tau_1^* \| \tau_2^*$ and halts with output $(\sigma^*, \tau_1^*, pk'^* \| \tau_2^*, \tau_1, pk' \| \tau_2)$. This simulation of the full-key exposure unpretendability experiment for $\mathcal{A}$ by $\mathcal{B}$ is perfect.

We claim that, in the above simulation, whenever $\mathcal{A}$ succeeds at breaking the unpretendability of Σ', that is, $\mathrm{Vf}'(pk', m^*, \sigma^*, \tau) = \mathsf{true}$ and $pk' \neq pk'^*$, then $\mathcal{B}$ also succeeds in breaking the binding property of Γ. From the definition of Vf', in order that $\mathrm{Vf}'(pk', m^*, \sigma^*, \tau) = \mathsf{true}$, it is necessary that $\mathrm{CVf}(\sigma^*, \tau_1, pk' \| \tau_2)$ is also true. Moreover, since $(\sigma^*, \tau^*) = \mathrm{Sig}'(sk'^*, m^*)$, also $\mathrm{Vf}'(pk'^*, m^*, \sigma^*, \tau^*) = \mathsf{true}$ holds, and from this it follows that $\mathrm{CVf}(\sigma^*, \tau_1^*, pk'^* \| \tau_2^*) = \mathsf{true}$. Now, $pk'^* \neq pk'$ so that $pk'^* \| \tau_2^* \neq pk' \| \tau_2$ and hence $\mathcal{B}$ has successfully violated the binding property of Γ. $\square$

$\square$

3.4.3 Boneh-Boyen short signature

In this section, we give a brief description of the Boneh-Boyen signature scheme [8] for completeness.

Parameter generation A bilinear group $(\mathbb{G}_1, \mathbb{G}_2)$ with a pairing $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$, where $|\mathbb{G}_1| = |\mathbb{G}_2| = |\mathbb{G}_T| = p$ for some prime p, is chosen. The message space is $\mathbb{Z}_p$, which gives no essential problem since the domain can be extended by using a (target) collision resistant hash function.

Key generation Key generation algorithm chooses random generators g_1 and g_2 of $\mathbb{G}_1$ and $\mathbb{G}_2$, respectively, and chooses $x, y \overset{\text{R}}{\leftarrow} \mathbb{Z}_p^*$, computes $u \leftarrow g_2^x \in \mathbb{G}_2$, $v \leftarrow g_2^y \in \mathbb{G}_2$. Then, $pk \overset{\text{def}}{=} (g_1, g_2, u, v)$, and $sk \overset{\text{def}}{=} (g_1, x, y)$.

Signing For a secret key (g_1, x, y) and a message $m \in \mathbb{Z}_p$, the signing algorithm chooses $\tau \overset{\text{R}}{\leftarrow} \mathbb{Z}_p \setminus \{-\frac{x+m}{y}\}$, and computes $\sigma \leftarrow g_1^{1/(x+m+y\tau)} \in \mathbb{G}_1$. Then the signature is the pair (σ, τ).

Verification For a public key (g_1, g_2, u, v), a message m, and a signature (σ, τ), the verification can be done by checking whether $e(\sigma, u \cdot g_2^m \cdot v^\tau) = e(g_1, g_2)$.

3.4.4 Security of Boneh-Boyen as an anonymous signature

The Boneh-Boyen short signature can be naturally considered as an anonymous signature, by regarding τ in $(\sigma = g_1^{1/(x+m+y\tau)}, \tau)$ as the verification token. To be precise, because τ should not be equal to $-(x+m)/y$ modulo p, we need to make slight modifications both to the signature scheme and to the formalism itself. For example, instead of choosing τ uniformly from $\mathbb{Z}_p \setminus \{-(x+m)/y\}$, τ may be chosen uniformly from $\mathbb{Z}_p$, and the signing algorithm may be allowed to fail in the negligible possibility that $\tau = -(x+m)/y$.

Then, the Boneh-Boyen short signature scheme becomes a secure anonymous signature scheme; we show that it is strongly unforgeable, anonymous with full key exposure, and *weakly* unpretendable with full key exposure.

Strong unforgeability

Because our definition of strong unforgeability for anonymous signatures is identical to the ordinary definition of strong unforgeability, the proof of Boneh and Boyen for the strong unforgeability of the short signature scheme is directly applicable. Their proof is based on the SDH assumption on bilinear groups $(\mathbb{G}_1, \mathbb{G}_2)$.

Anonymity with full key exposure

For a message $m \in \mathbb{Z}_p$ chosen by the adversary, consider the distribution of the signature σ, where $\sigma = g_1^{1/(x+m+y\tau)}$, for uniformly chosen token $\tau \overset{\text{R}}{\leftarrow} \mathbb{Z}_p$, when the secret key (g_1, x, y) is given to the adversary. Then, even conditioned on g_1, x, m, and y, $1/(x + m + y\tau)$ has uniform distribution on $\mathbb{Z}_p^* \cup \{\perp\}$, and σ has uniform distribution on $(\mathbb{G}_1 \setminus \{1\}) \cup \{\perp\}$. Since this is true for any secret key (g_1, x, y), we conclude that the Boneh-Boyen short signature scheme is anonymous with full key exposure.

Weak unpretendability with full key exposure

We will prove weak unpretendability of Boneh-Boyen signature with full key exposure, under the following assumption on the bilinear groups $(\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T)$ which we call 'adversarial pairing inversion assumption':

> With respect to any adversarially chosen $h \in \mathbb{G}_T \setminus \{1\}$, it is infeasible to find $X \in \mathbb{G}_2$ satisfying $e(g, X) = h$, for $g \overset{\text{R}}{\leftarrow} \mathbb{G}_1 \setminus \{1\}$.

It is a nonstandard variant of pairing inversion problem; it is known that some versions of pairing inversion problem is as hard as the computational Diffie-Hellman problem [15, 30], but here h is allowed to be chosen by the adversary, and it is not known whether this assumption can be derived from more traditional assumptions. Note also that this is an interactive assumption. But, the adversarial choice of h does not seem to allow any obvious attacks, and as a partial justification of the assumption, it can be shown that this assumption holds in generic bilinear groups.

Let $\mathcal{A}$ be an adversary of weak unpretendability of the Boneh-Boyen signature with key exposure. Using $\mathcal{A}$, we construct the adversary $\mathcal{B}$ of the adversarial pairing inversion problem. The challenger of the adversarial pairing inversion problem selects the description of the bilinear groups, and passes it to $\mathcal{B}$. $\mathcal{B}$ then runs $\mathcal{A}$ with the same input. After obtaining the public key output $(g_1, g_2, u, v) \in \mathbb{G}_1 \times \mathbb{G}_2 \times \mathbb{G}_2 \times \mathbb{G}_2$ of $\mathcal{A}$, $\mathcal{B}$ outputs $h \leftarrow e(g_1, g_2)$ as his chosen instance for the adversarial pairing inversion to the challenger.

Then, the challenger sends $\mathcal{B}$ a random $g \overset{\text{R}}{\leftarrow} \mathbb{G}_1 \setminus \{1\}$. $\mathcal{B}$ defines $g_1^* \overset{\text{def}}{=} g$, and randomly chooses $g_2^* \overset{\text{R}}{\leftarrow} \mathbb{G}_2 \setminus \{1\}$, x^*, $y^* \overset{\text{R}}{\leftarrow} \mathbb{Z}_p$, and sends g_1^*, g_2^*, x^*, y^* to $\mathcal{A}$. $\mathcal{A}$ will then output the challenge message m^*. $\mathcal{B}$ randomly chooses $\tau^* \overset{\text{R}}{\leftarrow} \mathbb{Z}_p$, computes $\sigma^* \leftarrow (g_1^*)^{1/(x^*+m^*+y^*\tau^*)}$, and sends (σ^*, τ^*) to $\mathcal{A}$. $\mathcal{A}$ will eventually halt with some τ. Using τ, $\mathcal{B}$ outputs X, where X is defined as

$$X \overset{\text{def}}{=} (ug_2^{m^*}v^\tau)^{1/(x^*+m^*+y^*\tau^*)}.$$

In the above, $\mathcal{B}$ provides perfect simulation for $\mathcal{A}$. Suppose that the attack of $\mathcal{A}$ is successful: then

$$e(g_1, g_2) = e(\sigma^*, ug_2^{m^*}v^\tau)$$

holds. Since $\sigma^* = (g_1^*)^{1/(x^*+m^*+y^*\tau^*)} = g^{1/(x^*+m^*+y^*\tau^*)}$, the above equation is equivalent to $e(g, X) = e(g_1, g_2) = h$. Hence $\mathcal{B}$ solves the pairing inversion whenever the weak unpretendability attack of $\mathcal{A}$ is successful.

On unpretendability of Boneh-Boyen

The Boneh-Boyen signature scheme satisfies weak unpretendability with full key exposure. However it is *not* unpretendable as it is easy to break unpretendability when the adversary is allowed to choose his public key adaptively.

Acknowledgements

The work in Chapter 3 was done jointly with Dr. Aaram Yun, Department of Computer Science and Engineering, University of Minnesota, who was supported by the US National Science Foundation grant no. CCF 0621462. The work resulted in a paper [28] published in ProvSec 2009. We thank anonymous reviewers for many constructive and helpful comments. Especially, we revised notions of the unpretendability, following criticisms on our previous definition. Also, a reviewer suggested the possibility of commitment-based generic construction of anonymous signatures. In our original manuscript on ePrint archive [29], we had presented a pseudorandom generator based construction which was a special case of commitment based construction, but we generalized it based on any commitment scheme, following the reviewer's suggestion.

Bibliography

[1] Michel Abdalla, Mihir Bellare, Dario Catalano, Eike Kiltz, Tadayoshi Kohno, Tanja Lange, John Malone-Lee, Gregory Neven, Pascal Paillier, and Haixia Shi. Searchable encryption revisited: Consistency properties, relation to anonymous ibe, and extensions. In Victor Shoup, editor, *CRYPTO*, volume 3621 of *Lecture Notes in Computer Science*, pages 205–222. Springer, 2005.

[2] Giuseppe Ateniese and Paolo Gasti. Universally anonymous ibe based on the quadratic residuosity assumption. In Marc Fischlin, editor, *CT-RSA*, volume 5473 of *Lecture Notes in Computer Science*, pages 32–47. Springer, 2009.

[3] Mihir Bellare, Alexandra Boldyreva, An, Desai, and David Pointcheval. Key-privacy in public-key encryption. In Colin Boyd, editor, *ASIACRYPT*, volume 2248 of *Lecture Notes in Computer Science*, pages 566–582. Springer, 2001.

[4] Mihir Bellare, Alexandra Boldyreva, Anand Desai, and David Pointcheval. Key-privacy in public-key encryption. In Colin Boyd, editor, *ASIACRYPT*, volume 2248, pages 566–582. Springer, 2001.

[5] Dan Boneh. Private communication, August 2007.

[6] Dan Boneh. Private communication, February 2007.

[7] Dan Boneh and Xavier Boyen. Secure identity based encryption without random oracles. In Matthew K. Franklin, editor, *CRYPTO*, volume 3152 of *Lecture Notes in Computer Science*, pages 443–459. Springer, 2004.

[8] Dan Boneh and Xavier Boyen. Short signatures without random oracles and the SDH assumption in bilinear groups. *Journal of Cryptology*, 21(2):149–177, 2008.

[9] Dan Boneh, Giovanni Di Crescenzo, Rafail Ostrovsky, and Giuseppe Persiano. Public key encryption with keyword search. In Christian Cachin and Jan Camenisch, editors, *EUROCRYPT*, volume 3027 of *Lecture Notes in Computer Science*, pages 506–522. Springer, 2004.

[10] Dan Boneh and Matthew K. Franklin. Identity-based encryption from the weil pairing. In Joe Kilian, editor, *CRYPTO*, volume 2139 of *Lecture Notes in Computer Science*, pages 213–229. Springer, 2001.

[11] Dan Boneh, Craig Gentry, and Michael Hamburg. Space-efficient identity based encryption without pairings. In *FOCS*, pages 647–657. IEEE Computer Society, 2007.

[12] Colin Boyd and Dong-Gook Park. Public key protocols for wireless communications. In *ICISC*, pages 47–57. Korea Institute of Information Security and Cryptology (KIISC), 1998.

[13] Xavier Boyen and Brent Waters. Anonymous hierarchical identity-based encryption (without random oracles). In Cynthia Dwork, editor, *CRYPTO*, volume 4117 of *Lecture Notes in Computer Science*, pages 290–307. Springer, 2006.

[14] Jan Camenisch and Anna Lysyanskaya. An efficient system for non-transferable anonymous credentials with optional anonymity revocation. In Birgit Pfitzmann, editor, *EUROCRYPT*, volume 2045 of *Lecture Notes in Computer Science*, pages 93–118. Springer, 2001.

[15] Jung Hee Cheon and Dong Hoon Lee. Diffie-Hellman problems and bilinear maps. Cryptology ePrint Archive, Report 2002/117, 2002.

[16] Clifford Cocks. An identity based encryption scheme based on quadratic residues. In Bahram Honary, editor, *IMA Int. Conf.*, volume 2260 of *Lecture Notes in Computer Science*, pages 360–363. Springer, 2001.

[17] Giovanni Di Crescenzo and Vishal Saraswat. Public key encryption with searchable keywords based on jacobi symbols. In K. Srinathan, C. Pandu Rangan, and Moti Yung, editors, *INDOCRYPT*, volume 4859 of *Lecture Notes in Computer Science*, pages 282–296. Springer, 2007.

[18] Marc Fischlin. Anonymous signatures made easy. In Tatsuaki Okamoto and Xiaoyun Wang, editors, *Public Key Cryptography*, volume 4450, pages 31–42. Springer, 2007.

[19] Steven D. Galbraith and Wenbo Mao. Invisibility and anonymity of undeniable and confirmer signatures. In Marc Joye, editor, *CT-RSA*, volume 2612, pages 80–97. Springer, 2003.

[20] Philippe Golle, Jessica Staddon, and Brent R. Waters. Secure conjunctive keyword search over encrypted data. In Markus Jakobsson, Moti Yung, and Jianying Zhou, editors, *ACNS*, volume 3089 of *Lecture Notes in Computer Science*, pages 31–45. Springer, 2004.

[21] Els Van Herreweghen. Secure anonymous signature-based transactions. In Frédéric Cuppens, Yves Deswarte, Dieter Gollmann, and Michael Waidner, editors, *ESORICS*, volume 1895 of *Lecture Notes in Computer Science*, pages 55–71. Springer, 2000.

[22] Bessie C. Hu, Duncan S. Wong, Zhenfeng Zhang, and Xiaotie Deng. Certificateless signature: a new security model and an improved generic construction. *Designs, Codes and Cryptography*, 42(2):109–126, 2007.

[23] H. Krawczyk. Skeme: a versatile secure key exchange mechanism for internet. In *Proceedings of the 1996 Symposium on Network and Distributed System Security (SNDSS '96)*, SNDSS '96, pages 114–, Washington, DC, USA, 1996. IEEE Computer Society.

[24] Byoungcheon Lee, Kwangjo Kim, and Joongsoo Ma. Efficient public auction with one-time registration and public verifiability. In C. Pandu Rangan and Cunsheng Ding, editors, *INDOCRYPT*, volume 2247 of *Lecture Notes in Computer Science*, pages 162–174. Springer, 2001.

[25] Alfred Menezes and Nigel P. Smart. Security of signature schemes in a multi-user setting. *Designs, Codes and Cryptography*, 33(3):261–274, 2004.

[26] Dong Jin Park, Kihyun Kim, and Pil Joong Lee. Public key encryption with conjunctive field keyword search. In Chae Hoon Lim and Moti Yung, editors, *WISA*, volume 3325 of *Lecture Notes in Computer Science*, pages 73–86. Springer, 2004.

[27] Kazue Sako. An auction protocol which hides bids of losers. In Hideki Imai and Yuliang Zheng, editors, *Public Key Cryptography*, volume 1751 of *Lecture Notes in Computer Science*, pages 422–432. Springer, 2000.

[28] Vishal Saraswat and Aaram Yun. Anonymous signatures revisited. In Josef Pieprzyk and Fangguo Zhang, editors, *ProvSec*, volume 5848 of *Lecture Notes in Computer Science*, pages 140–153. Springer, 2009.

[29] Vishal Saraswat and Aaram Yun. Anonymous signatures revisited. Cryptology ePrint Archive, Report 2009/307, 2009. `http://eprint.iacr.org/2009/307`.

[30] Takakazu Satoh. On pairing inversion problems. In Tsuyoshi Takagi, Tatsuaki Okamoto, Eiji Okamoto, and Takeshi Okamoto, editors, *Pairing*, volume 4575, pages 317–328. Springer, 2007.

[31] Brent Waters. Efficient identity-based encryption without random oracles. In Ronald Cramer, editor, *EUROCRYPT*, volume 3494 of *Lecture Notes in Computer Science*, pages 114–127. Springer, 2005.

[32] Brent R. Waters, Dirk Balfanz, Glenn Durfee, and Diana K. Smetters. Building an encrypted and searchable audit log. In *NDSS*. The Internet Society, 2004.

[33] Guomin Yang, Duncan S. Wong, Xiaotie Deng, and Huaxiong Wang. Anonymous signature schemes. In Moti Yung, Yevgeniy Dodis, Aggelos Kiayias, and Tal Malkin, editors, *Public Key Cryptography*, volume 3958, pages 347–363. Springer, 2006.

[34] Rui Zhang and Hideki Imai. Strong anonymous signatures. In Moti Yung, Peng Liu, and Dongdai Lin, editors, *Inscrypt*, volume 5487, pages 60–71. Springer, 2008.