Pairing-Based Cryptography : A Survey

Ratna Dutta, Rana Barua and Palash Sarkar
Cryptology Research Group
Stat-Math and Applied Statistics Unit
203, B. T. Road, Kolkata
India 700108
e-mail :{ ratna_r, rana, palash}@isical.ac.in

Abstract

The bilinear pairing such as Weil pairing or Tate pairing on elliptic and hyperelliptic curves have recently been found positive application in cryptography. Several ID-based cryptosystems were proposed using bilinear pairings of algebraic curves. In this survey, we have tried to cover different cryptographic protocols based on bilinear pairings which possess, to the best of our knowledge, proper security proofs in the existing security models.

1 Introduction

The concept of identity-based cryptosystem is due to Shamir [36]. Such a scheme has the property that a user's public key is an easily calculated function of his identity, while a user's private key can be calculated for him by a trusted authority, called private key generator (PKG). The ID-based public key cryptosystem can be an alternative for certificate-based public key infrastructure (PKI), especially when efficient key management and moderate security are required.

Earlier the bilinear pairings, namely Weil pairing and Tate pairing of algebraic curves were used in cryptography for the MOV attack [31] using Weil pairing and FR attack [18] using Tate pairing to reduce the discrete logarithm problem on some elliptic curves or hyperelliptic curves to the discrete logarithm problem in a finite field. In recent years, the bilinear pairings have been found positive application in cryptography [23] [7] [9] [25], [34] to construct new ID-based cryptographic primitives.

Joux [23], in 2000, showed that the Weil pairing can be used for "good" by using it in a protocol for three party one round Diffie-Hellman key aggrement. This was one of the breakthroughs in cryptography. After this, Boneh and Franklin [7] presented in Crypto 2001 an ID-based encryption scheme based on properties of bilinear pairings on elliptic curves which appears to be the first fully functioning, efficient and provably secure identity-based encryption scheme. In Asiacrypt 2001, Boneh, Lynn and Shacham proposed a basic signature scheme using pairing, the BLS [9] scheme, that has the shortest length among signature schemes in classical cryptography. Subsequently numerous cryptographic schemes based on BLS signature scheme were proposed [4] [8].

Apart from the three fundamental cryptographic primitives : encryption scheme, signature scheme and key agreement scheme, there are protocol designs for signcryption, threshold decryption, key sharing, identification scheme, chameleon hashes *etc.* The verious signature schemes have several important applications in the digital world like e-cash, e-voting, fair exchange *etc.*

Aggregate signature scheme has use in the secure Border Gateway Protocol for compressing the list of signatures on distinct messages issued by distinct parties. To enable fair exchange, verifiably encrypted signatures are used in optimistic contract signing protocols. Multisignatures can be applied to provide efficient batch verification of several signatures of the same message under different public keys. The concept of blind signatures provides anonymity of users in applications such as electronic voting, electronic payment systems *etc.* Ring signature and Group signature schemes are used to protect anonymity of a signer. The difference between these two is that there is no way to revoke the anonymity of the signer in ring signature while for group signature, the group manager can identify the signer. Proxy signatures have found numerous practical applications where delegation of rights is quite common, particularly in distributed systems, Grid Computing, mobile agent applications, distributed shared object systems and mobile communications [5]. Unique signature schemes, also known as invariant signature schemes, are desirable in cryptography and has an important application to construct verifiable random functions (VRFs). VRFs are objects that combine the properties of psudorandom functions (*i.e.* indistinguishability from random even after querying) with the verifiability property and can be viewed as a commitment to an arbitrary number of bits. Chameleon hashes have applications in constructing chameleon signatures. The recipient can verify that the signature of a certain message m is valid, but can not prove others that the signer actually signed m and not another message. These are closely related to undeniable signature [13].

Key agreement is required in situations where two or more parties want to communicate securely among themeselves. The situation where three or more parties share a secret key is often called conference keying. In this situation, the parties can securely send and receive message from each other. An adversary not having access to the secret key will not be able to decrypt the message.

Threshold cryptography approach is useful to remove single point failure. When the centralization of the power is a concern, threshold decryption can be used in particular.

The idea of signcryption scheme is to perform encryption and signature in a single logical step in order to obtain confidentiality, integrity, authentication and non-repudiation more efficiently than the sign-then-encrypt approach.

Identification scheme is another important and useful cryptographic tool where a prover interacts with a verifier to convince him of his identity.

In this paper, we have tried to survey over different cryptographic primitives and include only those schemes which have, to the best of our knowledge, concrete security proofs in the existing adversarial models. The rest of the paper is organized as follows: Section 2 briefly explains the cryptographic bilinear map and some versions of DH problems. The ID-based encryption schemes are discussed in Section 3. We describe various pairing based signature schemes in Section 4. Section 5 consists of key agreement schemes and Section 6 discusses threshold schemes using bilinear map. In Section 7, miscellaneous applications are described. Finally we conclude in Section 8.

2 Preliminaries

2.1 Cryptographic Bilinear Maps

Let G_1, G_2 be two groups of the same prime order q. We view G_1 as an additive group and G_2 as a multiplicative group. Let P be an arbitrary generator of G_1. (*aP denotes P added to itself a times*). Assume that discrete logarithm problem (DLP) is hard in both G_1 and G_2. A mapping $e : G_1^2 \to G_2$ satisfying the following properties is called a bilinear map from a cryptographic point of view :
Bilinearity : $e(aP, bQ) = e(P, Q)^{ab}$ for all $P, Q \in G_1$ and $a, b \in Z_q^*$.
Non-degeneracy : If P is a generator of G_1, then $e(P, P)$ is a generator of G_2. In other words, $e(P, P) \neq 1$.
Computable : There exists an efficient algorithm to compute $e(P, Q)$ for all $P, Q \in G_1$.
Modified Weil Pairing [7] and Tate Pairing [3], [20] are examples of cryptographic bilinear maps.

2.2 Diffie Hellman Assumptions

In this subsection we specify some versions of Diffie-Hellman problems. Consider (G_1, G_2, e) where G_1, G_2 are two cyclic subgroups of a large prime order q and $e : G_1^2 \to G_2$ is a cryptographic bilinear map. We take G_1 as an additive group and G_2 as a multiplicative group. (*By $a \in_R Z_q^*$, we mean a is randomly chosen from Z_q^*.*) A function is said to be *negligible* if it is less than $\frac{1}{m^l}$ for every fixed $l > 0$ and sufficiently large integer m.

1. **Decisional Diffie-Hellman (DDH) problem in G_1 :**
   Instance : (P, aP, bP, cP) for some $a, b, c \in Z_q^*$.
   Solution : Output yes if $c = ab \bmod q$ and output no otherwise.

   The advantage of any probabilistic, polynomial-time, 0/1-valued algorithm $\mathcal{A}$ in solving DDH problem in G_1 is defined to be :

   $$\mathsf{Adv}^{\mathsf{DDH}}_{\mathcal{A},G_1} = |\mathsf{Prob}[\mathcal{A}(P, aP, bP, cP) = 1] - \mathsf{Prob}[\mathcal{A}(P, aP, bP, abP) = 1] : a, b, c \in_R Z_q^*|.$$

   DDH problem in G_1 is easy : DDH problem in G_1 can be solved in polynomial time by verifying $e(aP, bP) = e(P, cP)$. This is the well known MOV reduction [7] : The DLP in G_1 is no harder than the DLP in G_2.

   DDH assumption : For every probabilistic, polynomial-time, 0/1-valued algorithm $\mathcal{A}$, $\mathsf{Adv}^{\mathsf{DDH}}_{\mathcal{A},G_2}$ is negligible.

2. **Computational Diffie-Hellman (CDH) problem in G_1 :**
   Instance : (P, aP, bP) for some $a, b \in Z_q^*$.
   Solution : Output abP.

   The advantage of any probabilistic, polynomial-time, 0/1-valued algorithm $\mathcal{A}$ in solving CDH problem in G_1 is defined to be :

   $$\mathsf{Adv}^{\mathsf{CDH}}_{\mathcal{A},G_1} = \mathsf{Prob}[\mathcal{A}(P, aP, bP, abP) = 1 : a, b \in_R Z_q^*].$$

   CDH assumption : For every probabilistic, polynomial-time, 0/1-valued algorithm $\mathcal{A}$, $\mathsf{Adv}^{\mathsf{CDH}}_{\mathcal{A},G_1}$ is negligible.

3. **Gap Diffie-Hellman (GDH) group :**
A prime order (q) group G is a GDH group if there exists an efficient polynomial-time algorithm $\mathcal{V}_{\mathsf{DDH}}(\ )$ which solves the DDH problem in G and there is no polynomial-time (in $|q|$) algorithm which solves the CDH problem with non-negligible probability of success. The domain of bilinear pairings provides examples of GDH groups.

4. **Bilinear Diffie-Hellman (BDH) problem in** (G_1, G_2, e) **:**
Instance : (P, aP, bP, cP) for some $a, b, c \in Z_q^*$.
Solution : Output $e(P, P)^{abc}$.

   The advantage of any probabilistic, polynomial-time, 0/1-valued algorithm $\mathcal{A}$ in solving BDH problem in (G_1, G_2, e) is defined to be :

$$\mathsf{Adv}_{\mathcal{A}}^{\mathsf{BDH}} = \mathsf{Prob}[\mathcal{A}(P, aP, bP, cP, e(P, P)^{abc}) = 1 : a, b, c \in_R Z_q^*].$$

   BDH assumption : For every probabilistic, polynomial-time, 0/1-valued algorithm $\mathcal{A}$, $\mathsf{Adv}_{\mathcal{A}}^{\mathsf{BDH}}$ is negligible.

5. **Weak Diffie-Hellman (W-DH) problem in a group** G **:**
Instance : $\langle P, Q, sP \rangle$ for $P, Q \in G$ and for some $s \in Z_q^*$.
Solution : Output sQ.

   The advantage of any probabilistic, polynomial-time, 0/1-valued algorithm $\mathcal{A}$ in solving W-DH problem in G is defined to be :

$$\mathsf{Adv}_{\mathcal{A}}^{\mathsf{W-DH}} = \mathsf{Prob}[\mathcal{A}(P, Q, sP, sQ) = 1 : s \in Z_q^*].$$

   W-DH assumption : For every probabilistic, polynomial-time, 0/1-valued algorithm $\mathcal{A}$, $\mathsf{Adv}_{\mathcal{A}}^{\mathsf{W-DH}}$ is negligible.

6. **Decisional Bilinear Diffie-Hellman (DBDH) problem in** (G_1, G_2, e) **:**
Instance : (P, aP, bP, cP, r) for some $a, b, c, r \in Z_q^*$.
Solution : Output yes if $r = e(P, P)^{abc} \bmod q$ and output no otherwise.

   This is a decision version of BDH problem in (G_1, G_2, e) . The advantage of any probabilistic, polynomial-time, 0/1-valued algorithm $\mathcal{A}$ in solving DBDH problem in (G_1, G_2, e) is defined to be :

$$\mathsf{Adv}_{\mathcal{A}}^{\mathsf{DBDH}} = |\mathsf{Prob}[\mathcal{A}(P, aP, bP, cP, r) = 1] - \mathsf{Prob}[\mathcal{A}(P, aP, bP, cP, e(P, P)^{abc}) = 1] : a, b, c, r \in_R Z_q^*|.$$

   DBDH assumption : For every probabilistic, polynomial-time, 0/1-valued algorithm $\mathcal{A}$, $\mathsf{Adv}_{\mathcal{A}}^{\mathsf{DBDH}}$ is negligible.

7. **Decisional Hash Bilinear Diffie-Hellman (DHBDH) problem in** (G_1, G_2, e) **:**
Instance : (P, aP, bP, cP, r) for some $a, b, c, r \in Z_q^*$ and a one way hash function $H : G_2 \to Z_q^*$.
Solution : Output yes if $r = H(e(P, P)^{abc}) \bmod q$ and output no otherwise.

   The DHBDH problem in (G_1, G_2, e) is a hash version of the decisional BDH problem in (G_1, G_2, e) .
   The advantage of any probabilistic, polynomial-time, 0/1-valued algorithm $\mathcal{A}$ in solving

DHBDH problem in (G_1, G_2, e) is defined to be :
$\mathsf{Adv}_{\mathcal{A}}^{\mathsf{DHBDH}} = |\mathsf{Prob}[\mathcal{A}(P, aP, bP, cP, r) = 1]$
$-\mathsf{Prob}[\mathcal{A}(P, aP, bP, cP, H(e(P, P)^{abc})) = 1] : a, b, c, r \in_R Z_q^*|$.

DHBDH assumption : For every probabilistic, polynomial-time, 0/1-valued algorithm $\mathcal{A}$, $\mathsf{Adv}_{\mathcal{A}}^{\mathsf{DHBDH}}$ is negliible.

8. **Reversion of CDH (RCDH) problem in G_1 :**
Instance : (P, aP, rP) for some $a, r \in Z_q^*$.
Solution : Output $bP, b \in Z_q^*$ satisfying $a = rb \bmod q$.

RCDHP is equivalent to CDH problem in G_1 [14].

9. **ROS problem :** (Schnorr)
Given an oracle access to a random function $F : Z_q^l \to Z_q$, find co-efficients $a_{k,i} \in Z_q^*$ and a solvable system of $l+1$ distinct equations in the unknowns $c_1, c_2, \ldots, c_l$ over Z_q^* :
$a_{k,1}c_1 + \cdots + a_{k,l}c_l = F(a_{k,1}, \ldots, a_{k,l})$ for $k = 1, 2, \ldots, t$, $t \geq l+1$.

10. **$(k+1)$-exponent problem ($(k+1)$-EP) in G_1:**
Instance : $(P, yP, y^2P, \ldots, y^kP)$for a random $y \in Z_q^*$.
Solution : Output $y^{k+1}P$.

The advantage of any probabilistic, polynomial-time, 0/1-valued algorithm $\mathcal{A}$ in solving $(k+1)$-EP in G_1 is defined to be :

$$\mathsf{Adv}_{\mathcal{A}}^{\mathsf{(k+1)-EP}} = \mathsf{Prob}[\mathcal{A}(P, yP, y^2P, \ldots, y^kP, y^{(k+1)}P) = 1 : y \in_R Z_q^*].$$

$(k+1)$**-EP assumption :** For every probabilistic, polynomial-time, 0/1-valued algorithm $\mathcal{A}$, $\mathsf{Adv}_{\mathcal{A}}^{\mathsf{(k+1)-EP}}$ is negligible.
$(k+1)$-EP is no harder than the CDH problem.

11. **Chosen-target CDH problem in G :**
Let $G = \langle g \rangle$ be a multiplicative group of a prime order q. Let x be a random element of Z_q^* and let $y = g^x$.
Let H_1 be a random instance of a hash function family $[\{0, 1\} \to G^*]$.
The adversary $\mathcal{A}$ is given input (q, g, H_1, y) and has access to the target oracle $\mathcal{T}_G$ that returns random elements z_i of G and the helper oracle $(\cdot)^x$.
Let q_T and q_H be the number of queries $\mathcal{A}$ made to the target oracle and helper oracle respectively.
The advantage of the adversary in attacking chosen-target CDH problem $\mathsf{Adv}_G^{\mathsf{ct-cdh}}(\mathcal{A})$ is defined to be the probability of $\mathcal{A}$ to output a set V of, say l paires $((v_1, j_1), (v_2, j_2), \ldots, (v_l, j_l))$, where for all i, $1 \leq i \leq l$, there exists j_i, $1 \leq j_i \leq q_T$ such that $v_i = z_{j_i}^x$, all v_i are distinct and $q_H < q_T, l$.
The chosen-target CDH assumption states that there is no probabilistic, polynomial- time, 0/1- valued adversary $\mathcal{A}$ with non-negligible $\mathsf{Adv}_G^{\mathsf{ct-cdh}}(\mathcal{A})$.

12. **Chosen-target Inverse CDH problem in G_1 :**
Let G_1 (additive) be a GDH group of prime order q and P be a generator of G_1. Let s be a random element of Z_q^* and $Q = sP$.

Let $H_1 : \{0,1\} \to G_1$ be a cryptographic hash function. The adversary $\mathcal{A}$ is given input (q, P, Q, H_1) and has access to the target oracle T_{G_1} that returns a random element U_i in G_1 and the helper oracle $\mathsf{Inv-cdh-s}(\cdot)$ that computes $s^{-1}\,(\cdot)$.
Let q_T and q_H be the number of queries $\mathcal{A}$ makes to the target oracle and the helper oracle respectively.
The advantage of the adversary in attacking the chosen-target inverse CDH problem $\mathsf{Adv}_{G_1}^{\mathsf{ct-icdh}}(\mathcal{A})$ is defined to be the probability of $\mathcal{A}$ to output a set of l pairs $((V_1, j_1), (V_2, j_2), \ldots (V_l, j_l)$, for all $i = 1, 2, \ldots, l$, there exists $j_i = 1, 2, \ldots, q_T$ such that $V_i = s^{-1}U_{j_i}$ where all V_i are distinct and $q_H < q_T, l$.
The chosen-target inverse CDH assumption states that there is no probabilistic, polynomial-time, 0/1-valued adversary $\mathcal{A}$ with non-negligible $\mathsf{Adv}_{G_1}^{\mathsf{ct-icdh}}(\mathcal{A})$.

13. **Generalized (or Many) Diffie-Hellman problem in $G = \langle g \rangle$:**
Instance : $(g, g^{\prod_{j \in J} y_j})$ for some $y_1, y_2, \ldots, y_l \in Z_q^*$ and for all $J \subset \{1, 2, \ldots, l\}$.
Solution : Output $g^{\prod_{j=1}^{l} y_j}$.

The advantage of any probabilistic, polynomial-time 0/1-valued algorithm $\mathcal{A}$ in solving Many-DH problem in G is defined to be

$$\mathsf{Adv}_{\mathcal{A}}^{\mathsf{Many-DH}} = \mathsf{Prob}[\mathcal{A}(g, g^{\prod_{j \in J} y_j}, g^{\prod_{j=1}^{l}}) = 1 | y_j \in Z_q^* \text{ for } 1 \le j \le l \text{ and for all } J \subset \{1, \ldots, l\}].$$

Many-DH assumption : For every probabilistic, polynomial-time, 0/1-valued algorithm $\mathcal{A}$, $\mathsf{Adv}_{\mathcal{A}}^{\mathsf{Many-DH}}$ is negligible.

14. **Co-GDH group :**
Consider a cryptographic bilinear map in the following setup :
a) G_1, G_2 and G_T are multiplicative cyclic groups of prime order q;
b) g_1 is a generator of G_1 and g_2 is a generator of G_2;
c) ψ is a computable isomorphism from G_1 to G_2, with $\psi(g_1) = g_2$; and
d) e is a computable bilinear map $e : G_1 \times G_2 \to G_T$ satisfying the following properties :
Bilinearity : For all $u \in G_1, v \in G_2$ and $a, b \in Z_q^*$,

$$e(u^a, v^b) = e(u, v)^{ab}.$$

Non-degeneracy : $e(g_1, g_2) \neq 1$.

These properties imply two more : for any $u \in G_1, v_1, v_2 \in G_2$, $e(u, v_1v_2) = e(u, v_1)\ e(u, v_2)$; for any $u, v \in G_1$, $e(u, \psi(v)) = e(v, \psi(u))$.
(such bilinear maps can be derived from Weil pairing and Tate pairing; for simplicity the reader may assume $G_1 = G_2$).
With this setup, we obtain natural generalizations of the CDH and DDH problems :

Decisional Co-Diffie-Hellman problem :
Instance : $\langle g_1, g_2, g_1^a, g_2^b, g_2^c \rangle$ for some $a, b, c \in Z_q^*$.
Solution : Output yes if $c = ab \bmod q$ and output no otherwise.

Computational Co-Diffie-Hellman problem :
Instance : $\langle g_1, g_2, g_1^a, g_2^b \rangle$ for some $a, b \in Z_q^*$.
Solution : Output $g_2^{ab} \in G_2$.

When $G_1 = G_2$ and $g_1 = g_2$, these problems reduced to the standard DDH and CDH problems respectively.

Groups G_1, G_2 are said to be Co-GDH groups if there exists an efficient algorithm to solve the Decisional Co-DH problem and there is no polynomial-time (in $|q|$) algorithm to solve the Computational Co-DH problem. The existence of a cryptographic bilinear map ensures the existance of Co-GDH groups.

3 Encryption Schemes

In identity-based public key encryption, the public key distribution problem is eliminated by making each user's public key derivable from some known aspect of his identity, such as his email address. When Alice wants to send a message to Bob, she simply encrypts her message using Bob's public key which she merely derives from Bob's identifying information. Bob, when receives the encrypted message, he obtains his private key from a third paty called a Private Key generator (PKG) after authenticating himself to PKG and can decrypt the message. The private key that PKG generates on Bob's query is a function of it's master key and Bob's identity.
Some disadvantages of ID-based system are : (1) the PKG knows Bob's private key, *i.e.* key escraw is inherent in the system which for some applications may be a serious problem, (2) Bob has to authenticate himself to it's PKG in the same way as he would authenticate himself to a CA, (3) Bob's PKG requires a secure channel to send Bob his private key, (4) Bob has to publish his PKG's public parameters and Alice must obtain these parameters before sending an encrypted message to Bob.
However, the advantage of ID-based encryption are compelling. It makes maintaining authenticated public key derectories unnecessary. Instead, a directory for authenticated public parameters of PKG's is required which is less burdensome than maintaining a public key directory since there are substantially fewer PKGs than total users. In particular, if everyone uses a single PKG, then everyone in the system can communicate securely and users need not to perform online lookup of public keys or public parameters. Bilinear pairing makes ID-based encryption schemes feasible. In 2001, Boneh, Franklin [7] proposed the first pairing based encryption scheme.

3.1 The ID-based Encryption Scheme

(Boneh, Franklin, [7], 2001)

Assumption : BDH problem hard.

Setup : Generate two groups G_1 (additive) and G_2 (multiplicative) of prime order q and a bilinear map $e : G_1 \times G_1 \to G_2$. ($G_1$ is a suitable subgroup of an elliptic curve).
Choose an arbitrary generator $P \in G_1$ and a random $s \in Z_q^*$.

Set $P_{pub} = sP$ and choose cryptographic hash functions $H_1 : \{0,1\}^* \to G_1^*$ and $H_2 : G_2 \to \{0,1\}^*$, n is the bit length of messages. Choose master key $s \in Z_q^*$.

Extract : (1) Compute $Q_{\mathsf{ID}} = H_1(\mathsf{ID}) \in G_1$, (2) set private key $d_{\mathsf{ID}} = sQ_{\mathsf{ID}}$.

Encrypt : (1) Choose a random $r \in Z_q^*$, (2) set the ciphertext for the message M to be

$$C = \langle rP, M \oplus H_2(g_{\mathsf{ID}}^r)\rangle,$$

where $g_{\mathsf{ID}} = e(Q_{\mathsf{ID}}, P_{pub})$

Decrypt : Given $C = \langle U, V\rangle$, compute

$$V \oplus H_2(e(d_{\mathsf{ID}}, U)).$$

Security : This is the basic scheme. Security against adaptive chosen ciphertext attack in the random oracle model under the BDH assumption is obtained after the Fujisaki-Okamoto [19] transformation.

Efficiency : **Setup**– 1 EC scalar multiplication; **Extract**– 1 Map-to-point hash operation + 1 elliptic curve (EC) scalar multiplication; **Encrypt**– 1 Map-to-point hash operation + 1 EC scalar multiplication + 1 hash function (H_2) evaluation + 1 XOR operation + 1 pairing computation + 1 group exponent in G_2; **Decrypt**– 1 hash function (H_2) evaluation + 1 XOR operation + 1 pairing computation.

3.2 Searchable Public Key Encryption

(Boneh, Crescenzo, Ostrovsky, Persiano, [6], 2003)

Assumption : BDH problem hard.

Suppose user Alice wishes to read her email on a number of devices : laptop, desktop, pager, etc. Alice's mail gateway is supposed to route email to the appropriate device based on the keywords in the email. Consider Bob sends email with keyword "urgent". The gateway routes the email to Alice's pager, after testing whether the email contains this keyword "urgent" without learning anything else about the mail. This mechanism is refered to as *Searchable Public Key Encryption* (SPKE).

To send a message M with keywords $W_1, \ldots, W_n$, Bob sends

$$E_{A_{pub}}(M)||\mathsf{SPKE}(A_{pub}, W_1)|| \ldots ||\mathsf{SPKE}(A_{pub}, W_n)$$

where $E_{A_{pub}}(M)$ is the encryption of M using Alice's public key A_{pub}. The point of searchable encryption is that given $\mathsf{SPKE}(A_{pub}, W')$ and a certain trapdoor T_W (that is given to the gateway by Alice), the gateway can test whether $W = W'$. If $W \neq W'$ the gateway learns nothing more about W'.

A non-interactive SPKE scheme using bilinear map :

KeyGen : The public key is $\mathsf{PK} = (P, A_{pub}), A_{pub} = sP, s \in Z_q^*$ and the secret key is $\mathsf{SK} = s$.

SPKE(PK, W) : Choose a random $r \in Z_q^*$. Output

$$\langle rP, H_2(e(H_1(W), A_{pub})^r)\rangle.$$

Trapdoor(SK, W) : Output $T_W = sH_1(W)$.

Test(PK, S, T_W) : Parse $S = \langle U, V\rangle$. Test if $V = H_2(e(T_W, U))$. If so, output yes, else output no.

Security : Semantically secure against a chosen keyword attack in the random oracle model assuming BDH problem is intractable.

Efficiency : **KeyGen**– 1 EC scalar multiplication; **SPKE**– 1 Map-to-point hash operation + 1 EC scalar multiplication + 1 hash function (H_2) evaluation + 1 pairing computation + 1 group exponent in G_2.

3.3 Hierarchical ID-Based Encryption (HIDE) Scheme

(Gentry, Silverberg, [21], 2002)

Assumption : BDH problem hard.

Although having a single private key generator (PKG) would completely eliminate online lookup, it is undesirable for a large network because the PKG has a burdensome job. Not only is private key generation computationally expensive, but also the PKG must verify proofs of identity and must establish secure channels to transmit private keys. HIDE allows a root PKG to distribute the workload by delegating private key generation and identity authentication to lower-level PKGs. In a HIDE scheme, a root PKG need only generate private keys for domain-level PKGs, who in turn generate private keys for users in their domains in the next level. Authentication and private key transmission can be done locally. To encrypt a message to Bob, Alice need only to obtain the public parameters of Bob's root PKG (and Bob's identitifying information); there are no "lower-level parameters". HIDE has the advantage of damage control : disclosure of a domain PKG's secret does not compromise the secrets of higher-level PKGs which is not possessed by schemes of Cocks [15] and Boneh-Franklin [7]. The bit-length of the ciphertext and the complexity of decryption graw only linearly with the level of the message recipient.

<u>BasicHIDE</u> :
The entities in the tree (other than the root) are the users of the tree. Let Level_i be the set of

entities at level i, where $\mathsf{Level}_0 = \{\mathsf{Root\ PKG}\}$.

Root Setup : The root PKG :
1. generates groups G_1, G_2 of some prime order q and an admissible bilinear pairing $e : G_1 \times G_1 \to G_2$. (For convenience, G_1 is an additive group and G_2 a multiplicative group.)
2. chooses an arbitrary generator $P_0 \in G_1$.
3. picks a random $s_0 \in Z_q^*$ and sets $Q_0 = s_0 P_0$.
4. chooses cryptographic hash functions $H_1 : \{0,1\}^* \to G_1$ and $H_2 : G_2 \to \{0,1\}^n$.
The message space is $\mathcal{M} = \{0,1\}^n$.
The ciphertext space is $\mathcal{C} = G_1^t \times \{0,1\}^n$ where t is the level of the recipient.
The system parameters are $\mathsf{params} = (G_1, G_2, e, P_0, Q_0, H_1, H_2)$. The root PKG's secret is $s_0 \in Z_q^*$.

Lower-level Setup : Entity $E_t \in \mathsf{Level}_t$ picks a random $s_t \in Z_q^*$ which it keeps secret.

Extract : Let E_t be an entity in Level_t with ID-tuple $(\mathsf{ID}_1, \ldots, \mathsf{ID}_t)$, where $(\mathsf{ID}_1, \ldots, \mathsf{ID}_i)$ for $1 \le i \le t$ is the ID-tuple of E_t's ancestor at Level_i. Set S_0 to be the identity element of G_1.
Then E_t's parent :
1. computes $P_t = H_1(\mathsf{ID}_1, \ldots, \mathsf{ID}_t) \in G_1$
2. sets E_t's secret point S_t to be $S_{t-1} + s_{t-1}P_t = \sum_{i=1}^{t} s_{i-1}P_i$
3. also gives E_t the values of $Q_i = s_i P_0$ for $1 \le i \le t-1$.

Encrypt : To encrypt $M \in \mathcal{M}$ with the ID-tuple $(\mathsf{ID}_1, \ldots, \mathsf{ID}_t)$, do the following :
1. compute $P_i = H_1(\mathsf{ID}_1, \ldots, \mathsf{ID}_i) \in G_1$ for $1 \le i \le t$.
2. choose a random $r \in Z_q^*$.
3. set the ciphertext to be

$$C = \langle rP_0, rP_2, \ldots, rP_t, M \oplus H_2(g^r) \rangle$$

where $g = e(Q_0, P_1) \in G_2$.

Decrypt : Let $C = \langle U_0, U_2, \ldots, U_t, V \rangle \in \mathcal{C}$ be the ciphertext encrypted using the ID-tuple $(\mathsf{ID}_1, \ldots, \mathsf{ID}_t)$. To decrypt C, E_t computes :

$$V \oplus H_2\left(\frac{e(U_0, S_t)}{\prod_{i=2}^{t} e(Q_{i-1}, U_i)} \right) = M.$$

Note : The scheme is derived from Boneh-Franklin [7] scheme. An interesting fact is that lower-level PKGs need not always use the same s_t for each private key extraction. Rather, s_t could be generated randomly for each of the PKG's children. Another fact is that H_1 can be chosen to be an iterated hash function, for example, P_i may be computed as $H_1(P_{i-1}, \mathsf{ID}_i)$ rather than $H_1(\mathsf{ID}_1, \ldots, \mathsf{ID}_i)$.

Security : Chosen ciphertext security of this basic scheme is obtained by using Fujisaki-Okamoto [19] padding in the random oracle model under the assumption that BDH problem is hard.

3.4 Dual-HIDE : Dual-Identity-Based Encryption

(Gentry, Silverberg [21], 2002)

Assumption : BDH problem hard.

Dual-HIDE may be more efficient than HIDE if the sender and recipient are close to each other in the hierarchy tree. Suppose two users, y and z, have the ID-tuples $(\mathsf{ID}_{y_1}, \ldots, \mathsf{ID}_{y_l}, \ldots, \mathsf{ID}_{y_m})$ and $(\mathsf{ID}_{z_1}, \ldots, \mathsf{ID}_{z_l}, \ldots, \mathsf{ID}_{z_n})$, where $(\mathsf{ID}_{y_1}, \ldots, \mathsf{ID}_{y_l}) = (\mathsf{ID}_{z_1}, \ldots, \mathsf{ID}_{z_l})$.
In other words, user y is in Level_m, user z is in Level_n and they share a common ancestor in Level_l. User y may use Dual-HIDE to encrypt a message to user z as follows :

Encrypt : To encrypt $M \in \mathcal{M}$, user y :
1. computes $P_{Z_i} = H_1(\mathsf{ID}_{z_1}, \ldots, \mathsf{ID}_{z_i}) \in G_1$ for $l+1 \le i \le n$
2. chooses a random $r \in Z_q^*$
3. sets the ciphertext to be

$$C = \langle rP_0, rP_{z_{l+1}}, \ldots, rP_{z_n}, M \oplus H_2(g_{y_l}^r) \rangle$$

where

$$g_{y_l} = \frac{e(P_0, S_y)}{\prod_{i=l+1}^{m} e(Q_{y_{(i-1)}}, P_{y_i})} = e(P_0, S_{y_l}).$$

S_y is y's secret point, S_{y_l} is the secret point of y's and z's common ancestor at level l and $Q_{y_i} = s_{y_i} P_0$ where s_{y_i} is the secret number chosen by y's ancestor at level i.

Decrypt : Let $C = \langle U_0, U_{l+1}, \ldots, U_n, V \rangle$ be the ciphertext. To decrypt C, user z computes :

$$V \oplus H_2\left(\frac{e(U_0, S_z)}{\prod_{i=l+1}^{n} e(Q_{z_{(i-1)}}, U_i)} \right) = M.$$

Note : If y and z have a common ancestor below the root PKG, then the ciphertext is shorter with Dual-Hide than with non-dual HIDE. Further, using Dual HIDE, the encrypter y computes $(m-l+1)$ pairings while the decrypter z computes $(n-l+1)$ pairings. In the non-dual HIDE scheme, the encrypter computes one pairing while the decrypter computes n pairings. Thus when $m < (2l-1)$, the total work is less with Dual-HIDE than with non-dual HIDE. Dual-HIDE also makes domain-specific broadcast encryption possible. One can restrict key escrow using Dual-HIDE.

Security : Secure in the random oracle model assuming the hardness of BDH problem.

4 Signature Schemes

Digital signatures are one of the most important cryptographic primitives. In traditional public key signature algorithms, the binding between the public key and the identity of the signer is obtained via a digital certificate. Shamir [36] first noticed that it would be more efficient if there was no need for such bindings, in that case given the user's identity, the public key could be easily derived using some public deterministic algorithm. This makes efficient ID-based signature schemes desirable.

In ID-based signature scemes, verification function is easily obtained from identity, possibly the same key and the same underlying computation primitives can be used. In 2001, Boneh, Lynn, Shacham [9] proposed a pairing based short signature scheme. Afterwords several ID-based signature schemes using pairing were developed.

4.1 BLS Short Signature Scheme

(Boneh, Lynn, Shacham, [9], 2001)

Assumption : GDH group.

Short signatures are needed in environments with space and bandwidth constraints. For example, when humans are asked to type in a digital signature the shortest possible signatures are needed.

KeyGen : Let G_1, G_2 be two multiplicative groups of same prime order q and $e : G_1 \times G_1 \to G_2$ be a bilinear map. $G_1 = \langle g \rangle$, $H : \{0,1\}^* \to G_1$, messages are arbitrary finite strings.
$x \in_R Z_q^*$ is the secret key and $y = g^x$ is the public key.

Sign : Given a secret key x and a message $m \in \{0,1\}^*$, compute $\sigma = H(m)^x$.

Verify : Given a public key $y = g^x$, a message m and a signature σ, verify if $\langle g, y, H(m), \sigma \rangle$ is a valid DDH tuple. In otherwords, verify if $e(g, \sigma) = e(y, H(m))$.

Security : Secure against existential forgery under adaptive chosen message attack in the random oracle model assuming the underlying group is GDH.

Efficiency : **KeyGen**– 1 group exponent in G_1; **Sign**– 1 Map-to-point hash operation + 1 group exponent in G_1; **Verify**– 1 Map-to-point hash operation + 2 pairing computation.

4.2 Blind Signature Scheme

(Boldyreva [4], 2003)

Assumption : Chosen-target CDH problem hard.

Blind signatures are the basic tools of digital cash schemes. Using a blind signature protocol, a user can obtain from a bank a digital coin, that is a token properly signed by the bank. The goal of blind signature protocol is to enable a user to obtain a signature from a signer so that the signer does not learn information about the message it signed and so that the user can not obtain more than one valid signature after one interaction with the signer.

KeyGen : Let G_1 (additive), G_2 (multiplicative) be two groups of same prime order q, $G_1 = \langle P \rangle$, $H : \{0,1\}^* \to G_1$, $e : G_1 \times G_1 \to G_2$.
Secret key is $x \in_R Z_q^*$ and public key is $P_{pub} = xP$.

Blind Signature Issuing Protocol : To sign $m \in \{0,1\}^*$

–(Blinding) The user chooses randomly $r \in Z_q^*$, computes $M' = rH(m)$ and sends M' to signer.

–(Signing) The signer computes $\sigma' = xM'$ and sends back σ' to the user.

–(Unblinding) The user then computes the signature $\sigma = r^{-1}.\sigma'$ and outputs (m, σ).

Verify : Given a public key P_{pub}, a message m and a signature σ, verify if

$$e(P_{pub}, H(m)) = e(P, \sigma).$$

Security : Secure against one more forgery under chosen message attack assuming the hardness of chosen-target CDH problem.

4.3 Multisignature Scheme

(Boldyreva, [4], 2003)

Assumption : GDH group.

A multisignature scheme allows any subgroup of a group of users to jointly sign a document such that a verifier is convinced that each member of the subgroup participated in signing.

KeyGen : G_1, G_2 both are multiplicative group, $e : G_1 \times G_1 \to G_2$ a bilinear map, $|G_1| = |G_2| = q$, $G_1 = \langle g \rangle$. User $u_i \in U$ has secret key $\mathsf{SK}_i = x_i \in Z_q^*$ and public key $\mathsf{PK}_i = g^{x_i}, 1 \leq i \leq n$.

Multisignature Creation : Any user $u_i \in U$ with secret key $\mathsf{SK}_i = x_i$ that wishes to participate in signing a message $m \in \{0,1\}^*$, computes $\sigma_i = H(m)^{x_i}$ and sends it to a designated signer D (which can be implemented by any user). Let $L = \{u_{i_1}, \ldots u_{i_l}\} \subseteq U$ be a subset of users contributed to the signing. D after getting all the σ_j for $j \in J = \{i_1, \ldots, i_l\}$, computes the multisignature $\sigma = \prod_{j \in J} \sigma_j$ and outputs (m, L, σ).

Multisignature Verification : Given $T = (m, L, \sigma)$ and the list of public keys of the user in L : $\mathsf{PK}_j = g^{x_j}, j \in J = \{i_1, \ldots, i_l\}$, the verifier computes $\mathsf{PK}_L = \prod_{j \in J} \mathsf{PK}_j = \prod_{j \in J} g^{x_j}$ and verify

$$e(g, \sigma) = e(\mathsf{PK}_L, H(m)).$$

Note : The above multisignature scheme is a simple modification of the BLS signature scheme.

Security : Secure against existential forgery under chosen message attack in the random oracle model under the assumption that the underlying group is GDH.

4.4 Optimistic Fair Exchange

The problem of *fair exchange* allows two parties to exchange items in a fair way, so that either each party gets the other's item, or neither party does. In digital world, this problem is roughly the

following :
Alice is willing to sign some statement, for example, e-cash payment, but only if Bob fulfills some obligation (delivers some good). On the other hand, Bob is not willing to fulfill this obligation unless he is sure that he gets the signature from Alice. This circularity can be overcome by introducing a semi-trusted *arbitrator* Charlie to the model. Alice will first register her key with Charlie. This registration is performed only once, and as a result, Charlie may possibly learn some part of Alice's secret. Upon the completion of one-time registration process, Alice can perform many fair exchanges with different marchants. In any such exchange, Alice first issues some verifiable "partial signature" σ' to Bob. Bob verifies the validity of this partial signature and fulfills his obligation by sending Alice the required information I, after which Alice sends "full signature" σ to complete the transaction. Thus, if no problem occures, Charlie does not participate in the protocol (such protocols are called *optimistic*). However, if Alice refuses to send her full signature σ at the end, Bob will send σ' to Charlie (and a proof of fulfilling his obligation, including the information I that should be sent to Alice), and Charlie will convert σ' into σ, sending σ to Bob and I to Alice.

Formal model for non-interactive fair exchange (*equivalently*, verifiably committed signature) :

A verifiably committed signature involves the signer Alice, the verifier Bob and the arbitrator Charlie, and is given by the following efficient algorithms :

Setup : This is an interactive protocol between Alice and Charlie, by the end of which either one of the parties aborts or Alice learns her secret signing key SK, Charlie learns his secret arbitration key ASK, and both parties agree on Alice's public verification key PK, and partial verification key APK.

Sign and **Verify :** These are conventional signing and verification algorithms of an ordinary signature scheme. $\mathsf{Sign}(m, \mathsf{SK})$ run by Alice outputs σ on m, while $\mathsf{Verify}(m, \sigma, \mathsf{PK})$ run by Bob (or any verifier) outputs 1 (accept) or 0 (reject).

PSign and **PVerify :** These are partial signing and verification algorithms, which are just like ordinary signing and verification algorithms, except they can depend on the public arbitration key APK. $\mathsf{PSign}(m, \mathsf{SK}, \mathsf{APK})$ run by Alice outputs a partial signature σ', while $\mathsf{PVerify}(m, \sigma', \mathsf{PK}, \mathsf{APK})$ run by Bob (or any verifier) outputs 1 (accept) or 0 (reject).

Resn : This is a resolution algorithm run by Charlie in case Alice refuses to open her signature σ to Bob, who in turn possesses a valid partial signature σ' on m (and a proof that he fulfills his obligation to Alice). In this case, $\mathsf{Resn}(m, \sigma', \mathsf{ASK}, \mathsf{PK})$ should output a legal signature σ of m.

Correctness states that $\mathsf{Verify}(\mathsf{Sign}(m, \mathsf{SK}), \mathsf{PK}) = 1$, $\mathsf{Pverify}(m, \mathsf{Psign}(m, \mathsf{SK}, \mathsf{APK}), \mathsf{PK}, \mathsf{APK}) = 1$ and $\mathsf{Verify}(m, \mathsf{Resn}(\mathsf{Psign}(m, \mathsf{SK}, \mathsf{APK}), \mathsf{ASK}, \mathsf{PK}), \mathsf{PK}) = 1$. Moreover, the "resolved signature" $\mathsf{Resn}(\mathsf{Psign}(m, \mathsf{SK}, \mathsf{APK}), \mathsf{ASK}, \mathsf{PK})$ is identical to the "actual signature" $\mathsf{Sign}(m, \mathsf{SK})$.

Construction of a Verifiably Committed Signature based on sequential two party multisignatures of Boldyreva :

(Dodis, Reyzin [17], 2003)

Assumption : GDH group.

Let G be a multiplicative group of prime order q, $H : \{0,1\}^* \to G$ be a hash function and $\mathcal{V}_{\mathsf{DDH}}$ be an efficient polynomial time algorithm which solves DDH problem.

Setup : Alice chooses random $g \in G, x, x_1 \in Z_q^*$, computes $x_2 = x - x_1 \bmod q$, $h = g^x, h_1 = g^{x_1}$, and sets $\mathsf{PK} = (g, h)$, $\mathsf{SK} = (x, x_1)$, $\mathsf{APK} = h_1$, $\mathsf{ASK} = x_2$. She then sends $\mathsf{PK}, \mathsf{APK}, \mathsf{ASK}$ to Charlie, who checks that $h = h_1 g^{x_2}$ (and rejects if this is not the case).

Sign, **Verify :** These are identical to the BLS signature : $\mathsf{Sign}(m) = H(m)^x$, $\mathsf{Verify}(m, \sigma) = \mathcal{V}_{\mathsf{DDH}}(g, h, H(m), \sigma)$.

PSign, **PVerify :** These are also identical to the BLS signature, but with public key h_1 : $\mathsf{PSign}(m) = H(m)^{x_1}$, $\mathsf{PVerify}(m, \sigma') = \mathcal{V}_{\mathsf{DDH}}(g, h_1, H(m), \sigma')$.

Resn : $\mathsf{Resn}(m, \sigma')$ first checks that $\mathsf{PVerify}(m, \sigma') = 1$, and outputs $\sigma = \sigma' H(m)^{x_2}$.

Security : Secure in the random oracle model.

4.5 Aggregate Signature

(Boneh, Gentry, Lynn, Shacham [8], 2003)

Assumption : Co-GDH group and existence of a bilinear map.

An aggregate signature scheme is a digital signature that supports aggregation : Given n signatures on n distinct messages m_i from n distinct users i, $1 \leq i \leq n$, it is possible to aggregate all these signatures into a single short signature. This single signature and the n original messages $m_i, 1 \leq i \leq n$ will convince the verifier that user i did indeed sign message m_i , $1 \leq i \leq n$.

KeyGen : G_1, G_2, G_T are multiplicative groups of some large prime order q, $\psi : G_1 \to G_2$ is a computable isomorphism and $e : G_1 \times G_2 \to G_T$ is the bilinear map. $G_1 = \langle g_1 \rangle$, $G_2 = \langle g_2 \rangle$, $g_2 = \psi(g_1)$, $H : \{0,1\}^* \to G_2$.
$x_i \in Z_q^*$ is the secret key and $v_i = g_1^{x_i}$ is the public key of user $u_i \in U, 1 \leq i \leq n$.

Aggregation : User $u_i \in U$ signs message $m_i \in \{0,1\}^*$ to generate BLS signature $\sigma_i = H(m_i)^{x_i}$, $1 \leq i \leq n$. The messages m_i must be all distinct. The aggregate signature is $\sigma = \sigma_1 \sigma_2 \ldots \sigma_n \in G_2$.

Aggregate verification : Given public keys $v_i = g_1^{x_i}$, distinct messages $m_i, 1 \leq i \leq n$ and an aggregate signature σ, verify if

$$e(g_1, \sigma) = \prod_{i=1}^{n} e(v_i, H(m_i)).$$

Security : Secure against existential forgery in the aggregate chosen key model assuming that the underlying groups are Co-GDH groups.

4.6 The Bilinear Verifiably Encrypted Signature

(Boneh, Gentry, Lynn, Shacham [8], 2003)

Assumption : Co-GDH group and existence of a bilinear map.

When Alice wants to sign a message for Bob but does not want Bob to possess her signature on the message immediately, Alice encrypts her signature using the public key of a trusted third party (adjudicator), and sending the result to Bob along with a proof that she has given him a valid encryption of her signature. Bob can verify that Alice has signed the message but can not deduce any information about her signature. Later in the protocol, Bob can either obtain the signature from Alice or resort to the adjudicator who can reveal Alice's signature.

KeyGen : $G_1, G_2, G_T, G_1 = \langle g_1 \rangle, G_2 = \langle g_2 \rangle$, $g_2 = \psi(g_1)$, ψ is an isomorphism from G_1 to G_2, $e : G_1 \times G_2 \to G_T$ is a bilinear map, $H : \{0,1\}^* \to G_2$ a hash function.
(x, g_1^x) is private/public key pair of signer.
$(x', g_1^{x'})$ is the private/public key pair of adjudicator.

Sign, Verify : For a message $m \in \{0,1\}^*$, the signature of a signer with private key x is $\sigma = H(m)^x \in G_2$ and the verification is

$$e(g_1, \sigma) = e(g_1^x, H(m)).$$

Verifiably Encrypted Signature Creation : Given a secret key $x \in Z_q^*$, a message $m \in \{0,1\}^*$ and an adjudicator's public key $v' = g_1^{x'} \in G_1$, compute $h = H(m) \in G_2$ and $\sigma = h^x$. Select a random $r \in Z_q^*$ and set $\mu = \psi(g_1)^r$ and $\sigma' = \psi(v')^r$. Aggregate σ, σ' as $w = \sigma\sigma' \in G_2$ and output the pair (w, μ).

Verifiably Encrypted Signature Verification : Given a public key v, a message m, an adjudicator's public key v' and a verifiably encrypted signature (w, μ), set $h = H(m)$; accept if $e(g_1, w) = e(v, h).e(v', \mu)$ holds.

Adjudication : Given an adjudicator's public key v' and corresponding private key $x' \in_R Z_q^*$, a public key v and a verifiably encrypted signature (w, μ) on some message m, ensure the verifiably encrypted signature is valid; then compute $\sigma = w/\mu^{x'}$.
(Before giving the signature, the adjudicator must perform the validity test to prevent a malicious user from tricking him into signing arbitrary messages under his adjudication key).
No involvement of adjudicator during generation of encrypted signature or its verification. Adjudi-

cator involves only during signature revelation phase.

Security : Secure against existential forgery and aggregate extraction assuming that Co-GDH [9] signature scheme is secure against existential forgery and extraction respectively.

Efficiency : **Verifiably Encrypted Signature Creation**– 1 Map-to-point hash operation + 3 group exponent in G_2 + 1 multiplication in G_2; **Verifiably Encrypted Signature Verification**– 1 Map-to-point + 3 pairing computation + 1 group exponent in G_T; **Adjudication**– 1 group exponent in G_2 + 1 multiplication in G_2.

4.7 Bilinear Ring Signature

(Boneh, Gentry, Lynn, Shacham [8], 2003)

Assumption : Co-GDH group and existence of a bilinear map.

Consider a set U of users each having a public/private key pair. Ring signature on U is a signature that is constructed using all these public keys of the users in U, and a single private key of any user in U. A ring signature has the property of signer-ambiguity : a verifier is convinced that the signature was produced using one of the private keys of U, but is not able to determine which one. Let G_1, G_2, G_T be multiplicative groups of same prime order q, $e : G_1 \times G_2 \to G_T$ a bilinear map, $\psi : G_1 \to G_2$ a computable isomorphism, $G_1 = \langle g_1 \rangle$, $G_2 = \langle g_2 \rangle$, $g_2 = \psi(g_1)$, $H : \{0,1\}^* \to G_2$.

KeyGen : $x \in_R Z_q^*$ is the secret key and $v = g_1^x$ is the public key of a particular user.

Ring Signing : Given public keys $v_1, \ldots, v_n \in G_1$, a message $M \in \{0,1\}^*$, and a private key x corresponding to one of the public keys v_s for some s, choose $a_i \in_R Z_q$ $\forall i \neq s$.
Compute $h = H(M) \in G_2$ and set

$$\sigma_s = \left(\frac{h}{\psi(\prod_{i \neq s} v_i^{a_i})} \right)^{1/x}.$$

For all $i \neq s$, let $\sigma_i = g_2^{a_i}$. Output the ring signature $\sigma = (\sigma_1, \ldots, \sigma_n) \in G_2^n$.

Ring Verification : Given public keys $v_1, \ldots, v_n \in G_1$, a message $M \in \{0,1\}^*$, and a ring signature σ, compute $h = H(M)$ and verify that

$$e(g_1, h) = \prod_{i=1}^{n} e(v_i, \sigma_i).$$

Security : The identity of the signer is unconditionally protected and the scheme is resistant to forgery in the random oracle model assuming the underlying groups are Co-GDH.

4.8 Non-interactive Deniable Ring Authentication

(Zhang, Safavi-Naini, Susilo [40], 2004)

Consider a situation when Alice, who is a member of the parliament, wishes to inform the prime minister about very sensitive information related to the country. In this situation, Alice does not want her identity to be revealed by the prime minister, and on the other hand, she also wants the prime minister to keep this information for himself and not to be forwarded to any other person. To make the information reliable, it must be authenticated and this must be verifiable by the prime minister that it comes from one of the parliament's member, so that the prime minister can make his decision on this matter. The notion of non-interactive deniable ring authentication allows a signer to sign a message m on behalf of an ad hoc collection of participants, and to convince a designated verifier V that this message is correct. Moreover, it is required that the designated verifier V can not convince any other third party that the message m was indeed authenticated.

For this chameleon hash functions are used. Chameleon hash function is associated with a pair of public and private keys and has the following properties :
(1) Anyone who knows the public key can compute the associated hash function.
(2) For people who do not have the knowledge of the trapdoor (*i.e.* the secret key), the hash function is collision resistant : it is infeasible to find two inputs which are mapped to the same output.
(3) The trapdoor information holder can easily find collisions for every given input.

Concrete example of Chameleon Hash :

Following is a chameleon hash which is based on the hardness of discrete log problem.

$x \in_R Z_q^*$ is the private key of user V,
$y = g^x$ is the public key of user V.
For a given message $m \in Z_q^*$, choose a random $r \in Z_q^*$ and set

$$\mathsf{Cham-Hash}_V(m, r) = g^m y^r.$$

Note that the above chameleon hash function is collision resistant for any user $U \neq V$. User V can always find any other message $\hat{m} \neq m$ and compute the appropriate $\hat{r}$ to find the same hash value, because he knows x and can easily solve $m + xr = \hat{m} + x\hat{r}$.

Non-interactive deniable ring authentication scheme :

Assumption : Co-GDH group, existence of a bilinear map and the above discrete log based hash function.

KeyGen : $\overline{x} \in_R Z_q^*$ is the secret key and $v = g_1^{\overline{x}} \in G_1$ is the public key of user u.

Non-interactive Ring Signing : Given public keys $v_1, \ldots, v_n \in G_1$, a message $m \in Z_q$ and a private key $\overline{x}$ corresponding to one of the public key v_s for some s, do the following :
1. Choose randomly $r \in Z_q^*$ and compute $\hat{h} = H(\mathsf{Cham-Hash}_V(m, r)) \in G_2$
2. Choose randomly $a_i \in Z_q^* \forall i \neq s$.
3. Set

$$\sigma_s = \left(\frac{\hat{h}}{\psi(\prod_{i \neq s} v_i^{a_i})} \right)^{1/\overline{x}}.$$

4. $\forall i \neq s$, set $\sigma_i = g_2^{a_i}$.
The ring signature is $\sigma = (\sigma_1, \ldots, \sigma_n) \in G_2^n$.

Non-interactive Ring Verification : Given public keys $v_1, \ldots, v_n \in G_1$, a message $m \in Z_q$ and a ring signature σ and r, compute

$$\hat{h} = H(\mathsf{Cham} - \mathsf{Hash}_V(m, r))$$

and verify that

$$e(g, h) = \prod_{i=1}^{n} e(v_i, \sigma_i).$$

Security : The signature scheme is non-transferable and provides signer-ambiguity assuming the above assumption.

4.9 ZSS Short Signature Scheme from Bilinear Pairing

(Zhang, Safavi-Naini, Susilo, [38], 2004)

Assumption : $(k+1)$-exponent problem hard.

KeyGen : $\mathsf{params} = (G_1, G_2, q, e, P, H)$ is the publicly available parameters.
For a signer pick randomly $x \in Z_q^*$ and set $P_{pub} = xP$. x is the secret key and P_{pub} is the public key of that user.

Sign : Given a secret key x and a message m, compute signature

$$S = \frac{1}{H(m) + x} P.$$

Verify : Given a public key P_{pub}, a message m and a signature S, verify if

$$e(H(m)P + P_{pub}, S) = e(P, P).$$

Security : Existentially unforgeable under an adaptive chosen message attack in the random oracle model assuming that $(k+1)$-exponent problem is hard.

Efficiency : **KeyGen**– 1 EC scalar multiplication; **Sign**– 1 inversion + 1 EC scalar multiplication; **Verify**– 2 pairing computation (one of which can be precomputed) + 1 EC scalar multiplication + 1 hash function (H) evaluation + 1 EC addition. This scheme is more efficient than BLS scheme as it requires less pairing computation and no computation of the expensive special hash function Map-to-point that encodes finite strings to elements of group G_1.

4.10 A New Verifiably Encrypted Signature Scheme

(Zhang, Safavi-Naini, Susilo, [39], 2003)

Assumption : $(k+1)$-exponent problem hard, DLP hard.

KeyGen : $\mathsf{params} = (G_1, G_2, q, e, P, H)$.
(P_{pub}, x) is the signer's public/private key pair.
(P_{pubAd}, x_a) is the adjudicator's public/private key pair.
$P_{pub} = xP, P_{pubAd} = x_a P$.

Sign, Verify : For a message m, the signature is $\sigma = \frac{1}{H(m)+x}P$, the verification is $e(H(m)P + P_{pub}, \sigma) = e(P, P)$.

VESig Creation : Given a secret key $x \in Z_q^*$, a message m and an adjudicator's public key P_{pubAd}, compute $\sigma' = \frac{1}{H(m)+x}P_{pubAd}$. Verifiably encrypted signature for message m is σ'.

VESig Verification : Given a public key P_{pub}, a message m, an adjudicator's public key P_{pubAd} and a verifiably encrypted signature σ', verify if

$$e(H(m)P + P_{pub}, \sigma') = e(P, P_{pubAd}).$$

Adjudication : Given an adjudicator's public key P_{pubAd} and corresponding private key $x_a \in Z_q^*$, a certified public key P_{pub} and a verifiably encrypted signature σ' on some message m, ensure that the verifiably encrypted signature is valid; then output $\sigma = x_a^{-1}\sigma'$.

Security : Secure against existential forgery in the random oracle model under the assumption that $(k+1)$-exponent problem is hard; secure against extraction assuming DLP is hard.

Efficiency : **VESig Creation**– 1 inversion + 1 EC scalar multiplication; **VESig Verification** – 2 pairing computation (one of which can be precomputed) + 1 EC scalar multiplication + 1 hash function (H) evaluation + 1 EC addition; **Adjudication**– 1 inversion + 1 EC scalar multiplication.

4.11 Partially Blind Signature Scheme

To use blind signature in designing e-cash schemes, there are two shortcomings :
1. To prevent a customer from double-spending his e-cash, the bank has to keep a database which stores all spent e-cash to check whether a specified e-cash has been spent or not by searching this database. Certainly, the database kept by the bank may grow unlimitedly.
2. The bank can not inscribe the value on the blindly issued e-cash.

Partially blind signature allows the signer to explicitly include some agreed information in the blind signature. By embedding an expiration date into each e-cash issued by the bank, all expired e-cash recorded in the bank's database can be removed and thus preventing the bank's database from growing unlimitedly. At the same time, each e-cash can be embedded the face value, the bank can know the value on the blindly issued e-cash.

A new partially blind signature scheme :
(Zhang, Safavi-Naini, Susilo [39], 2003)

Assumption : Chosen-target inverse CDH problem hard.

KeyGen : $\mathsf{params} = (G_1, G_2, q, e, P, H, H_0)$, $H_0 : \{0,1\}^* \to G_1$.
The signer picks randomly $x \in Z_q^*$ and computes $P_{pub} = xP$. The public key is P_{pub} and the secret key is x.

Partially blind signature issuing protocol : Suppose that m be the message to be signed and c be the public information.
–(Generation of the public information) The user and signer generate the public information c together.
–(Blinding) The user randomly chooses a number $r \in Z_q^*$, computes $U = r\ H_0(m||c)$ and sends U to the signer.
–(Signing) The signer computes $V = \frac{1}{H(c)+x}U$ and sends it to the user.
–(Unblinding) The user computes $S = r^{-1}V$.
Then (S, m, c) is the partially blind signature of the message m and public information c.

Verification : A verifier can accept this partially blind signature if and only if

$$e(H(c)P + P_{pub}, S) = e(P, H_0(m||c)).$$

Security : Secure against one more forgery under chosen message attack in the random oracle model assuming hardness of Chosen-target inverse CDH problem.

4.12 ID-based Blind Signature scheme (Schnorr type)

(Zhang, Kim [37], 2002)

Assumption : ROS-problem is hard.

Setup : $G_1 = \langle P \rangle, P_{pub} = sP, s \in_R Z_q^*$. $\mathsf{params} = (G_1, G_2, q, e, P, P_{pub}, H, H_1)$ is the publicly available parameters and s is the master key kept secret. H, H_1 are hash functions.

Extract : $Q_{\mathsf{ID}} = H_1(\mathsf{ID})$ is the public key for the public identity $\mathsf{ID} \in \{0,1\}^*$. The corresponding private key is $S_{\mathsf{ID}} = sQ_{\mathsf{ID}}$.

Blind Signature Issuing Protocol : To sign message $m \in \{0,1\}^*$

–The signer randomly chooses a number $r \in Z_q$, computes $R = rP$ and sends R to the user as a commitment.

–(Blinding) The user randomly chooses $a, b \in Z_q^*$ as blinding factors, computes

$$c = H(m, e(bQ_{\mathsf{ID}} + R + aP, P_{pub})) + b$$

and sends c to the signer.

–(Signing) The signer sends back S, where $S = cS_{\mathsf{ID}} + rP_{pub}$.

–(Unblinding) The user computes $S' = S + aP_{pub}$ and $c' = c - b$ and outputs (m, S', c'). Then (S', c') is the blind signature of the message m.

Verification : Accept if and only if

$$c' = H(m, e(S', P)e(Q_{\mathsf{ID}}, P_{pub})^{-c'}).$$

Security : Secure against one more forgery in the random oracle model under the above assumption.

4.13 ID-based Ring Signature

(Zhang, Kim [37], 2002)

Assumption : Intractability of the CDH problem.

Setup : $G_1 = \langle P \rangle$, $P_{pub} = sP, s \in Z_q^*$. $\mathsf{params} = (G_1, G_2, q, e, P, P_{pub}, H, H_1)$, s is the master key.

Extract : Given public identity $\mathsf{ID} \in \{0,1\}^*$, compute the public key $Q_{\mathsf{ID}} = H_1(\mathsf{ID})$ and the secret key $S_{\mathsf{ID}} = sQ_{\mathsf{ID}}$.

Let ID_i be a user's identity and S_{ID_i} be the private key associated with ID_i for $i = 1, \ldots, n$. Let $L = \{\mathsf{ID}_i\}$ be the set of identities. The real signer's identity ID_k is listed in L.
Signing :
–(Initialization) : Choose randomly an element $A \in G_1$ and compute $c_{k+1} = H(L||m||e(A, P))$.
–(Generate forward ring sequence) : For $i = k+1, \ldots n-1, 0, 1, \ldots, k-1$ (*i.e.* the value of i all modulo n), choose randomly $T_i \in G_1$ and compute

$$c_{i+1} = H(L||m||e(T_i, P)e(c_i H_1(\mathsf{ID}_i), P_{pub})).$$

–(Forming the ring) : Compute $T_k = A - c_k S_{\mathsf{ID}}$.
–(Output the ring signature) : Select 0 (*i.e.* n) as the glue value, the resulting signature for m and L is the $(n+1)$-tuple : $(c_0, T_0, T_1, \ldots, T_{n-1})$.

Verification : Given $(c_0, T_0, T_1, \ldots, T_{n-1})$, m and L, compute

$$c_{i+1} = H(L||m||e(T_i, P)e(c_i H_1(\mathsf{ID}_i), P_{pub}))$$

for $i = 0, 1, \ldots n-1$. Accept if $c_n = c_0$ and reject otherwise.

Security : The scheme is unconditionally signer-ambiguous and non-forgeable in the random oracle model under the assumption of the intractability of the CDH problem.

4.14 ID-based Group Signature Scheme

(Chen, Zhang, Kim [14], 2003)

Assumption : Reversion of CDH problem hard.

Group signature allows any member of a group to sign on behalf of the group. Anyone can verify the signature with a group public key while no one can know the identity of the signer except the Group Manager.

Suppose there exists a hierarchical ID-based system. If the group manager is not a KGC, he joins the system and becomes a KGC. So the case that group manager is a KGC is considered only.

Setup : The KGC chooses a random $s \in Z_q^*$ and sets $P_{pub} = sP$. The systems public parameters are $\mathsf{params} = (G_1, G_2, e, q, P, P_{pub}, H_1, H_2)$. The master key s is kept secret by the KGC.

Extract : A user submits his identity ID and authenticates himself to KGC. The user then randomly chooses an integer $r \in Z_q^*$ as it's long term private key and sends rP to KGC. KGC computes the user's public key $Q_{\mathsf{ID}} = H_2(\mathsf{ID}||T, rP)$ and sends $S_{\mathsf{ID}} = sQ_{\mathsf{ID}}$ to the user via a secure channel, where T is the life span of r. The user's private key pair are S_{ID} and r. The user should update his key pair after period of T.
Every user with identity ID who gets his private key S_{ID} from the KGC is a "potential" group member. S_{ID} is only used for ordinary signature. Some users may get the private key from KGC just for ordinary signature and they are not "real" group member.

Join : When a user later wants to be a "real" member of the group, he and KGC perform the Join protocol as follows :
– The user randomly chooses $x_i \in Z_q$ for $i = 1, 2, \ldots, k$. He then sends $rx_iP, x_iP, rP, \mathsf{ID}$, and S_{ID} to KGC.
– If $S_{\mathsf{ID}} = sH_2(\mathsf{ID}||T, rP)$ and $e(rx_iP, P) = e(x_iP, rP)$, KGC sends the user $S_i = sH_2(T, rx_iP)$ for $i = 1, 2, \ldots, k$. Otherwise, the protocol is terminated.
– The user's member certificates are (S_i, rx_iP) and his private signing keys are $rx_i, i = 1, 2, \ldots, k$.
– KGC adds $rx_iP, x_iP, rP, \mathsf{ID}$ to the member list.

Sign, Verify : To sign a message m, the user randomly chooses a certain signing key and corresponding member certificate and then computes the following values :
– $U = aH_2(T, rx_iP)$ for $a \in Z_q^*$ and certain i
– $V = rx_iH_2(m, U)$
– $h = H_1(m, U + V)$
– $W = (a + h)S_i$

(U, V, W, T, rx_iP) is the signature of the message m. If T is a valid period, the verifier computes $Q = H_2(T, rx_iP)$, $H_2(m, U)$, $h = H_2(m, U + V)$.
He accepts the signature if the following equations hold :

$$e(W, P) = e(U + hQ, P_{pub})$$

$$e(V,R) = e(H_2(m,U), rx_iP).$$

Open : Given a valid group signature, KGC can easily identify the user from rx_iP. The user can not deny his signature because KGC can provide a proof that it is indeed the user's signature :

$$e(rx_iP, P) = e(x_iP, rP)$$

$$e(S_{\mathsf{ID}}, P) = e(H_2(\mathsf{ID}||T, rP), P_{pub}).$$

Also, KGC can not misattribute a signature to frame the user unless he can compute bP given P, aP and rP which satisfies :

$$a \equiv rb \bmod q$$

which is defined to be the Reversion of CDH problem and is equivalent to CDHP in G_1.

Security : Secure in the random oracle model under the assumption of CDH problem.

Efficiency : The scheme is ID-based having fixed length group public key and fixed length signature. The computation and communication is linear in the number of group members, anonymity is preserved computationally, identification is done by Group manager (KGC) and new members can be included. A user has many certificates in this scheme.

4.15 Delegation-by-certificate proxy signature scheme

(Boldyreva [4], 2003)

A proxy signature permits an entity to delegate it's signing rights to another entity. To differentiate among signatures created for standard signing, proxy delegation and proxy signing, the following way is introduced :
in order to sign a message M, the user actually signs message $11||M$, whereas in order to sign a warrant w during delegation process, the user actually signs $00||w$, and for proxy signature of a message M, the proxy signer signs $01||M$. Verification of standard (respectively, proxy) signature is then performed by first prepending 11 (respectively, 01) to the message. A warrant is a message containing the public key of the designated proxy signer and possibly, restrictions on the messages the proxy signer is allowed to sign. Infact, to sign a message M on behalf of user i (with public/private key pair $(\mathsf{PK}_i, \mathsf{SK}_i)$), the proxy signer j (with public/private key pair $(\mathsf{PK}_j, \mathsf{SK}_j)$) signs message $01||\mathsf{PK}_i||M$.

Generic construction of a delegation-by-certificate proxy signature scheme from any digital signature scheme :

Let $\mathsf{DS} = (\mathcal{G}, \mathcal{K}, \mathcal{S}, \mathcal{V})$ be a standard digital signature scheme where $\mathcal{G}$ is system parameters generation algorithm, $\mathcal{K}$ is the key generation algorithm, $\mathcal{S}$ is the signature generation algorithm and $\mathcal{V}$ is the signature verification algorithm.
The proxy signature scheme $\mathsf{PS}[\mathsf{DS}] = (\mathcal{G}_1, \mathcal{K}_1, \mathcal{S}_1, \mathcal{V}_1, (\mathcal{D}, \mathcal{P}), \mathcal{PS}, \mathcal{PV}, \mathcal{ID})$ is derived from DS as follows :

1. $\mathcal{G}_1 = \mathcal{G}, \mathcal{K}_1 = \mathcal{K}$

2. $\mathcal{S}_1(\mathsf{SK}, M) = \mathcal{S}(\mathsf{SK}, 11||M)$ is the standard signature on message M.

3. $\mathcal{V}_1(\mathsf{PK}, M, \sigma) = \mathcal{V}(\mathsf{PK}, 11||M, \sigma)$ is the standard signature verification on message M.

4. To designate user j as a proxy signer, user i sends an appropriate warrant w together with a certificate which is a signature of user i on message $00||\mathsf{PK}_j||w$. The corresponding proxy signing key of user j is $\mathsf{skp} = (\mathsf{SK}_j, \mathsf{PK}_i, \mathsf{PK}_j||w, \mathsf{cert})$.

5. Proxy signer j gives a proxy signature for message M on behalf of user i using the proxy signing key $\mathsf{skp} = (\mathsf{SK}_j, \mathsf{PK}_i, \mathsf{PK}_j||w, \mathsf{cert})$ as follows :
$$\mathcal{PS}(\mathsf{skp}, M) = (w, \mathsf{PK}_j, \mathsf{cert}, \mathcal{S}(\mathsf{SK}_j, 01||\mathsf{PK}_i||M)).$$

6. Proxy signature verification is defined via
$$\mathcal{PV}(\mathsf{PK}, M, (w, \mathsf{PK}', \mathsf{cert}, \sigma)) = \mathcal{V}(\mathsf{PK}, 00||\mathsf{PK}'||w, \mathsf{cert}) \wedge \mathcal{V}(\mathsf{PK}', 01||\mathsf{PK}||M, \sigma).$$

7. The identification algorithm is defined as $\mathcal{ID}((w, \mathsf{PK}', \sigma)) = \mathsf{PK}'$.

Security : The proxy signature scheme is secure if the underlying standard digital signature scheme is secure.
Aggregate-signature-based proxy signature scheme (Boldyreva [4], 2003)

Assumption : GDH group.

Let $\mathsf{AS} = (\mathcal{G}, \mathcal{K}, \mathcal{S}, \mathcal{V}, \mathcal{A}, \mathcal{AV})$ be an aggregate signature scheme. The proxy signature scheme $\mathsf{PS}[\mathsf{AS}] = (\mathcal{G}_1, \mathcal{K}_1, \mathcal{S}_1, \mathcal{V}_1, (\mathcal{D}, \mathcal{P}), \mathcal{PS}, \mathcal{PV}, \mathcal{ID})$ is derived from AS as follows :

1. The algorithms $\mathcal{G}_1, \mathcal{K}_1, \mathcal{S}_1, \mathcal{V}_1, (\mathcal{D}, \mathcal{P})$ use the algorithms $\mathcal{G}, \mathcal{K}, \mathcal{S}, \mathcal{V}$ in the same way as those in the previous construction.

2. The proxy signing algorithm $\mathcal{PS}$ uses $\mathcal{A}$ to aggregate the certificate and a proxy signature as follows :
$\mathcal{PS}((\mathsf{SK}_j, \mathsf{PK}_i, \mathsf{PK}_j||w, \mathsf{cert}), M) = (w, \mathsf{PK}_j, \mathcal{A}(\mathsf{PK}_i, \mathsf{PK}_j, 00||\mathsf{PK}_j||w, 01||\mathsf{PK}_i||M, \mathsf{cert}, \mathcal{S}(\mathsf{SK}_j, 01||\mathsf{PK}_j||M)))$.

3. The proxy verification algorithm $\mathcal{PV}$ is defined by
$\mathcal{PV}(\mathsf{PK}, M, (w, \mathsf{PK}', \sigma)) = \mathcal{AV}(\mathsf{PK}, \mathsf{PK}', 00||\mathsf{PK}'||w, 01||\mathsf{PK}||M, \sigma)$.

4. The identification algorithm is defined by
$$\mathcal{ID}((w, \mathsf{PK}', \sigma)) = \mathsf{PK}'.$$

Security : The scheme is secure provided the underlying aggregate signature scheme is secure.

4.16 Hierarchical ID-Based Signature (HIDS) Scheme

(Gentry, Silverberg [21], 2002)

Assumption : CDH problem hard.

As noted by Moni Naor, an IBE scheme can be immediately converted into a public key signature scheme as follows :
The signer's private key is the master key in the IBE scheme. The signer's signature on M is the IBE decryption key d corresponding to the "public key" $H_1(\mathsf{ID}) = H_1(M)$.
The verifier checks the signature by choosing a random message M', encrypting M' with $H_1(M)$, and trying to decrypt the resulting ciphertext with d. If the ciphertext decrypts correctly, the signature is considered valid.
When viewed in isolation, a HIDS scheme is not especially useful, but becomes quite useful when viewed in combination with the HIDE scheme as a complete package.
<u>HIDS Scheme</u> :
Let Level_i be the set of entities at level i, where $\mathsf{Level}_0 = \{\mathsf{Root\ PKG}\}$.

Root Setup : The root PKG :
1. generates groups G_1, G_2 of prime order q and an admissible bilinear pairing $e : G_1 \times G_1 \to G_2$
2. chooses an arbitrary generator $P_0 \in G_1$
3. picks a random $s_0 \in Z_q^*$ and sets $Q_0 = s_0 P_0$
4. chooses cryptographic hash functions $H_1 : \{0,1\}^* \to G_1$ and $H_3 : \{0,1\}^* \to G_1$.

The signature space is $\mathcal{S} = G_1^{t+1}$ where t is the level of the signature. The system parameters are $\mathsf{params} = (G_1, G_2, e, P_0, Q_0, H_1, H_3)$. The root PKG's secret is $s_0 \in Z_q^*$.

Lower-level Setup : Entity $E_t \in \mathsf{Level}_t$ picks a random $s_t \in Z_q^*$, which it keeps secret.

Extract : Let E_t be an entity in Level_t with ID-tuple $(\mathsf{ID}_1, \ldots, \mathsf{ID}_t)$, where $(\mathsf{ID}_1, \ldots, \mathsf{ID}_i)$ for $1 \le i \le t$ is the ID-tuple of E_t's ancestor at Level_i. Set S_0 to be the identity element of G_1.
Then E_t's parent :
1. computes $P_t = H_1(\mathsf{ID}_1, \ldots, \mathsf{ID}_t) \in G_1$
2. sets E_t's secret point S_t to be $S_{t-1} + s_{t-1}P_t = \sum_{i=1}^{t} s_{i-1}P_i$
3. also gives E_t the values of $Q_i = s_i P_0$ for $1 \le i \le t-1$.

Sign : To sign M with $\mathsf{ID-tuple} = (\mathsf{ID}_1, \ldots, \mathsf{ID}_t)$ (using the secret point $S_t = \sum_{i=1}^{t} s_{i-1}P_i$ and the points $Q_i = s_i P_0$ for $1 \le i \le t$), do the following :
1. compute $P_M = H_3(\mathsf{ID}_1, \ldots, \mathsf{ID}_t, M) \in G_1$
2. compute $\mathsf{sig}(\mathsf{ID-tuple}, M) = S_t + s_t P_M$
3. send $\mathsf{sig}(\mathsf{ID-tuple}, M)$ and $Q_i = s_i P_0$ for $1 \le i \le t$.

Verify : Let $(\mathsf{sig}, Q_1, \ldots, Q_t) \in \mathcal{S}$ be the signature for $(\mathsf{ID-tuple}, M)$. The verifier confirms that :

$$e(P_0, \mathsf{sig}) = e(Q_0, P_1)\ e(Q_t, P_M)\ \prod_{i=2}^{t} e(Q_{i-1}, P_i).$$

Security : Secure in the random oracle model assuming the CDH problem is hard.

4.17 Unique Signature Scheme (Standard model)

Unique signature schemes, also known as invariant signature schemes, are secure signature schemes where the signature is a hand-to-compute function of the public key and the message. If a signature scheme allows the signer to easily ($i.e$ more efficiently than the cost of verifying a signature) generate many signatures on the same message, then it simply leads to denial-of-service attack on a verifier who is forced to verify many signatures on the same message. This illustrates that intutively unique signatures are desirable. Boneh and Silverberg [11] proposed a unique signature scheme based on the existence of multi-linear maps. Currently, no such suitable maps are known and the existence of such maps is presently a research problem [11]. Lysyanskaya proposed a unique signature scheme based on this idea making use of bilinear pairing. This scheme is proved to be secure in the standard model under Many-DH assumption.
(Anna Lysyanskaya [29], 2002)

Assumption : GDH group, Generalized (or Many)-DH problem hard.

KeyGen : $\mathsf{params} = (G_1, G_2, q, e, g)$, G_1 is generated by g, both G_1, G_2 are multiplicative groups of a large prime order q, $e : G_1 \times G_1 \to G_2$ is a bilinear map.
Choose n pairs of random elements in Z_q :

$$(a_{1,0}, a_{1,1}), (a_{2,0}, a_{2,1}), \ldots, (a_{n,0}, a_{n,1}).$$

This is the secret key SK. Compute

$$A_{i,b} = g^{a_{i,b}}, 1 \le i \le n, b \in \{0, 1\}.$$

The public key is

$$\mathsf{PK} = \{A_{i,0}, A_{i,1} | 1 \le i \le n\}.$$

Sign : Assume that the messages being signed are n-bit codewords of a code of distance Cn, where $0 < C \le 1/2$ is a constant. To sign an n-bit codeword $m = m_1 \circ m_2 \circ \ldots \circ m_n$ output

$$\sigma_{\mathsf{PK}}(m) = \{s_{m,i} = g^{\prod_{j=1}^{i} a_{j,m_j}} : 1 \le i \le n\}.$$

Verify : Let $s_{m,0} = 1$. Verify that, for all $i, 1 \le i \le n$,

$$e(g, s_{m,i}) = e(s_{m,i-1}, A_{i,m_i}).$$

Graphically, we view the message space as the leaves of a balanced binary tree of depth n. Each internal node of the tree is assigned a label, as follows : the label of the root is g. The label of a child, denoted l_c is obtained from the label of it's parent, denoted l_p as follows : if the depth of the child is i, and it is the left child, then its label is $l_c = l_p^{a_{i,0}}$, while if it is the right child, its label will be $l_c = l_p^{a_{i,1}}$. The signature on an n-bit message consists of all the labels on the path from the leaf corresponding to this message all the way to the root. To verify the correctness of a signature, the fact that Decision Diffie-Hellman is easy in G_1 is used.

Security : Provably secure against existential forgery under adaptive chosen message attack in the standard model assuming the underlying group is a GDH group and the hardness of Many-DH problem.

Application : Constructing Verifiably Random Functions (Anna Lysyanskaya [29], 2002)

VRFs are objects that combine the properties of psudorandom functions (*i.e.* indistinguishability from random event after querying) with the verifiability property. These objects are very useful for cryptographic protocol design, because they can be viewed as a commitment to an arbitrary number of bits. They are similar to psudorandom functions, except that they are also verifiable.

The definition of VRF is formally given below. The intution of this definition is that a function is a verifiable random function if it is like a psudorandom function with a public key and proofs.

Definition : A VRF is a function family $F_{(\cdot)}(\cdot) : \{0,1\}^k \to \{0,1\}^{l(k)}$ with the following algorithms :
1. A probabilistic key generation algorithm $G(\cdot)$ that on input k, generates public/private key pair $(\mathsf{PK}, \mathsf{SK})$.
2. A deterministic algorithm $\mathsf{Eval}(\cdot,\cdot)$ that on input SK, x computes the value $y = F_{\mathsf{PK}}(x)$.
3. A deterministic algorithm $\mathsf{Prove}(\cdot,\cdot)$ that on input SK, x computes the proof π that $y = F_{\mathsf{PK}}(x)$.
4. A probabilistic algorithm $\mathsf{Verify}(\cdot,\cdot,\cdot,\cdot)$ that on input PK, x, y, π checks whether $y = F_{\mathsf{PK}}(x)$ using the proof π.

In order to obtain verifiably random functions, it is sufficient to construct unique signatures.

4.18 A Secure Signature Scheme from Bilinear Map (standard model)

(Boneh, Mironov, Shoup [10], 2003)

Assumption : CDH problem hard.

Setup : G_0, G_1, G_2 are three multiplicative groups of order q, $e : G_0 \times G_1 \to G_2$ is a bilinear map, $\mathcal{H}_k : \mathcal{M} \to \{0,1\}^s$ is a family of collision resistant hash functions. The signature scheme allows signing l^n messages, where l and n are arbitrary positive integer, n is the branching factor of the authentication tree.

KeyGen :
1. Pick randomly $\alpha_i \in Z_q^*, 1 \le i \le n$ and $H \in_R G_1$.
Choose a random k for the collision resistant hash function $\mathcal{H}_k$.
Compute $H_1 = H^{1/\alpha_1}, \ldots, H_n = H^{1/\alpha_n} \in G_1$.

2. Pick randomly $g \in G_0$. Compute $y = e(g, H)$.

3. Pick randomly $\beta_0 \in Z_q$. Compute $x_0 = y^{\beta_0}$.

4. The public key is $k, H, H_1, \ldots, H_n, y, x_0$.
The private key is $\alpha_1, \alpha_2, \ldots, \alpha_n, \beta_0, g$.

Sign : Each node in the tree is authenticated with respect to it's parent; messages to be signed are authenticated with respect to the leaves, which are selected in sequential order and never resued. To sign ith message $M \in \mathcal{M}$, the signer generates the ith leaf of the authenticated tree together with a path from the leaf to the root.
Denote the path from leaf to root by $(x_l, i_l, x_{l-1}, i_{l-1}, \ldots, i_1, x_0)$:
x_j is the i_jth child of $x_{j-1}(i_j \in \{1, \ldots, n\})$.

1. $x_j = y^{\beta_j}$ for some $\beta_j \in_R Z_q^*, 1 \le j \le l$. The secret β_j is stored for as long as node x_j is an anscestor of the current signing leaf.

2. Compute $f_j = g^{\alpha_{i_j}(\beta_{j-1} + \mathcal{H}_k(x_j))}$. This is the authenticated value of x_j, the i_jth child of x_{j-1}.

3. Compute $f = g^{\beta_l + \mathcal{H}_k(M)}$. This is the authenticated value of M.

4. The signature on M is $(f, f_l, i_l, \ldots, f_1, i_1)$.

Verify : Given a signature $(\hat{f}, \hat{f}_l, \hat{i}_l, \ldots, \hat{f}_1, \hat{i}_1)$ on a message M, do the followings :

1. Compute $\hat{x}_l = e(\hat{f}, H) y^{-\mathcal{H}_k(M)}$.

2. Compute $\hat{x}_{j-1} = e(\hat{f}_j, H_{i_j}) y^{-\mathcal{H}_k(\hat{x}_j)}$ for $l \le j \le 1$.

3. Accept the signature if $\hat{x}_0 = x_0$.

Security : Provably secure against existential forgery against adaptive chosen message attack assuming that the CDH problem is hard.

4.19 ID-based Signature from Pairing

(K. G. Paterson, [32], 2002)

Assumption : Generalized ElGamal Signature Scheme is secure.

KeyGen : $(G_1, G_2, q, e, P, P_{pub}, H_1, H_2, H_3)$ are publicly available parameters where $P_{pub} = sP$, s is randomly choosen from Z_q^* is called the master key.
A user's public key for signature verification is $Q_{\mathsf{ID}} = H_1(\mathsf{ID})$ while secret key for signing is $D_{\mathsf{ID}} = sQ_{\mathsf{ID}}$.

Sign : To sign a message $M \in \{0,1\}^*$, choose a random $k \in Z_q^*$ and compute

$$R = kP, S = k^{-1}(H_2(M)P + H_3(R)D_{\mathsf{ID}}).$$

The pair (R, S) is the signature on message M.

Verify : Given a signature (U, V), a message M and P_{pub}, Q_{ID}, verify if

$$e(R, S) = e(P, P)^{H_2(M)} e(P_{pub}, Q_{\mathsf{ID}})^{H_3(R)}.$$

If (R, S) is a valid signature on M, then so too is $(lR, l^{-1}S)$ for any $l \in Z_q^*$. This "homomorphic" property does not appear to help an attaker to compute signatures on new messages. This scheme is similar to the generalized ElGamal signature scheme.

Security : Security of this ID-based signature scheme is linked to the security of an ordinary signature scheme which resembles the well-known non-id-based generalized ElGamal signature scheme.

Efficiency : Signature generation requires only two hash function evaluation + some computation in G_1 + an inversion modulo q. Verification requires two hash function evaluation + two exponentiations in G_2 + 3 pairing computations (two of which can be precomputed when verifying any particular user's signature) one of which can be precomputed. Size of the signature is twice the size of group elements in G_1.

4.20 ID-based Signature from Pairing

(F. Hess, [22], 2000)

Assumption : Weak-DH problem hard.

KeyGen : Public key Q_{ID}, private key $D_{\mathsf{ID}} = sQ_{\mathsf{ID}}$ where $s \in_R Z_q^*$ is the master key. Public parameters are $\mathsf{params} = (G, V, e, q, P, h_3)$, $G = \langle P \rangle$, $e : G \times G \to V$.

Sign : To sign a message m, the signer chooses an arbitrary $P_1 \in G_1^*$ and a random $k \in Z_q^*$ and computes
1. $r = e(P_1, P)^k$,
2. $v = h_3(m, r)$,
3. $u = vD_{\mathsf{ID}} + kP_1$.
The signature is then the pair $(u, v) \in (G, Z_q^*)$.

Verify : On receiving a message m and a signature (u, v) the verifier computes :
1. $r = e(u, P)e(Q_{\mathsf{ID}}, -P_{pub})^v$
2. Accept the signature if and only if $v = h_3(m, r)$.

Security : Secure against existential forgery under adaptive chosen message attack in the random oracle model assuming Weak-DH problem is hard.

Efficiency : The signing operation can be optimized by the signer pre-computing $e(P_1, P)$ for P_1 of his choice, for example $P_1 = P$, and storing this value with the signing key. This means that the signing operation involves one exponentiation in the group V, one hash function evaluation and one simultaneous multiplication in the group G.
The verification operation requires one exponentiation in V, one hash function evaluation and two evaluations of the pairing. One of the pairing evaluation can be eliminated, if a large number of

verifications are to be performed for the same identity, by pre-computing $e(Q_{ID}, -P_{pub})$.
This scheme is very efficient in terms of communication requirements. One needs to transmit one element of the group G and one element of Z_q^*.

5 Key Agreement Schemes

Key agreement is one of the fundamental cryptographic primitives. This is required when two or more parties want to communicate securely. In one of the breakthroughs in key agreement, Joux [23] proposed a three party single round key agreement protocol using pairing. This was the first positive application of bilinear pairing in cryptography. Afterwards, pairings were used widely to get a huge number of cryptographic protocols. Several key agreement protocols were proposed that prevents man-in-the-middle attack against a passive adversary. These protocols are called unauthenticated. The protocols for authenticated key agreement enables a group of parties within a large and completely insecure public network to establish a common secret key and furthermore ensures that they are indeed sharing this key with each other. Achieving authenticated key agreement are crucial for allowing symmetric-key encryption/authentication of data among the parties. Furthermore, authenticated key agreement protocols are used for constructing "secure channels" on top of which higher-level protocols can be designed, analyzed and implemented in a modular manner. Group key agreement protocols (where number of parties agreeing upon a common key is more than 2) are essential for applications such as secure video or tele-conferencing and also for collaborative (peer-to-peer) applications which are likely to involve a large number of users. A formal model of security for group authenticated key agreement can be found in [12]. Much research work remaines to be done in this area.

5.1 Joux's One Round Three Party Key Agreement Protocol

(Joux [23], 2000)

Assumption : BDH problem hard.

Let G_1, G_2 be two groups of same prime order q. We vtake G_1 to be an additive group and G_2 a multiplicative group.
The public parameters are $params = (G_1, G_2, e, q, P)$, $G_1 = \langle P \rangle$.

Consider three party A, B, C with secret keys $a, b, c \in Z_q^*$ respectively.

A sends aP to both B, C
B sends bP to both A, C
C sends cP to both A, B

A computes $K_A = e(bP, cP)^a$
B computes $K_B = e(aP, cP)^b$
C computes $K_C = e(aP, bP)^c$

Common agreed key of A, B, C is

$$K_{\mathsf{ABC}} = K_A = K_B = K_C = e(P, P)^{abc}.$$

Security : Secure against passive adversary under the assumption that BDH problem is hard.

Efficiency : # rounds = 1, # group elements sent = 3, # elliptic curve (EC) scalar multiplication = 3, # pairing computation = 3, # group exponentiation in $G_2 = 3$.

5.2 Extending Joux's Protocol to Multi Party Key Agreement

(Barua, Dutta, Sarkar, [2], 2003)

Assumption : Decisional hash bilinear Diffie-Hellman (DHBDH) problem hard.

The public parameters are $\mathsf{params} = (G_1, G_2, e, q, P)$, $G_1 = \langle P \rangle$, $H : G_2 \to Z_q^*$ is a hash function.

procedure CombineThree (3-group DH protocol)

Consider three user sets U_1, U_2, U_3 with $s_1, s_2, s_3 \in Z_q^*$ respectively as their private keys. Let $\mathsf{Rep}(U_i)$ be the representative of the user set U_i.

$\mathsf{Rep}(U_1)$ sends s_1P to all members of both U_2, U_3;
$\mathsf{Rep}(U_2)$ sends s_2P to all members of both U_1, U_3;
$\mathsf{Rep}(\mathsf{U}_3)$ sends s_3P to all members of both U_1, U_2;

each member of U_1 computes $H(e(s_2P, s_3P)^{s_1})$;
each member of U_2 computes $H(e(s_1P, s_3P)^{s_2})$;
each member of U_3 computes $H(e(s_1P, s_2P)^{s_3})$;

Common agreed key of user sets U_1, U_2, U_3 is $H(e(P, P)^{s_1 s_2 s_3})$;

procedure CombineTwo (2-group DH protocol)

Consider two user sets U_1, U_2 with $s_1, s_2 \in Z_q^*$ respectively as their private keys and $\mathsf{Rep}(U_i)$ is the representative of the user set U_i.

$\mathsf{Rep}(U_1)$ generates $\overline{s} \in Z_q^*$ at random and sends $\overline{s}P$ to the rest of the users;

$\mathsf{Rep}(U_1)$ sends s_1P to all members of U_2;
$\mathsf{Rep}(U_2)$ sends s_2P to all members of U_1;

each member of U_1 computes $H(e(s_2P, \overline{s}P)^{s_1})$;
each member of U_2 computes $H(e(s_1P, \overline{s}P)^{s_2})$;

Common agreed key of user sets U_1, U_2 is $H(e(P,P)^{s_1 s_2 \overline{s}})$;

procedure KeyAgreement (n-party key agreement protocol)

Next we describe the tree structure KeyAgreement as a top down recursive procedure which uses the above two subroutines CombineTwo and CombineThree.

```
procedure KeyAgreement($m, U[i+1,\ldots,i+m]$)
    if ($m = 1$) then
        $KEY = s[i+1]$;
    end if
    if ($m = 2$) then
        call CombineTwo($U[i+1,i+2], s[i+1,i+2]$);
        Let $KEY$ be the agreed key between user sets $U_{i+1}, U_{i+2}$;
    end if
    $n_0 = 0$; $n_1 = \lfloor \frac{m}{3} \rfloor$; $n_3 = \lceil \frac{m}{3} \rceil$; $n_2 = m - n_1 - n_3$;
    $j = 1$ to $3$ do
        call KeyAgreement($n_j, U[i+n_{j-1}+1,\ldots,i+n_{j-1}+n_j]$);
        $\widehat{U}_j = U[i+n_{j-1}+1,\ldots,i+n_{j-1}+n_j]$; $\widehat{s}_j = KEY$; $n_j = n_{j-1}+n_j$;
    end do;
    call CombineThree($\widehat{U}[1,2,3], \widehat{s}[1,2,3]$);
    Let KEY be the agreed key among user sets $\widehat{U}_1, \widehat{U}_2, \widehat{U}_3$;
end KeyAgreement
```

The start of the recursive protocol KeyAgreement is made by the following two statements:

1. $U_j = j$ for $1 \leq j \leq n$;
2. **call** KeyAgreement($n, U[1,\ldots,n]$);

Let $p = \lfloor \frac{n}{3} \rfloor$ and $r = n \bmod 3$.
The set of users $U = \{1,2,\ldots,n\}$ is partitioned into three user sets U_1, U_2, U_3 with cardinality p,p,p respectively if $r = 0$ or with cardinality $p,p,p+1$ respectively if $r = 1$ or with cardinality $p, p+1, p+1$ respectively if $r = 2$.
This top down procedure is used recursively for further partitioning. Essentially a ternary tree structure is obtained. The lower level 0 consists of singleton users having a secret key. Key agreement is done by invoking CombineTwo for user sets of two users and CombineThree for user sets of three users in the key tree. With this tree structure, CombineTwo is never invoked above level 1.

Security : Secure against passive adversary under the assumption that DHBDH problem is hard.

Efficiency : # rounds = $\lceil \log_3 n \rceil$, # group elements (of G_1) sent = $n\lceil \log_3 n \rceil$, # EC scalar multiplication $< \frac{5}{2}(n-1)$, # pairing computation = $n\lceil \log_3 n \rceil$, # group exponent in $G_2 = n\lceil \log_3 n \rceil$, # hash function ($H$) evaluation = $n\lceil \log_3 n \rceil$.

6 Threshold Schemes

6.1 Threshold Signature Scheme

The idea behind the (t, n)-threshold cryptosystem approach is to distribute secret information (*i.e.* the secret key) and computation (*i.e.* signature generation or decryption) among n parties in order to remove single point failure. The goal is to allow a subset of more than t players to jointly reconstruct a secret and perform the computation while preserving security even in the presence of an active adversary which can corrupt upto t (a threshold) parties. The secret key is distributed among n parties with the help of a trusted dealer or without it by running an interactive protocol among all parties.

(Boldyreva, [4], 2003)

Assumption : GDH group.

KeyGen : G_1, G_2 both multiplicative group, $|G_1| = |G_2| = q$, $e : G_1 \times G_1 \rightarrow G_2$ is a bilinear map, $H : \{0,1\}^* \rightarrow G_1$ a hash function.
There are n servers $u_i, 1 \leq i \leq n$. The private key $x \in Z_q^*$ is shared among these users using Shamir's secret sharing scheme such that any subset S of $t+1$ servers can reconstruct x using Lagrange interpolation :

$$x = \sum_{i \in S} L_i x_i,$$

x_i is the private key share and $y_i = g^{x_i}$ is the public key share of user u_i.

Signature Share Generation : To sign a message $m \in \{0,1\}^*$, user u_i outputs $\sigma_i = H(m)^{x_i}$.

Signature Share Verification : Given m, σ_i, y_i, anyone can check whether user u_i is honestly behaving in giving it's share σ_i of signature by checking

$$e(g, \sigma_i) = e(y_i, H(m)).$$

If σ_i passes through this test, call it an acceptable share.

Signature Reconstruction : Suppose a set S of $(t+1)$ honest servers are found and accordingly $(t+1)$ acceptable shares $\sigma_i, i \in S$. The resulting signature on m is $\sigma = \prod_{i \in S} \sigma_i^{L_i}$.
The correctness of the scheme is easy to verify since

$$e(g, \sigma) = e(H(m), g^x).$$

Security : Secure in the random oracle model against an adversary which is allowed to corrupt any $t < n/2$ players under the assumption that the underlying group is GDH.

6.2 Pairing Based (t, n)-Threshold Decryption

(Libert, Quisquater [27], 2003)

Assumption : BDH problem hard.

The following scheme is a threshold adaption of the Boneh-Franklin IBE scheme where a fixed PKG plays the role of a trusted dealer.

Setup : $\mathsf{params} = \langle G_1, G_2, q, e, H_1, H_2, P, P_{pub}^{(1)}, \ldots, P_{pub}^{(n)}, P_{pub} \rangle$

1. Generates two groups G_1 (additive), G_2 (multiplicative) of same prime order q, $e : G_1 \times G_1 \to G_2$ is a bilinear map, G_1 is generated by P, $s \in_R Z_q^*$ is the master key.
2. Choose a $(t-1)$-degree polynomial

$$f(x) = s + a_1 x + \cdots + a_{t-1} x^{t-1}$$

for random $a_1, \ldots, a_{t-1} \in Z_q^*$.
3. For $i = 1, 2, \ldots, n$, compute $P_{pub}^{(i)} = f(i)P \in G_1$ and $P_{pub} = sP$.

Before requesting his private share, each player can check that

$$\sum_{i \in S} L_i P_{pub}^{(i)} = P_{pub}$$

for any subset $S \subset \{1, \ldots, n\}$ such that $|S| = t$ where L_i denotes the appropriate Lagrange coefficient.

KeyGen : Given a user's identity ID, the PKG playes the role of the trusted dealer. For $i = 1, \ldots, n$, it delivers $d_{\mathsf{ID}_i} = f(i) Q_{\mathsf{ID}} \in G_1$ to player i. After receiving d_{ID_i}, player i checks

$$e(P_{pub}^{(i)}, Q_{\mathsf{ID}}) = e(P, d_{\mathsf{ID}_i}).$$

If verification fails, he complains to the PKG that issues a new share.

Encrypt : Given message m and identity ID,
1. compute $Q_{\mathsf{ID}} = H_1(\mathsf{ID})$
2. choose a random $r \in Z_q^*$
3. set the ciphertext to be

$$C = \langle rP, m \oplus H_2(e(P_{pub}, Q_{\mathsf{ID}})^r) \rangle.$$

Decryption Share Generation : When receiving $\langle U, V \rangle$, player i computes his decryption share $e(U, d_{\mathsf{ID}_i})$ and gives it to the recombiner who may be a designated player.

Recombination : The recombiner selects a set $S \subset \{1, \ldots, n\}$ of t acceptable share $e(U, d_{\mathsf{ID}_i})$ and computes

$$g = \prod_{i \in S} e(U, d_{\mathsf{ID}_i})^{L_i}.$$

Once he has g, he recovers the plaintext

$$m = V \oplus H_2(g).$$

Correctness of the scheme is easy to verify since

$$g = e(rP, \sum_{i \in S} L_i d_{\mathsf{ID}_i}) = e(rP, sQ_{\mathsf{ID}}) = e(P_{pub}, Q_{\mathsf{ID}})^r.$$

To check publicly the share of a player is acceptable or not, do the following :
each player chooses a random $R \in G_1$ and computes $w_1 = e(P, R), w_2 = e(U, R)$ and

$$h = H(e(U, d_{\mathsf{ID}_i}), e(P_{pub}, Q_{\mathsf{ID}}), w_1, w_2).$$

After that player i computes $V = R + hd_{\mathsf{ID}_i} \in G_1$ and joins the tuple (w_1, w_2, h, V) to it's share. The other players can check that

$$e(P, V) = e(P, R)e(P_{pub}^{(i)}, Q_{\mathsf{ID}})^h$$

$$e(U, V) = e(U, R)e(U, d_{\mathsf{ID}_i})^h.$$

If this test fails, player i is a dishonest player.

Security : This threshold IBE scheme is provably secure against chosen plaintext attacks in the ID-based setting under the BDH assumption.

6.3 ID-based (t, n)-Threshold Decryption

(Baek, Zheng [1], 2003)

Assumption : BDH problem hard.

Consider a situation where Alice wishes to send a confidential message to a committee in an organization. Alice can first encrypt the message using the identity of the committee and then send over the ciphertext. Let us assume that Bob who is the committee's president has created the identity and hence has obtained a matching private decryption key from the PKG. Preparing for the time when Bob is away, he can share his private key out among a member of decryption server in such a way that any committee member can successfully decrypt the ciphertext if and only if the committee member obtains a certain number of decryption shares from the decryption servers. *i.e.* Bob himself plays the role of a trusted dealer.

The following scheme provide the feature that a user who obtained a private key from the PKG can share the key among decryption servers at will. After key generation, the PKG can be closed. Also this protocol achieves chosen ciphertext security under BDH assumption.

KeyGen : $\mathsf{params} = (G_1, G_2, q, e, P, H_1, H_2, H_3, H_4, P_{pub})$, $P_{pub} = sP$, $x \in_R Z_q^*$ is the master key of PKG.

Extract : Given an identity ID, compute $Q_{\mathsf{ID}} = H_3(\mathsf{ID}), D_{\mathsf{ID}} = xQ_{\mathsf{ID}}$ and returns D_{ID}.

Private Key Distribution : Given a private key D_{ID}, n decryption shares and a threshold parameter $t \leq n$, pick randomly $R_1, R_2, \ldots, R_{t-1} \in G_1^*$ and compute

$$F(u) = D_{\mathsf{ID}} + uR_1 + u^2R_2 + \ldots + u_{t-1}R_{t-1}$$

for $u \in \{0\} \cup N$. Compute $S_i = F(i), y_i = e(S_i, P), 1 \leq i \leq n$ and sends (S_i, y_i) secretly to server $\Gamma_i, 1 \leq i \leq n$. Γ_i then keeps S_i as secret while it publishes y_i.

Encrypt : Given a plaintext $m \in \{0,1\}^l$, identity ID,
1. Choose $r \in Z_q^*$ at random and set $U = rP$.
2. Compute $Q_{\mathsf{ID}} = H_3(\mathsf{ID}), d = e(Q_{\mathsf{ID}}, P_{pub})$ and $\kappa = d^r$.
3. Compute $V = H_1(\kappa) \oplus m, W = rH_2(U, V)$.
4. Set the ciphertext to be $C = (U, V, W)$.

Decryption Share Generation : Given a ciphertext $C = (U, V, W)$, decryption server Γ_i with secret key S_i computes $H_2 = H_2(U, V)$ and checks if $e(P, W) = e(U, H_2)$.
If the test holds then compute

$$\kappa_i = e(S_i, U), \widetilde{\kappa}_i = e(Q_i, U), \widetilde{y}_i = e(Q_i, P), \lambda_i = H_4(\kappa_i, \widetilde{\kappa}_i, \widetilde{y}_i), L_i = Q_i + \lambda_i S_i,$$

where Q_i is chosen randomly from G_1. Output $\delta_i = (i, \kappa_i, \widetilde{\kappa}_i, \widetilde{y}_i, L_i)$.

Decryption Share Verification : Given a ciphertext $C = (U, V, W)$ and a decryption share $\delta_i = (i, \kappa_i, \widetilde{\kappa}_i, \widetilde{y}_i, L_i)$, compute $\lambda_i = H_4(\kappa_i, \widetilde{\kappa}_i, \widetilde{y}_i)$. Check if

$$\frac{e(L_i, U)}{\kappa_i^{\lambda_i}} = \widetilde{\kappa}_i, \quad \frac{e(L_i, P)}{y_i^{\lambda_i}} = \widetilde{y}_i.$$

If the above test holds, then share δ_i of server Γ_i is an acceptable share. Given acceptable shares $S_j, j \in S \subseteq \{1, \ldots, n\}$ where $|S| \geq t$, D_{ID} can be recovered as follows :

$$D_{\mathsf{ID}} = F(0) = \sum_{j \in S} c_{oj} S_j,$$

c_{0j} are appropriate Lagrange co-efficients.

Share Combining : Given a ciphertext $C = (U, V, W)$ and a set of decryption shares $\{\delta_j\}_{j \in S \subseteq \{1,2,\ldots,n\}}$ where $|S| \geq t$, compute $H_2 = H_2(U, V)$, check if $e(P, W) = e(U, H_2)$. If C passes this test (*i.e.* C is a valid ciphertext), compute $\kappa = \prod_{j \in S} \kappa_j^{c_{0j}}$ and $m = H_1(\kappa) \oplus V$. Output m.

The correctness of the scheme is easy to verify since

$$\prod_{j \in S} \kappa_j^{c_{0j}} = \prod_{j \in S} e(S_j, U)^{c_{0j}} = e(\sum_{j \in S} c_{0j} S_j, U) = e(\sum_{j \in S} c_{0j} S_j, rP) = e(D_{\mathsf{ID}}, P)^r.$$

Security : This protocol achieves chosen ciphertext security in the random oracle model under BDH assumption.

7 Miscellaneous Applications

7.1 Key Sharing Scheme :

(Sakai, Ohgishi, Kasahara [34], 2000)

Assumption : BDH problem hard,

The idea of Key Sharing Scheme is quite simple :
Suppose a PKG has a master key s, and it issues private keys to users of the form sP_y, where $P_y = H_1(\mathsf{ID}_y)$ and ID_y is the identity of user y.
Then users y and z have a shared secret that only they (and the PKG) may compute, namely

$$e(sP_y, P_z) = e(P_y, P_z)^s = e(P_y, sP_z).$$

They may use this shared secret to encrypt their communications. This key sharing scheme is non-interactive and can be viewed as a type of "dual-identity-based encryption", where the word "dual" indicates that the identities of both the sender and the recipient (rather than just the recipient) are required as input into the encryption and decryption algorithm.
# pairings = 2 for each key agreement.

7.2 ID-based Chameleon Hashes from Bilinear Pairings :

(Zhang, Safavi- Naini, Susilo [41], 2003)

A chameleon hash function is a trapdoor one-way hash function : without knowledge of the associated trapdoor, the chameleon hash function is resistant to the computation of pre-images and collisions. However, with the knowledge of the trapdoor, collisions are efficiently computable.

Scheme 1 :

Setup : PKG chooses a random number $s \in Z_q^*$ and sets $P_{pub} = sP$. Public parameters are

$$\mathsf{params} = (G_1, G_2, e, q, P, P_{pub}, H_0, H_1).$$

The master key s is kept secret by PKG.

Extract : A user submits his identity ID to PKG which computes the public key as $Q_{\mathsf{ID}} = H_0(\mathsf{ID})$ and returns $S_{\mathsf{ID}} = sQ_{\mathsf{ID}}$ to the user as his private key.

Hash : Given a mesage m, choose a random element R from G_1, define the hash as

$$\mathsf{Hash}(\mathsf{ID}, m, R) = e(R, P)e(H_1(m)H_0(\mathsf{ID}), P_{pub}).$$

Forge :

$$\mathsf{Forge}(\mathsf{ID}, S_{\mathsf{ID}}, m, R, m') = R' = (H_1(m) - H_1(m'))S_{\mathsf{ID}} + R.$$

The forgery is right because

$$\mathsf{Hash}(\mathsf{ID}, m', R') = e(R', P)\ e(H_1(m')H_0(\mathsf{ID}), P_{pub})$$

$$= e((H_1(m) - H_1(m'))S_{\mathsf{ID}} + R, P)\ e(H_1(m')H_0(\mathsf{ID}), P_{pub})$$

$$= e((H_1(m) - H_1(m'))S_{\mathsf{ID}}, P)\ e(R, P)\ e(H_1(m')H_0(\mathsf{ID}), P_{pub})$$

$$= e((H_1(m) - H_1(m'))H_0(\mathsf{ID}), P_{pub})\ e(R, P)\ e(H_1(m')H_0(\mathsf{ID}), P_{pub})$$

$$= e(R, P)\ e(H_1(m)H_0(\mathsf{ID}), P_{pub})$$

$$= \mathsf{Hash}(\mathsf{ID}, m, R)$$

Security : Simantically secure and resistant to collision forgery under active attacks provided BLS signature scheme is secure.

Efficiency : Using precomputation for $a = e(P, P)$ and $b = e(H_0(\mathsf{ID}), P_{pub})$, to compute the chameleon hash of a message m, the sender requires only 1 EC scalar multiplication of G_1 + 2 group exponentiation in G_2. *i.e.* $R = rP, \mathsf{Hash}(\mathsf{ID}, m, R) = a^r b^{H_1(m)}$.

Scheme 2 :

Setup : Public parameters are $\mathsf{params} = (G_1, G_2, e, q, P, P_{pub}, H_0, H_1)$, $P_{pub} = sP$ where $s \in_R Z_q^*$ is the master key to be kept secret.

Extract :

$$S_{\mathsf{ID}} = \frac{1}{s + H_0(\mathsf{ID})} P$$

Hash :

$$\mathsf{Hash}(\mathsf{ID}, m, R) = e(P, P)^{H_1(m)} e(H_0(\mathsf{ID}) + P_{pub}, R)^{H_1(m)}$$

Forge :

$$\mathsf{Forge}(\mathsf{ID}, S_{\mathsf{ID}}, m, R, m') = R' = H_1(m')^{-1}((H_1(m) - H_1(m'))S_{\mathsf{ID}} + H_1(m)R).$$

The forgery is correct because

$$\mathsf{Hash}(\mathsf{ID}, m', R') = e(P, P)^{H_1(m')}\ e(H_0(\mathsf{ID}) + P_{pub}, R')^{H_1(m')}$$

$$= e(P, H_1(m')P)\ e(H_0(\mathsf{ID}) + P_{pub}, H_1(m')H_1(m')^{-1}((H_1(m) - H_1(m'))S_{\mathsf{ID}} + H_1(m)R))$$

$$= e(P, H_1(m')P)\ e(H_0(\mathsf{ID}) + P_{pub}, (H_1(m) - H_1(m'))S_{\mathsf{ID}}) e(H_0(\mathsf{ID}) + P_{pub}, H_1(m)R)$$

$$= e(P, H_1(m')P)\ e(P, (H_1(m) - H_1(m'))P)\ e(H_1(\mathsf{ID}) + P_{pub}, H_1(m)R))$$

$= e(P,P)^{H_1(m)}\ e(H_1(\mathsf{ID}) + P_{pub}, R)^{H_1(m)}$

$= \mathsf{Hash}(\mathsf{ID}, m, R).$

Security : Semantically secure and resistant to collision forgery under active attacks, provided ZSS signature scheme is secure.

Efficiency : Precomputing $a = e(P,P)$, to compute the chameleon hash of a message m, the sender only needs to compute 1 EC scalar multiplication of G_1 + 1 group exponentiation in G_2. *i.e.* $R = rS_{\mathsf{ID}}, \mathsf{Hash}(\mathsf{ID}, m, R) = a^{(r+1)H_1(m)}$.

Applications : ID-based chameleon hash functions can be used to construct ID-based chameleon signature schemes which achieves the goal of ID-based undeniable signature and is non-interactive. An ID-based chameleon signature scheme is an ID-based signature computed over the ID-based chameleon hash of m under the identity of the intended recipient. The recipient can verify that the signature of a certain message m is valid, but can not prove to others that the signer actually signed m and not another message. Indeed, the recipient can find collisions of the chameleon hash function, thus finding a message different from m which would pass the signature verification procedure.

7.3 Signcryption Schemes

The idea of this primitive is to perform encryption and signature in a single logical step in order to obtain confidentiality, integrity, authentication and non-repudiation more efficiently than the sign-then-encrypt approach. The drawback of this latter situation is to expand the final ciphertext size and increase the sender and receiver's computing time which may be impractical for low bandwidth network. Malone-Lee [30] defines extended security notions for ID-based signcryption schemes.

7.3.1 Identity-Based Signcryption

(Malone-Lee [30], 2003)

Assumption : BDH problem hard.

Setup : $t \xleftarrow{R} Z_q^*$, $Q_{\mathsf{TA}} = tP$.
The public parameters are $\mathsf{params} = (G_1, G_2, e, q, n, P, Q_{\mathsf{TA}}, H_1, H_2, H_3)$. The master key t generated by the trusted party TA is kept secret.

Extract(ID) **:** $Q_{\mathsf{ID}} = H_1(\mathsf{ID})$, $S_{\mathsf{ID}} = tQ_{\mathsf{ID}}$.

Signcrypt$(S_{\mathsf{ID}_a}, \mathsf{ID}_b, m)$ **:**
$Q_{\mathsf{ID}_b} = H_1(\mathsf{ID}_b)$, $x \xleftarrow{R} Z_q^*$, $U = xP$, $r = H_2(U||m)$, $W = xQ_{\mathsf{TA}}$, $V = rS_{\mathsf{ID}_a} + W$, $y = e(W, Q_{\mathsf{ID}_b})$, $\kappa = H_3(y)$, $c = \kappa \oplus m$, $\sigma = (c, U, V)$

Unsigncrypt $(\mathsf{ID}_a, S_{\mathsf{ID}_b}, \sigma)$ **:**
$Q_{\mathsf{ID}_a} = H_1(\mathsf{ID}_a)$. Parse σ as (c, U, V), $y = e(S_{\mathsf{ID}_b}, U)$, $\kappa = H_3(y)$, $m = \kappa \oplus c$, $r = H_2(U||m)$.

Return m if and only if $e(V, P) = e(Q_{\mathsf{ID}_a}, Q_{\mathsf{TA}})^r \ e(U, Q_{\mathsf{TA}})$.

Consistency constraint : if $\sigma = \mathsf{Signcrypt}(S_{\mathsf{ID}_a}, \mathsf{ID}_b, m)$, then $m = \mathsf{Unsigncrypt}(\mathsf{ID}_a, S_{\mathsf{ID}_b}, \sigma)$.

This scheme is the result of a combination of the simplified version of Boneh and Franklin's IBE cryptosystem with a varient of Hess's identity based signature.

Security : This protocol achieves the security IND-ISC-CCA (indistinguishability of identity-based signcryptions under chosen ciphertext attack) and also the security EF-ISC-ACMA (existentially unforgeability of identity-based signcryptions under adaptive chosen message attack) in the random oracle model assuming BDH problem is hard.

Efficiency : The size of the cryptogram is $n + 2|G_1|$ when a message of n-bit is sent. The computation requirement for Signcrypt is 1 pairing evaluation + 3 EC scalar multiplication in G_1 and for Unsigncrypt is 4 pairing enaluation + 1 group exponentiation in G_2.

7.3.2 A new Identity-Based Signcryption :

(Libert, Quisquater [28], 2003)

Assumption : DBDH problem hard.

Setup : $s \xleftarrow{R} Z_q^*$, $P_{pub} \leftarrow sP$. The public parameters are $\mathsf{params} = \langle G_1, G_2, e, n, q, P, P_{pub}, H_1, H_2, H_3 \rangle$ and the master key s is kept secret. Choose a secure symmetric cipher (E, D).

Extract(ID) : $Q_{\mathsf{ID}} = H_1(\mathsf{ID})$, $S_{\mathsf{ID}} = sQ_{\mathsf{ID}}$.

Signcrypt($S_{\mathsf{ID}_a}, \mathsf{ID}_b, m$) :
$Q_{\mathsf{ID}_b} = H_1(\mathsf{ID}_b)$, $x \xleftarrow{R} Z_q^*$, $\kappa_1 = e(P, P_{pub})^x$, $\kappa_2 = H_2(e(P_{pub}, Q_{\mathsf{ID}_b})^x)$, $c = E_{\kappa_2}(m)$, $r = H_3(c, \kappa_1)$, $S = xP_{pub} - rS_{\mathsf{ID}_a}$, $\sigma = (c, r, S)$.

Unsigncrypt($\mathsf{ID}_a, S_{\mathsf{ID}_b}, \sigma$) :
$Q_{\mathsf{ID}_a} = H_1(\mathsf{ID}_a)$, Parse σ as (c, r, S), $\kappa_1 = e(P, S) \ e(P_{pub}, Q_{\mathsf{ID}_a})^r$, $\tau = e(S, Q_{\mathsf{ID}_b}) \ e(Q_{\mathsf{ID}_a}, S_{\mathsf{ID}_b})^r$, $\kappa_2 = H_2(\tau)$, $m = D_{\kappa_2}(c)$.
Accept if and only if $r = H_3(c, \kappa_1)$.

Security : This protocol achieves IND-ISC-CCA security for confidentiality and also EF-ISC-ACMA security for unforgeability in the random oracle model assuming DBDH problem is hard.

7.4 Identification Scheme based on GDH

(Kim, Kim, [25], 2002)

Assumption : GDH group.

Identification scheme is a very important and useful cryptographic tool. It is an interactive protocol where a prover, $\mathcal{P}$, tries to convince a verifier, $\mathcal{V}$, of his identity. Only $\mathcal{P}$ knows the secret value corresponding to his public one, and the secret value allows to convince $\mathcal{V}$ of his identity.

KeyGen : The public parameter is $\mathsf{params} = (G_1, G_2, e, q, n, P, aP, bP, cP, v)$ where $v = e(P,P)^{abc}$ and the secret parameter is $\mathsf{sec} = (a, b, c)$.

Protocol actions between $\mathcal{P}$ and $\mathcal{V}$: This scheme consists of several rounds each of these is performed as follows :
1. $\mathcal{P}$ chooses randomly $r_1, r_2, r_3 \in Z_q^*$ and computes $x = e(P,P)^{r_1 r_2 r_3}$, $Q_1 = r_1 P$, $Q_2 = r_2 P$ and $Q_3 = r_3 P$ and sends $\langle x, Q_1, Q_2, Q_3 \rangle$ to $\mathcal{V}$.
2. $\mathcal{V}$ pickd $w \in Z_q^*$ at random and sends w to $\mathcal{P}$.
3. $\mathcal{P}$ computes $y = e(wP, P)^{abc}\ e(P,P)^{r_1 r_2 r_3}$ and sends to $\mathcal{V}$; $\mathcal{V}$ accepts if $y = v^w\ x$ and rejects otherwise.

Security : Secure against active attacks assuming that the underlying group is a GDH group.

8 Conclusion

Several cryptographic primitives using pairings have been described in this survey. All the presented encryption schemes, varities of signature schemes, key agreement schemes, key sharing schemes, chameleon hash, threshold schemes, signcryption scheme and identification schemes have proper security proofs in the existing security models [7] [9] [4] [8] [5] [27] [28] [25] [12] to the best of our knowledge.

References

[1] J. Baek, Y. Zheng. *Identity-Based Threshold Decryption.* Cryptology ePrint Archive, Report 2003/164, available at http://eprint.iacr.org/2003/164, PKC 2004, to appear.

[2] R. Barua, R. Dutta, P. Sarkar. *Extending Joux Protocol to Multi Party Key Agreement.* Indocrypt 2003, Also available at http://eprint.iacr.org/2003/062.

[3] P. S. L. M. Berreto, H. Y. Kim and M. Scott. *Efficient algorithms for pairing-based cryptosystems.* Advances in Cryptology - Crypto '2002, LNCS 2442, Springer-Verlag (2002), pp. 354-368.

[4] A. Boldyreva. *Efficient Threshold Signature, Multisignature and Blind Signature Schemes Based on the Gap-Diffie-Hellman-Group Signature Scheme.* PKC 2003, LNCS 2139, pp. 31-46, Springer-Verlag, 2003.

[5] A. Boldyreva, A. Palacio and B. Warinschi. *Secure Proxy Signature Schemes for Delegation of Signing Rights.* Cryptology ePrint Archive, Report 2003/096, available at http://eprint.iacr.org/2003/096.

[6] D. Boneh, G. D. Crescenzo, R. Ostrovsky, G. Persiano. *Searchable Public Key Encryption.* Cryptology ePrint Archive, Report 2003/195, available at http://eprint.iacr.org/2003/195.

[7] D. Boneh, M. Franklin. *Identity Based Encryption from the Weil Pairing.* SIAM J. of Computing, Vol. 32, No. 3, pp. 586-615, 2003, Extended Abstract in Crypto 2001.

[8] D. Boneh, C. Gentry, B. Lynn and H. Shacham. *Aggregate and Verifiably Encrypted Signature from Bilinear Maps.* Eurocrypt 2003, LNCS 2248, pp. 514-532, Springer-Verlag, 2003.

[9] D. Boneh, B. Lynn, H. Shacham. *Short Signatures from the Weil Pairing.* In Proceedings of Asiacrypt 2001.

[10] D. Boneh, I. Mironov, V. Shoup. *A secure Signature Scheme from Bilinear Maps.* CT-RSA-2003, 98-110.

[11] D. Boneh, A. Silverberg. *Applications of Multilinear forms to Cryptography*, Report 2002/080, http://eprint.iacr.org, 2002.

[12] E. Bresson, O. Chevassut, D. Pointcheval. *Dynamic Group Diffie-Hellman Key Exchange under Standard Assumptions*, Advances in Cryptology - Eurocrypt 2002, LNCS 2332, Springer-Verlag, 2002, pp. 321-336.

[13] D. Chaum, H. V. Antwerpen. *Undeniable Signaturs*, Advances in Cryptology - Crypto '89, LNCS 435, pp 212-217, Springer-Verlag, 1989.

[14] X. Chen, F. Zhang, K. Kim. *A New ID-Based Group Signature Scheme from Bilinear Pairings.* Cryptology ePrint Archive, Report 2003/116, available at http://eprint.iacr.org/2003/116.

[15] C. Cocks. *An Identity Based Encryption Scheme based on Quadratic Residues.* In Cryptography and Coding, LNCS 2260, pp. 360-363, Springer-Verlag, 2001.

[16] W. Diffie, M. Hellman. *New Directions In Cryptography.* IEEE Transaction on Information Theory, IT-22 (6) : 644-654, November, 1976.

[17] Y. Dodis, L. Reyzin. *Breaking and Repairing Optimistic Fair Exchange from PODC 2003.* Cryptology ePrint Archive, available at http://eprint.iacr.org/2003.

[18] G. Frey, H. Ruck. *A remark concerning m-divisibility and the discrete logarithm in the divisor class group of curves*, Mathematics of Computation, 62, pp. 865-874, 1994.

[19] E. Fujisaki, T. Okamoto. *Secure Integration of Asymmetric and Symmetric Encryption Schemes*, in Advances in Cryptology - Crypto'99, LNCS 1666, Springer-Verlag, 1999.

[20] S. Galbraith, K. Harrison and D. Soldera. *Implementing the Tate Pairing.* Algorithm Number Theory Symposium - ANTS V, LNCS 2369, Springer- Verlag (2002), pp. 324-337.

[21] C. Gentry and A. Silverberg, *Hierarchical ID-Based Cryptography.* In Y. Zheng, editor, Proceedings of Asiacrypt 2002, LNCS, Springer-Verlag, 2002.

[22] F. Hess. *Efficient Identity Based Signature Schemes Based on Pairings.* SAC 2002, LNCS 2595, pp. 310-324, Springer-Verlag, 2000.

[23] A. Joux. *A One Round Protocol for Tripartite Diffie-Hellman.* Proceedings of ANTS 4, LNCS 1838, pp. 385-394, 2000.

[24] A. Joux, K. Nguyen. *Seperating DDH from DH in cryptographic groups.* available from eprint.iacr.org.

[25] M. Kim, K. Kim. *A New Identification Scheme Based on the Gap Diffie-Hellman Problem.* SCIS 2002.

[26] M. Kim, K. Kim. *A New Identification Scheme Based on the Bilinear Diffie-Hellman Problem.* ACISP, 2002.

[27] B. Libert, J. J. Quisquater. *Efficient Revocation and Threshold Pairing Based Cryptosystems.* PODC 2003, pp. 163-171.

[28] B. Libert, J. J. Quisquater. *New Identity Based Signcryption Schemes from Pairings.* Cryptology ePrint Archive, Report 2003/023, available at http://eprint.iacr.org/2003/023.

[29] A. Lysyanskaya. *Unique Signatures and Verifiable random functions from DH-DDH Seperation.* Proceedings of Crypto 2002, pp. 597-612, 2002.

[30] J. Malone-Lee. *Identity-Based Signcryption.* Cryptology ePrint Archive, Report 2002/098, available at http://www.iacr.org/2002/098.

[31] A. Menezes, T. Okamoto, S. Vanstone. *Reducing Elliptic Curve Logarithms to Logarithms in a finite field*, IEEE, Transaction of Information Theory, 39 : 1639-1646, 1993.

[32] K. G. Paterson. *ID-Based Signatures from Pairings on Elliptic Curves.* Electron. Lett., Vol. 38, No. 18, pp. 1025-1026, 2002. Also available at http://eprint.iacr/org/2002/004.

[33] R. Sakai and M. Kasahara. *Cryptosystems Based on Pairing over Elliptic Curve.* SCIS 2003, 8c-1, Jan. 2003. Japan. Also available at http://eprint.iacr.org/2003/054.

[34] R. Sakai, K. Ohgishi and M. Kasahara. *Cryptosystems Based on Pairing.* SCIS 2000-c20, Okinawa, Japan, January 2000.

[35] C. P. Schnorr. *Security of Blind Discrete Log Signatures Against Interactive Attacks.* ICICS 2001, LNCS 2229, pp. 1-12, Springer-Verlag, 2001.

[36] A. Shamir. *Identity-based Cryptosystems and Signature Schemes* in Proc. Crypto '84, LNCS Vol. 196, Springer, pp. 47-53, 1985.

[37] F. Zhang, K. Kim. *ID-Based Blind Signature and Ring Signature from Pairings.* Advances in Cryptology - Asiacrypt 2002, LNCS Vol. 2510, Springer-Verlag, 2002.

[38] F. Zhang, R. Safavi-Naini and W. Susilo. *An Efficient Signature Scheme from Bilinear Pairings and it's Applications.* PKC 2004, to appear.

[39] F. Zhang, R. Safavi-Naini and W. Susilo. *Efficient Verifiably Encrypted Signature and Partially Blind Signature from Bilinear Pairings.* In Proceedings of Indocrypt 2003, LNCS, Springer-Verlag, 2003.

[40] F. Zhang, R. Safavi-Naini and W. Susilo. *Non-Interactive Deniable Ring Authentication*, manuscript, 2003.

[41] F. Zhang, R. Safavi-Naini and W. Susilo. *ID-Based Chameleon Hashes from Bilinear Pairings*. Cryptology ePrint Archive, Report 2003/208, available at http://www.iacr.org/2003/208.