Pairing-Based Cryptographic Protocols : 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

Contents

Abstract

The bilinear pairing such as Weil pairing or Tate pairing on elliptic and hyperelliptic curves have recently been found applications in design of cryptographic protocols. 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 [40]. 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 bilinear pairings, namely Weil pairing and Tate pairing of algebraic curves were used in cryptography for the MOV attack [35] using Weil pairing and FR attack [22] using Tate pairing. These attacks reduce the discrete logarithm problem on some elliptic or hyperelliptic curves to the discrete logarithm problem in a finite field. In recent years, bilinear pairings have found positive application in cryptography to construct new cryptographic primitives. The current work is an attempt to survey this field.

Protocols from pairings can be broadly classified into two types:

– Construction of primitives which can not be constructed using other techniques (ex: ID-based encryption, non-trivial aggregate signature *etc*).

– Construction of primitives which can be constructed using other techniques, but for which pairings provide improved functionality (ex: Joux's three-party key agreement, threshold scheme, searchable public key encryption *etc*).

Joux [27], in 2000, showed that the Weil pairing can be used for "good" by using it in a protocol to construct three-party one-round Diffie-Hellman key aggrement. This was one of the breakthroughs in key agreement protocols. After this, Boneh and Franklin [11] presented in Crypto 2001 an ID-based encryption scheme based on properties of bilinear pairings on elliptic curves which is the first fully functional, efficient and provably secure identity-based encryption scheme. In Asiacrypt 2001, Boneh, Lynn and Shacham proposed a basic signature scheme using pairing, the BLS [13] scheme, that has the shortest length among signature schemes in classical cryptography. Subsequently numerous cryptographic schemes based on BLS signature scheme were proposed.

Apart from the three fundamental cryptographic primitives: encryption, signature and key agreement, there are protocol designs for signcryption, threshold decryption, key sharing, identification scheme, chameleon hashes *etc*. We provide the following classification of the protocols:

1. Encryption: Encryption schemes are used for the purpose of achieving privacy and confidentiality. In recent years, pairings made ID-based public key encryption feasible. 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, on receiving receives the encrypted message, obtains his private key from a third party called a Private Key generator (PKG) after authenticating himself to PKG and decrypts the message. The private key that PKG generates on Bob's query is a function of it's master key and Bob's identity.

2. Signature:
– *Short Signature*: These are required in environments with space and bandwidth constraints. When a human is asked to manually key in the signature, the shortest possible signature is needed.
– *Blind Signature*: Blind signatures play a central role in digital cash schemes. A user can obtain from a bank a digital coin using a blind signature protocol. The coin is essentially a token properly signed by the bank. The blind signature protocols enable a user to obtain a signature from a signer so that the signer does not learn any 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. The concept of blind signatures provides anonymity of users in applications such as electronic voting, electronic payment systems *etc.*
– *Multisignature*: Multisignature scheme allows any subgroup of a group of users to jointly sign a message such that a verifier is convinced that each member of the subgroup participated in signing. The goal of multisignature is to prove that each member of the stated subgroup signed the message and the size of this subgroup can be arbitrary. It is up to a particular application to decide what subgroup is required to sign a message. A verifier might reject a multisignature not because it's invalid, but because the verifier in not satisfied with the subgroup which signed the message. Multisignatures can be applied to provide efficient batch verification of several signatures of the same message under different public keys.
– *Aggregate Signature*: Consider n users $U = \{1, 2, \ldots, n\}$. Each user $i \in U$ has a public-private key pair $(\mathsf{PK}_i, \mathsf{SK}_i)$. User i signs message M_i and outputs signature σ_i. A public aggregation algorithm outputs a compressed short signature σ on input all of $\sigma_1, \sigma_2, \ldots, \sigma_n$. This aggregation of n signatures can be done by anyone. Additionally, there is an aggregate verification algorithm that takes $\mathsf{PK}_1, \mathsf{PK}_2, \ldots, \mathsf{PK}_n$, $M_1, M_2, \ldots, M_n$, and σ as input and decides whether the aggregate signature σ is valid. Aggregate signature scheme has use in the secure border gateway protocol for compressing the list of signatures on distinct messages issued by distinct parties.
– *Verifiably Encryption Signature*: These signatures enable user Alice to give Bob a signature on a message M encrypted using a third party's public key and Bob to verify offline that the encrypted signature is valid. Bob can verify that Alice has signed the message, but can not deduce any information about her signature. To enable fair exchange, verifiably encrypted signatures are used in optimistic contract signing protocols.
– *Ring Signature*: Consider a set of n users $U = \{1, 2, \ldots, n\}$. Each user $i \in U$ has a public-private key pair $(\mathsf{PK}_i, \mathsf{SK}_i)$. A ring signature on U is a signature that is constructed using all the public keys of the users in U, and a single private key of any user in U. A ring signature protects the anonymity of a signer since the verifier knows that the signature is from a member of the ring U, but does not know exactly who the signer is. There is also no way to revoke the anonymity of the signer. Ring signatures have applications in authenticated (yet repudiable)

communication and leaking secrets.
– *Group Signature*: Group signatures permits 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. Group signature provides anonymity of users with the property that group manager can identify the signer. In group signature, it is computationally hard to decide whether two different signatures were issued by the same member.
– *Proxy Signature*: A proxy signature allows an entity, called the delegator to delegate its signing rights to another entity, called a proxy signer. The proxy signer signs messages on behalf of the delegator, in case of say, temporal absence, lack of time or computational power, *etc.* 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 [7].
– *Unique Signature*: Unique signature schemes are secure signature schemes where the signature is hard-to-compute function of the public key and the message. Unique signature schemes, also known as invariant signature schemes, are desirable in cryptography and have an important application to construct verifiable random functions (VRFs). VRFs are objects that combine the properties of pseudorandom functions with the verifiability property and can be viewed as a commitment to an arbitrary number of bits.

3. Key Agreement: 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.

4. Threshold: 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. In the (t, n)-threshold scheme, $t \leq n$, there are n users. A secret information is distributed among these n-users. Any subset of more than t users are allowed to reconstruct the secret. The computation is performed preserving security even in the presence of an active adversary that can corrupt up to t users.

5. Miscellaneous:
– *Chameleon Hash*: Chameleon hashing is basically non-interactive commitment scheme. A chameleon hash function is associated with a pair of public-private keys. Anyone who knows the public key, can compute the associated hash function. Without the knowledge of associated trapdoor, the chameleon hash function is collision resistant. However, the trapdoor information holder can easily find collisions for every given input. 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 [17].
– *Signcryption*: A signcryption scheme is a scheme that provides private and authenticated delivery of messages between two parties in a more efficient manner than a straightforward composition of an encryption scheme with a signature scheme. It combines the functionality of signature and encryption. 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*: Identification scheme is another important and useful cryptographic tool where a prover $\mathcal{P}$ interacts with a verifier $\mathcal{V}$ to convince him of his identity. Only $\mathcal{P}$ knows the secret value corresponding to his public one, and this secret information permits to convince $\mathcal{V}$ of his identity.

In this paper, we have tried to survey different cryptographic primitives and include only those schemes which have, to the best of our knowledge, concrete security proofs in the existing adversarial models. Barretto's pairing based crypto lounge [4] is an excellent compilation of existing work on pairing based cryptography. This survey does not consider algebraic theory of pairings nor algorithms to compute them. The rest of our 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

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 $e : G_1^2 \rightarrow G_2$ satisfying the following properties is called a cryptographic bilinear map.

– Bilinearity : $e(aP, bQ) = e(P, Q)^{ab}$ for all $P, Q \in G_1$ and $a, b \in Z_q^*$. This can be restated in the following way. For $P, Q, R \in G_1$, $e(P + Q, R) = e(P, R)\ e(Q, R)$ and $e(P, Q + R) = e(P, Q)\ e(P, R)$.
– 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 [11] and Tate Pairing [5], [24] are examples of cryptographic bilinear maps. Currently, active research is being carried out to obtain efficient algorithms to compute pairings. Our survey excludes this area.

Now we specify some versions of Diffie-Hellman problems. Each problem comes in two flavours : computational followed by decisional. We define the following two terms.

– advantage : When adversary has to distinguish between two probability distribution.
– success probability : When adversary has to find an object of interest.

For a set S, by $a \in_R S$, we mean that a is randomly chosen from S. A function $f(m)$ is said to be *negligible* if it is less than $\frac{1}{m^l}$ for every fixed $l > 0$ and sufficiently large integer m.

Unless otherwise stated, we assume that the messages are arbitrary length finite binary strings and the above setup holds for the cryptographic protocols throughout the paper.

In the subsequent discussion, we formalize advantage of DDH and success probability of CDH problems and describe the corresponding assumptions. For each of the other problems, there is a corresponding assumption which can be formalized in a way similar to the DDH and CDH problems.The following classification of the problems is provided.

2.1 Diffie-Hellman Problems

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

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

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

   CDH assumption : For every probabilistic, polynomial-time, 0/1-valued algorithm $\mathcal{A}$, $\mathsf{Succ}^{\mathsf{CDH}}_{\mathcal{A},G_1}$ is negligible.
   (See sections 4.9, 4.12).

2. Decisional Diffie-Hellman (DDH) problem in G_1 :
   Instance : (P, aP, bP, cP) for some $a, b, c \in Z_q^*$.
   Output : yes if $c = ab \bmod q$ and output no otherwise.
   Comments : 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 [11] : The DLP in G_1 is no harder than the DLP in G_2.

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

   Gap Diffie-Hellman (GDH) group : A prime order group G_1 is a GDH group if there exists an efficient polynomial-time algorithm which solves the DDH problem in G_1 and there is no probabilistic polynomial-time algorithm which solves the CDH problem with non-negligible probability of success. The domains of bilinear pairings provide examples of GDH groups. The MOV reduction provides a method to solve DDH in G_1, whereas there is no known efficient algorithm for CDH in G_1. (See sections 4.1, 4.3, 4.11, 7.4).

3. Weak Diffie-Hellman (W-DH) problem in a group G_1 :
   Instance : (P, Q, sP) for $P, Q \in G_1$ and for some $s \in Z_q^*$.
   Output : sQ.

Comments : W-DH problem is no harder than CDH problem.
(See section 4.10).

4. Reversion of CDH (RCDH) problem in G_1 :
Instance : (P, aP, rP) for some $a, r \in Z_q^*$.
Output : $bP, b \in Z_q^*$ satisfying $a = rb \bmod q$.
Comments : RCDH problem is equivalent to CDH problem in G_1 [18].

5. $(k+1)$-exponent problem ($(k+1)$-EP) in G_1:
Instance : $(P, yP, y^2P, \ldots, y^kP)$for a random $y \in Z_q^*$.
Output : $y^{k+1}P$.
Comments : $(k+1)$-EP is no harder than the CDH problem.
(See section 4.7).

6. k-Diffie-Hellman Inversion (k-DHI) problem in G_1 :
Instance : $(P, yP, y^2P, \ldots, y^kP)$ for a random $y \in Z_q^*$.
Output : $\frac{1}{y}P$.
Comments : k-DHI problem is polynomially equivalent to $(k+1)$-EP.

7. k-Strong Diffie-Hellman (k-SDH) problem in G_1 :
Instance : $(P, yP, y^2P, \ldots, y^kP)$ for a random $y \in Z_q^*$.
Output : $(c, \frac{1}{y+c}P)$ where $c \in Z_q^*$.
Comments : k-SDH problem is a stronger version of k-DHI problem. When c is pre-specified, k-SDH problem is polynomially equivalent to k-DHI. k-SDH problem has a simple random self reduction in G_1.
(See section 4.13).

8. Collusion Attack Algorithm with k-traitors (k-CAA) :
Instance : $(P, yP, h_1, \ldots, h_k \in Z_q^*, \frac{1}{h_1+y}P, \ldots, \frac{1}{h_k+y}P)$ for a random $y \in Z_q^*$.
Output : $\frac{1}{h+y}P$ for some $h \notin \{h_1, \ldots, h_k\}$.
Comments : k-CAA is polynomially equivalent to $(k-1)$-DHI problem.

9. l- Many Diffie-Hellman problem in G_1:
Oracle : $\mathcal{O}_{P,\tilde{y}}(J) = (\prod_{j \in J} y_j)P \in G_1$ where vector $\tilde{y} = (y_1, y_2, \ldots, y_l) \in_R (Z_q^*)^l$ and J is any strict subset of $\{1, 2, \ldots, l\}$.
Instance : $(P, \mathcal{O}_{P,\tilde{y}}, J)$ for any vector $\tilde{y} = (y_1, y_2, \ldots, y_l) \in_R (Z_q^*)^l$ and for all $J \subset \{1, 2, \ldots, l\}$.
Output: $(\prod_{j=1}^{l} y_j)P$.
Comments : $(l-1)$-DHI assumption implies l-Many-DH assumption. This reduction is also valid for the decision version of DHI and Many-DH problems. l-DHI assumption is easier to state than l-Many-DH assumption since there is no need for an oracle.
(See section 4.11).

10. Chosen-target CDH problem in G_1 :
Let s be a random element of Z_q^* and $Q = sP$.
Oracles : 1) A target oracle $\mathcal{T}_{G_1}$ that returns a random element $U_i \in G_1$. 2) A helper oracle $s(\cdot)$ that returns sU on a randomly chosen input $U \in G_1$.
Instance : (q, P, Q, H_1) where $H_1 : \{0,1\}^* \rightarrow G_1^*$ is a cryptographic hash function and access

to the target and helper oracles with at most $q_{\mathcal{T}}$ and $q_{\mathcal{H}}$ queries respectively.
Output : A set V of, say l pairs $((V_1, j_1), (V_2, j_2), \ldots, (V_l, j_l))$, where for all $i, 1 \leq i \leq l$, there exists $j_i, 1 \leq j_i \leq q_{\mathcal{T}}$ such that $V_i = sU_{j_i}$ where all V_i are distinct and $q_{\mathcal{H}} < q_{\mathcal{T}}, l$.
(See section 4.1).

11. Chosen-target Inverse CDH problem in G_1 :
Let s be a random element of Z_q^* and $Q = sP$.
Oracles : 1) A target oracle $\mathcal{T}_{G_1}$ that returns a random element $U_i \in G_1$. 2) A helper oracle $\mathsf{Inv-cdh}-s(\cdot)$ that computes $s^{-1}U$ for a randomly chosen input $U \in G_1$.
Instance : (q, P, Q, H_1) where $H_1 : \{0,1\}^* \rightarrow G_1^*$ is a cryptographic hash function and access to the target and helper oracles with at most $q_{\mathcal{T}}$ and $q_{\mathcal{H}}$ queries respectively.
Output : A set V of, say l pairs $((V_1, j_1), (V_2, j_2), \ldots, (V_l, j_l))$, where for all $i, 1 \leq i \leq l$, there exists $j_i, 1 \leq j_i \leq q_{\mathcal{T}}$ such that $V_i = s^{-1}U_{j_i}$ where all V_i are distinct and $q_{\mathcal{H}} < q_{\mathcal{T}}, l$.

2.2 Bilinear Diffie-Hellman Problems

1. Bilinear Diffie-Hellman (BDH) problem in (G_1, G_2, e) :
Instance : (P, aP, bP, cP) for some $a, b, c \in Z_q^*$.
Output : $e(P, P)^{abc}$.
(See sections 3.1, 3.2, 3.3, 3.4, 5.1, 6.2, 6.3, 7.1, 7.3.1).

2. Decisional Bilinear Diffie-Hellman (DBDH) problem in (G_1, G_2, e) :
Instance : (P, aP, bP, cP, r) for some $a, b, c \in_R Z_q^*$, $r \in_R G_2$.
Output : yes if $r = e(P, P)^{abc}$ and output no otherwise.
(See sections 3.6, 7.3.2).

3. 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 $H : G_2 \rightarrow Z_q^*$.
Output : yes if $r = H(e(P, P)^{abc}) \bmod q$ and output no otherwise.
Comments : The DHBDH problem in (G_1, G_2, e) is a hash version of the decisional BDH problem in (G_1, G_2, e) .
(See section 5.2).

4. k-Bilinear Diffie-Hellman Inversion (k-BDHI) problem in (G_1, G_2, e) :
Instance : $(P, yP, y^2P, \ldots, y^kP)$ for some $y \in Z_q^*$.
Output : $e(P, P)^{\frac{1}{y}} \in G_2$.
Comments : 1-BDHI assumption is polynomially equivalent to the standard BDH assumption. It is not known if the k-BDHI assumption, for $k > 1$, is polynomially equivalent to BDH.

5. k-Decisional Bilinear Diffie-Hellman Inversion (k-DBDHI) problem in (G_1, G_2, e) :
Instance : $(P, yP, y^2P, \ldots, y^kP, r)$ for some $y \in Z_q^*$, $r \in_R G_2$.
Output : yes if $r = e(P, P)^{\frac{1}{y}} \in G_2$ and output no otherwise.
(See section 3.5).

2.3 Miscellaneous Problems

1. ROS problem : (Schnorr)
   Oracle : A random function $F : Z_q^l \rightarrow Z_q$.
   Instance : A system of t equations in l 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 $k = 1, 2, \ldots, t$, $t \geq l + 1$.
   Output : Co-efficients $a_{k,i} \in Z_q^*$ and a solvable subsystem of $l + 1$ equations in the unknowns $c_1, c_2, \ldots, c_l$.
   (See section 4.8).

2. Co-Gap Diffie-Hellman (Co-GDH) group : Consider a cryptographic bilinear map in the following setup :
   a) G_1, G_2 are two additive groups and G_T is a multiplicative group of prime order q;
   b) P_1 is a generator of G_1 and P_2 is a generator of G_2;
   c) ψ is a computable isomorphism from G_1 to G_2, with $\psi(P_1) = P_2$; and
   d) e is an efficiently computable bilinear map $e : G_1 \times G_2 \rightarrow G_T$ satisfying the following properties :
   – *Bilinearity :* For all $Q_1 \in G_1, Q_2 \in G_2$ and $a, b \in Z_q^*$, $e(aQ_1, bQ_2) = e(Q_1, Q_2)^{ab}$.
   – *Non-degeneracy :* $e(P_1, P_2) \neq 1$.

   These properties imply one more : for any $Q_1, Q_2 \in G_1$, $e(Q_1, \psi(Q_2)) = e(Q_2, \psi(Q_1))$. (Such bilinear maps can be derived from Weil pairing and Tate pairing; for simplicity the reader may assume $G_1 = G_2$). We refer this setup as the Co-GDH setup. With this setup, we obtain natural generalizations of the CDH and DDH problems :

   Computational Co-Diffie-Hellman (Co-CDH) problem :
   Instance : (P_1, P_2, aP_1, bP_2) for some $a, b \in Z_q^*$.
   Output : $abP_2 \in G_2$.

   Decisional Co-Diffie-Hellman (Co-DDH) problem :
   Instance : $(P_1, P_2, aP_1, bP_2, cP_2)$ for some $a, b, c \in Z_q^*$.
   Output : yes if $c = ab \bmod q$ and output no otherwise.

   When $G_1 = G_2$ and $P_1 = P_2$, these problems reduced to the standard CDH and DDH problems respectively.

   Groups G_1, G_2 are said to be Co-GDH groups if there exists an efficient algorithm to solve the Co-DDH problem and there is no polynomial-time (in $|q|$) algorithm to solve the Co-CDH problem. The existence of a cryptographic bilinear map ensures the existence of Co-GDH groups. (See sections 4.4, 4.5, 4.6).

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 derives from Bob's identifying information. Bob, after receiving the encrypted message, obtains his private key from a third party called a Private Key generator (PKG) after authenticating himself to PKG and can then 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.
Shamir [40] introduced this concept of identity-based cryptosystem. The first ID-based encryption was proposed by Boneh and Franklin [11] in 2001 that uses bilinear pairing.
The advantage of ID-based encryption are compelling. It makes maintaining authenticated public key directories 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 perform online lookup of public keys or public parameters.
Some disadvantages of ID-based system are : (1) the PKG knows Bob's private key, *i.e.* key escrow 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 certifying authority (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.

3.1 ID-Based Encryption Scheme

(Boneh, Franklin, [11], 2001)

• Protocol Description :
– *Setup* : Choose $s \in_R Z_q^*$ and set $P_{pub} = sP$. Choose cryptographic hash functions $H_1 : \{0,1\}^* \to G_1^*$ and $H_2 : G_2 \to \{0,1\}^n$, n is the bit length of messages. The master key is s and the global public key is P_{pub}.

– *Extract* : Given a public identity $\mathsf{ID} \in \{0,1\}^*$, compute the public key $Q_{\mathsf{ID}} = H_1(\mathsf{ID}) \in G_1$ and the private key $S_{\mathsf{ID}} = sQ_{\mathsf{ID}}$. The computation $Q_{\mathsf{ID}} = H_1(\mathsf{ID})$ maps an arbitrary string to a point of the group G_1. This operation is called Map-to-point and is more expensive than computation of usual message digest.

– *Encrypt* : Choose a random $r \in Z_q^*$, 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

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

• Assumption :
BDH problem is hard.

•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 [23] transformation.

• Efficiency :
– *Setup* : 1 scalar multiplication in G_1.
– *Extract* : 1 Map-to-point hash operation; 1 scalar multiplication in G_1.
– *Encrypt* : 1 Map-to-point hash operation; 1 scalar multiplication in G_1; 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, [10], 2003)

Suppose 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. Suppose Bob sends an 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 referred 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 SPKE scheme using bilinear map :

• Protocol Description :
– *KeyGen* : Choose $s \in_R Z_q^*$ and set $P_{pub} = sP$. The secret key is s and the public key is P_{pub}. Let K be the set of all keywords and $H_1 : K \to G_1$, $H_2 : G_2 \to Z_q^*$ be two hash functions.

– *SPKE* : Given a keyword W and the public key P_{pub}, choose a random $r \in Z_q^*$ and output

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

– *Trapdoor* : Given a keyword W and the secret key s, output $T_W = sH_1(W)$.

– *Test* : Given Trapdoor T_W, a SPKE $S = \langle U, V \rangle$ and the public key P_{pub}, test if $V = H_2(e(T_W, U))$. If true, output yes, else output no.

• Assumption :
BDH problem is hard.

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

• Efficiency :
– *KeyGen* : 1 scalar multiplication in G_1.
– *SPKE* : 1 Map-to-point hash operation; 1 scalar multiplication in G_1; 1 hash function (H_2) evaluation; 1 pairing computation; 1 group exponent in G_2.
– *Trapdoor* : 1 scalar multiplication in G_1.
– *Test* : 1 pairing computation; 1 hash function (H_2) evaluation.

3.3 Hierarchical ID-Based Encryption (HIDE) Scheme

(Gentry, Silverberg, [25], 2002)

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 only needs to obtain the public parameters of Bob's parent PKG (and Bob's identifying 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.

• Protocol Description : BasicHIDE :
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 chooses an arbitrary generator $P_0 \in G_1$, picks a random $s_0 \in Z_q^*$ and sets $Q_0 = s_0 P_0$. Let $H_1 : \{0,1\}^* \rightarrow G_1$ and $H_2 : G_2 \rightarrow \{0,1\}^n$ be two cryptographic hash functions. 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 root PKG's secret is $s_0 \in Z_q^*$ and global public key is (P_0, Q_0).

– *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 $1 \leq i \leq t$ 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 $(t \geq 1)$ 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 $Q_i = s_i P_0$ for $1 \leq i \leq t-1$.

– *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 $1 \leq i \leq t$,
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 [11] 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 $H_1(\mathsf{ID}_1, \ldots, \mathsf{ID}_i)$.

• Assumption :
BDH problem is hard.

• Security :
Chosen ciphertext security of this basic scheme is obtained by using Fujisaki-Okamoto [23] padding in the random oracle model under the assumption that BDH problem is hard.

• Efficiency :
– *Setup* : 1 scalar multiplication in G_1.
– *Extract* : 1 Map-to-point hash operation; 2 scalar multiplications in G_1; 1 addition in G_1.
– *Encrypt* : For an identity at level t, t scalar multiplications in G_1; 1 Map-to-point hash operation; 1 hash function (H_2) evaluation; 1 group exponent in G_2; 1 XOR operation; 1 pairing computation.
– *Decrypt* : For an identity at level t, t pairing computations; 1 hash function (H_2) evaluation; 1 XOR operation.
The bit-length of the ciphertext and the complexity of decryption grow linearly with the level of the message recipient.

3.4 Dual-HIDE : Dual-Hierarchical-Identity-Based Encryption

(Gentry, Silverberg [25], 2002)

• Protocol Description :
Suppose two users, Alice and Bob, have the ID-tuples $(\mathsf{ID}_{y_1}, \ldots, \mathsf{ID}_{y_l}, \ldots, \mathsf{ID}_{y_m})$ and $(\mathsf{ID}_{z_1}, \ldots, \mathsf{ID}_{z_l}, \ldots, \mathsf{ID}_{z_n})$ respectively, where $(\mathsf{ID}_{y_1}, \ldots, \mathsf{ID}_{y_l}) = (\mathsf{ID}_{z_1}, \ldots, \mathsf{ID}_{z_l})$.
In other words, Alice is in Level_m, Bob is in Level_n and their common ancestor upto Level_l are same.
Alice may use Dual-HIDE to encrypt a message to Bob as follows :

– *Encrypt :* To encrypt $M \in \mathcal{M}$, Alice :
1. computes $P_{z_i} = H_1(\mathsf{ID}_{z_1}, \ldots, \mathsf{ID}_{z_i}) \in G_1$ for $l+1 \leq i \leq n$,
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 Alice's secret point, S_{y_l} is the secret point of Alice's and Bob'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 Alice'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, Bob computes :

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

• Assumption :
BDH problem is hard.

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

• Efficiency :
– *Encrypt* : 1 Map-to-point hash operation; $(n-l+1)$ scalar multiplications in G_1; 1 hash function (H_2) evaluation; 1 XOR operation; $(m-l+1)$ pairing computation; 1 group exponent in G_2.
– *Decrypt* : $(n-l+1)$ pairing computation; 1 hash function (H_2) evaluation; 1 XOR operation.
If Alice and Bob have a common ancestor below the root PKG, then the ciphertext is shorter than for normal HIDE. Further, using Dual HIDE, the encrypter Alice computes $(m-l+1)$ pairings while the decrypter Bob 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. Furthermore, one can restrict key escrow using Dual-HIDE.

3.5 ID-Based Encryption Scheme Without Random Oracle

(Boneh, Boyen [9], 2004)

• Protocol Description :
– *Setup* : The public keys (ID) are assumed to be elements of Z_q^* and messages are elements of G_2. Select random elements $x, y \in Z_q^*$ and set $U = xP, V = yP$. The public parameters are (U, V) and the master key is (x, y).

– *Extract* : Given a public key $\mathsf{ID} \in Z_q^*$, choose a random $r \in Z_q^*$ and compute $K = \frac{1}{\mathsf{ID}+x+ry} P \in G_1$. Output the private key $S_{\mathsf{ID}} = (r, K)$.

– *Encrypt* : To encrypt a message $M \in G_1$ under public key $\mathsf{ID} \in Z_q^*$, pick a random $s \in Z_q^*$ and output the ciphertext

$$C = \langle s(\mathsf{ID})P + sU, sV, e(P,P)^s \ M \rangle.$$

– *Decrypt* : To decrypt a ciphertext $C = \langle X, Y, Z \rangle$ using the private key $S_{\mathsf{ID}} = (r, K)$, output $Z/e(X + rY, K)$.

• Assumption :
q-DBDHI problem is hard.

• Security :
Secure against selective-ID adaptive chosen ciphertext attack without random oracles under q-DBDHI assumption.

• Efficiency :
– *Setup* : 2 scalar multiplications.
– *Extract* : 1 inversion in Z_q^*; 1 scalar multiplication in G_1.
– *Encrypt* : 4 scalar multiplications in G_1; 1 group exponent in G_1; 1 multiplication in G_2.
Note that $e(P, P)$ can be precomputed once and for all so that encryption requires no pairing computation.
– *Decrypt* : 1 scalar multiplication in G_1; 1 addition in G_1; 1 inversion in G_2.

3.6 Hierarchical ID-Based Encryption (HIBE) Scheme Without Random Oracle

(Boneh, Boyen [9], 2004)

• Protocol Description :
– *Setup* : The public keys (ID) of depth l are assumed to be vectors of elements in Z_q^l. The j-th component for an identity $\mathsf{ID} = (\mathsf{ID}_1, \ldots, \mathsf{ID}_l) \in Z_q^l$ corresponds to the identity at level j.
The system parameters for an HIBE of maximum depth l is generated as follows :
Choose a random $\alpha \in Z_q^*$ and set $P_1 = \alpha P \in G_1$.
Choose random elements $h_1, \ldots h_l \in G_1$ and another generator $P_2 \in G_1^*$. The public parameters are $(P, P_1, P_2, h_1, \ldots, h_l)$ and master key is αP_2.

For $j = 1, \ldots, l$, define $F_j(x) = xP_1 + h_j$.
The messages are assumed to be elements of G_2.

– *Extract* : For an identity $\mathsf{ID} = (\mathsf{ID}_1, \ldots, \mathsf{ID}_j) \in Z_q^j$ of depth $j \leq l$, pick random $r_1, \ldots, r_j \in Z_q$ and set the private key

$$S_{\mathsf{ID}} = (\alpha P_2 + \sum_{k=1}^{j} r_k F_k(\mathsf{ID}_k), r_1 P, \ldots, r_j P).$$

Note that, if at depth $(j-1)$, the private key for identity $\mathsf{ID}_{|j-1} = (\mathsf{ID}_1, \ldots, \mathsf{ID}_{j-1}) \in Z_q^{j-1}$ is $S_{\mathsf{ID}|j-1} = (d_0, \ldots, d_{j-1})$, then the private key S_{ID} for ID is generated by choosing randomly $r_j \in Z_q$ and setting $S_{\mathsf{ID}} = (d_0 + r_j F_j(\mathsf{ID}_j), d_1, \ldots, d_{j-1}, r_j P)$.

– *Encrypt* : To encrypt a message $M \in G_2$ under the public key $\mathsf{ID} = (\mathsf{ID}_1, \ldots, \mathsf{ID}_j) \in Z_q^j$, pick randomly $s \in Z_q^*$ and output

$$C = \langle e(P_1, P_2)^s \ M, sP, sF_1(\mathsf{ID}_1), \ldots, sF_j(\mathsf{ID}_j) \rangle.$$

– *Decrypt* : Consider an identity $\mathsf{ID} = (\mathsf{ID}_1, \ldots, \mathsf{ID}_j)$. To decrypt a ciphertext $C = \langle A, B, C_1, \ldots, C_j \rangle$ using the private key $S_{\mathsf{ID}} = (d_0, d_1, \ldots, d_j)$, output

$$A \ \prod_{k=1}^{j} e(C_j, d_j)/e(B, d_0) = M.$$

• Assumption :
DBDH problem is hard.

• Security :
Secure against selective-ID adaptive chosen ciphertext attack without random oracles under DBDH assumption.

• Efficiency :
– *Setup* : 2 scalar multiplications in G_1.
– *Extract* : For an identity at depth j, $(2j+1)$ scalar multiplications in G_1; $(j+1)$ additions in G_1.
– *Encrypt* : 1 group exponent in G_2; 1 multiplications in G_2; $(j-1)$ scalar multiplications in G_1.
Note that encryption does not require any pairing computation as $e(P_1, P_2)$ can be precomputed once and included in the system parameters.
– *Decrypt* : For an identity at depth j, j multiplications in G_2; j pairing computations; 1 inversion in G_2.

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 [40] 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 schemes, verification function is easily obtained from the identity, possibly the same key and the same underlying computation primitives can be used. Boneh, Lynn, Shacham [13] proposed a pairing based short signature scheme in 2001. This was followed by a large number of pairing based signature schemes for different applications.

4.1 BLS Short Signature Scheme

(Boneh, Lynn, Shacham, [13], 2001)

Short signatures are needed in environments with space and bandwidth constraints. For example, when a human is asked to type in a digital signature the shortest possible signatures are desired. Two most frequently used signature schemes are RSA and DSA. If one uses 1024 bit modulus, RSA signatures are 1024 bit long and standard DSA or ECDSA (elliptic curve DSA) signatures are 320 bit long. These signatures are too long to be keyed. The following signature scheme provides short signature of length approximately 160 bits with a level of security similar to 320 bit DSA signatures.

• Protocol Description :
– *KeyGen* : Let $H : \{0,1\}^* \rightarrow G_1$ be a Map-to-point hash function. The secret key is $x \in_R Z_q^*$ and the public key is $P_{pub} = xP$ for a signer.

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

– *Verify* : Given public key $P_{pub} = xP$, a message m and a signature σ, verify $e(P, \sigma) = e(P_{pub}, H(m))$.

• Assumption :
Existence of GDH group.

• Security :
Secure against existential forgery under adaptive chosen message attack in the random oracle model assuming CDH problem is hard in G_1.

• Efficiency :
– *KeyGen* : 1 scalar multiplication in G_1.
– *Sign* : 1 Map-to-point hash operation; 1 scalar multiplication in G_1.
– *Verify* : 1 Map-to-point hash operation; 2 pairing computations.

4.2 Blind Signature Scheme

(Boldyreva [6], 2003)

Blind signatures are the basic tools of digital cash schemes. The goal of a blind signature protocol is to enable a user to obtain a signature from a signer so that the signer does not learn any 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.

• Protocol Description :
– *KeyGen* : Let $H : \{0,1\}^* \rightarrow G_1$ be a Map-to-point hash function. The secret key is $x \in_R Z_q^*$ and the public key is $P_{pub} = xP$ for a signer.

– *Blind Signature Issuing Protocol* : Given secret key x and a message $m \in \{0,1\}^*$,

  – (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 public key P_{pub}, a message m and a signature σ, verify $e(P_{pub}, H(m)) = e(P, \sigma)$.

• Assumption :
Chosen-target CDH problem is hard.

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

• Efficiency :
– *KeyGen* : 1 scalar multiplication in G_1.
– *Blind Signature Issuing Protocol* : 1 Map-to-point hash operation; 3 scalar multiplications in G_1.
– *Verify* : 2 pairing computations; 1 Map-to-point hash operation.

4.3 Multisignature Scheme

(Boldyreva, [6], 2003)

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.

• Protocol Description :
– *KeyGen* : Let $H; \{0,1\}^* \rightarrow G_1$ be a Map-to-point hash function. Consider a set U of n users. The secret key is $x_i \in_R Z_q^*$ and the public key is $P_{pub_i} = x_i P$, for user $u_i \in U, 1 \leq i \leq n$.

– *Multisignature Creation* : Any user $u_i \in U$ with secret key x_i that wishes to participate in signing a message $m \in \{0,1\}^*$, computes $\sigma_i = x_i H(m)$ 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. After getting all the σ_j for $j \in J = \{i_1, \ldots, i_l\}$, D computes the multisignature $\sigma = \sum_{j \in J} \sigma_j$ and outputs (m, L, σ).

– *Multisignature Verification* : Given $T = (m, L, \sigma)$ and the list of public keys of the users in L : $P_{pub_j} = x_j P, j \in J = \{i_1, \ldots, i_l\}$, the verifier computes $P_{pub_L} = \sum_{j \in J} P_{pub_j} = \sum_{j \in J} x_j P$ and verifies

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

• Assumption :
Existence of GDH group.

• Security :
Secure against existential forgery under chosen message attack in the random oracle model under the assumption that the CDH problem is hard in G_1.

• Efficiency :
– *KeyGen* : n scalar multiplications in G_1.
– *Multisignature Creation* : If $l \leq n$ users are participating in signing, then 1 Map-to-point hash operation; l scalar multiplications in G_1; $(l-1)$ additions in G_1.
– *Multisignature Verification* : If number of users in the list L is l, then $(l-1)$ additions in G_1; 2 pairing computations.

4.4 Aggregate Signature

(Boneh, Gentry, Lynn, Shacham [12], 2003)

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 indeed signed message m_i , $1 \leq i \leq n$.

• Protocol Description :
– *KeyGen* : Consider the Co-GDH setup. Let U be a set of n users and $H : \{0,1\}^* \rightarrow G_2$ be a Map-to-point hash function. The secret key is $x_i \in_R Z_q^*$ and the public key is $P_{pub_i} = x_i P_1$ for 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 $\sigma_i = x_i H(m_i)$, $1 \leq i \leq n$. The messages m_i must be all distinct. The aggregate signature is $\sigma = (\sigma_1 + \sigma_2 + \cdots + \sigma_n) \in G_2$.

– *Aggregate verification* : Given public keys P_{pub_i}, distinct messages $m_i, 1 \leq i \leq n$ and an aggregate signature σ, verify $e(P_1, \sigma) = \prod_{i=1}^{n} e(P_{pub_i}, H(m_i))$.

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

• Security :
Secure against existential forgery in the aggregate chosen key model assuming that the Co-CDH problem is hard in (G_1, G_2).

• Efficiency :
– *KeyGen* : n scalar multiplications in G_1.

– *Aggregation* : n Map-to-point hash operations; n scalar multiplications in G_2; $(n-1)$ additions in G_2.
– *Aggregate verification* : n Map-to-point hash operations; $(n+1)$ pairing computations.

4.5 The Bilinear Verifiably Encrypted Signature

(Boneh, Gentry, Lynn, Shacham [12], 2003)

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.

• Protocol Description :
– *KeyGen* : Consider the Co-GDH setup. Let $H : \{0,1\}^* \rightarrow G_2$ be a Map-to-point hash function. Choose $x, x' \in_R Z_q^*$ and set $P_{pub} = xP_1, P'_{pub} = x'P_1$. The private/public key pair for signer is (x, P_{pub}) and that of the adjudicator is (x', P'_{pub}).

– *Sign, Verify* : For a message $m \in \{0,1\}^*$, the signature of a signer with private key x is $\sigma = xH(m) \in G_2$ and the verification is $e(P_1, \sigma) = e(P_{pub}, H(m))$.

– *Verifiably Encrypted Signature Creation* : Given a secret key $x \in Z_q^*$, a message $m \in \{0,1\}^*$ and an adjudicator's public key $P'_{pub} \in G_1$, compute $h = H(m) \in G_2$ and $\sigma = xh$. Select a random $r \in Z_q^*$ and set $\mu = r\psi(P_1)$ and $\sigma' = r\psi(P'_{pub})$. Aggregate σ, σ' as $w = (\sigma + \sigma') \in G_2$ and output the pair (w, μ).

– *Verifiably Encrypted Signature Verification* : Given a public key P_{pub}, a message m, an adjudicator's public key P'_{pub} and a verifiably encrypted signature (w, μ), set $h = H(m)$; accept if $e(P_1, w) = e(P_{pub}, h)\ e(P'_{pub}, \mu)$ holds.

– *Adjudication* : Given an adjudicator's public key P'_{pub} and corresponding private key $x' \in_R Z_q^*$, a public key P_{pub} and a verifiably encrypted signature (w, μ) on some message m, ensure the verifiably encrypted signature is valid; then compute $\sigma = w - x'\mu$.
(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. Adjudicator involves only during signature revelation phase.

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

• Security :
Secure against existential forgery and aggregate extraction assuming that Co-GDH [13] signature

scheme is secure against existential forgery and extraction respectively. Co-GDH signature scheme is infact the BLS signature scheme in Co-GDH setup.

• Efficiency :
– *KeyGen* : 2 scalar multiplications in G_1.
– *Sign* : 1 Map-to-point hash operation; 1 scalar multiplication in G_2.
– *Verify* : 1 Map-to-point hash operation; 2 pairing computations.
– *Verifiably Encrypted Signature Creation* : 1 Map-to-point hash operation; 3 scalar multiplications in G_2; 1 addition in G_2;
– *Verifiably Encrypted Signature Verification* : 1 Map-to-point hash operation; 3 pairing computations; 1 multiplication in G_T.
– *Adjudication* : 1 scalar multiplication in G_2 + 1 inversion in G_2.

4.6 Bilinear Ring Signature

(Boneh, Gentry, Lynn, Shacham [12], 2003)

Consider a set U of n 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.

• Protocol Description :
– *KeyGen :* Consider the Co-GDH setup. Let $H : \{0,1\}^* \rightarrow G_2$ be a Map-to-point hash function. The secret key is $x_i \in_R Z_q^*$ and the public key is $P_{pub_i} = x_i P_1$ for user $u_i \in U$.

– *Ring Signing :* Given public keys $P_{pub_1}, \ldots, P_{pub_n} \in G_1$, a message $m \in \{0,1\}^*$, and a private key x_s for a certain $s, 1 \leq s \leq n$, choose $a_i \in_R Z_q$ for all $i \neq s$, compute $h = H(m) \in G_2$ and set

$$\sigma_s = \frac{1}{x_s}(h - \psi(\sum_{i \neq s} a_i P_{pub_i})).$$

For all $i \neq s$, let $\sigma_i = a_i P_2$. Output the ring signature $\sigma = (\sigma_1, \ldots, \sigma_n) \in G_2^n$.

– *Ring Verification :* Given public keys $P_{pub_1}, \ldots, P_{pub_n} \in G_1$, a message $m \in \{0,1\}^*$, and a ring signature σ, compute $h = H(m)$ and verify $e(P_1, h) = \prod_{i=1}^{n} e(P_{pub_i}, \sigma_i)$.

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

• Security :
The identity of the signer is unconditionally protected and the scheme is resistant to forgery in the random oracle model assuming that the Co-CDH problem is hard in (G_1, G_2).

• Efficiency :
– *KeyGen* : n scalar multiplications in G_1.

– *Ring Signing* : 1 inversion in Z_q^*; 1 Map-to-point hash operation; $(n-1)$ scalar multiplications in G_2; $(n-1)$ scalar multiplications in G_1; 1 inversion in G_2.
– *Ring Verification* : 1 Map-to-point hash operation; $(n+1)$ pairing computations.

4.7 ZSS Short Signature Scheme

(Zhang, Safavi-Naini, Susilo, [42], 2004)

• Protocol Description :
– *KeyGen* : Let $H : \{0,1\}^* \rightarrow Z_q^*$ be a hash function. The secret key is $x \in_R Z_q^*$ and the public key is $P_{pub} = xP$ for a signer.

– *Sign* : Given a secret key x and a message $m \in \{0,1\}^*$, compute signature $S = \frac{1}{H(m)+x}P$.

– *Verify* : Given a public key P_{pub}, a message m and a signature S, verify $e(H(m)P + P_{pub}, S) = e(P, P)$.

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

• 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 scalar multiplication in G_1.
– *Sign* : 1 inversion in Z_q^*; 1 hash function (H) evaluation; 1 scalar multiplication in G_1.
– *Verify* : 2 pairing computation (one of which, $e(P, P)$ can be precomputed); 1 scalar multiplication in G_1; 1 hash function (H) evaluation; 1 addition in G_1.
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.

4.8 ID-Based Blind Signature Scheme (Schnorr type)

(Zhang, Kim [41], 2002)

• Protocol Description :
– *Setup :* Let $H : \{0,1\}^* \rightarrow G_1$ be a Map-to-point hash function. Consider another hash function $H_1 : \{0,1\}^* \times G_2 \rightarrow Z_q$. Choose $s \in_R Z_q^*$ and set $P_{pub} = sP$. The master key is s and the global public key is P_{pub}.

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

– *Blind Signature Issuing Protocol :* Given a signer's private key S_{ID} and a message $m \in \{0,1\}^*$,

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

• Assumption :
ROS-problem is hard.

• Security :
Secure against one more forgery in the random oracle model under the assumption that ROS problem is hard.

• Efficiency :
– *Setup* : 1 scalar multiplication in G_1.
– *Extract* : 1 Map-to-point hash operation; 1 scalar multiplication in G_1.
– *Blind Signature Issuing Protocol* : 6 scalar multiplications in G_1; 1 pairing computation; 1 hash function (H) evaluation; 1 addition in Z_q; 4 additions in G_1; 1 inversion in Z_q.
– *Verification* : 1 hash (H) function evaluation; 2 pairing computations; 1 exponentiation in G_2.

4.9 ID-Based Ring Signature

(Zhang, Kim [41], 2002)

• Protocol Description :
– *Setup :* Let $H_1 : \{0,1\}^* \to G_1$ be a Map-to-point hash function and $H : \{0,1\}^* \to Z_q^*$ be another hash function. Choose $s \in_R Z_q^*$ and set $P_{pub} = sP$. The master key is s and the global public key is P_{pub}.

– *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 $L = \{\mathsf{ID}_i : 1 \le i \le n\}$ be the set of identities. The real signer's identity ID_k is listed in L.

– *Signing :* Given signer's private key S_{ID_k} and a message $m \in \{0,1\}^*$,

–(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 $i = k+1, \ldots n-1, 0, 1, \ldots, k-1$, 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 $T_k = A - c_k S_{\mathsf{ID}_k}$.

–(Output the ring signature) : 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.

• Assumption :
CDH problem is hard.

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

• Efficiency :
– *Setup* : 1 scalar multiplication in G_1.
– *Extract* : 1 Map-to-point hash operation; 1 scalar multiplication in G_1.
– *Signing* : n hash function (H) evaluation; $(2n-1)$ pairing computations.
– *Verification* : $2n$ pairing computations; n hash function (H) evaluation.

4.10 ID-Based Signature from Pairing

(Hess, [26], 2002)

• Protocol Description :
– *Setup* : Choose $s \in_R Z_q^*$ and set $P_{pub} = sP$. The master key is s and the global public key is P_{pub}. Let $H_1 : \{0,1\}^* \to G_1$ be a Map-to-point hash function and $H : \{0,1\}^* \times G_2 \to Z_q^*$ be another hash function.

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

– *Sign* : Given a secret key S_{ID} and a message $m \in \{0,1\}^*$, 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. $v = H(m, r)$,

3. $u = vS_{\mathsf{ID}} + kP_1$.
The signature is then the pair $(u, v) \in G \times Z_q^*$.

– *Verify* : Given a public key Q_{ID}, 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 $v = H(m, r)$.

• Assumption :
Weak-DH problem is hard.

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

• Efficiency :
– *Setup* : 1 scalar multiplication in G_1.
– *Extract* : 1 Map-to-point hash operation; 1 scalar multiplication in G_1.
– *Sign* : 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 G_2, one hash function (H) evaluation and one simultaneous multiplication in the group G_1.

– *Verify* : The verification operation requires one exponentiation in G_2, one hash function (H) 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_1 and one element of Z_q^*.

4.11 Unique Signature Scheme Without Random Oracle

(Lysyanskaya [33], 2002)

Unique signature schemes, also known as invariant signature schemes, are secure signature schemes where the signature is a hard-to-compute function of the public key and the message. One must verify a signature again even if it has been accepted before. Because this time the signature may come from an unauthorized party. If a signature scheme allows the signer to easily 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 [15] 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 [15]. Lysyanskaya proposed a unique signature scheme based on this idea while making use of bilinear pairing. This scheme is proved to be secure in the standard model under Many-DH assumption.

• Protocol Description :
– *KeyGen* : 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 for a signer. Compute $A_{i,b} = a_{i,b}P, 1 \leq i \leq n, b \in \{0,1\}$. The public key for the signer is $P_{pub} = \{A_{i,0}, A_{i,1} | 1 \leq i \leq n\}$.

– *Sign* : Assume that the messages being signed are n-bit codewords of a code of distance Cn, where $0 < C \leq 1/2$ is a constant. Given the secret key and an n-bit codeword $m = m_1 \circ m_2 \circ \ldots \circ m_n$, output the signature

$$\sigma = \{s_{m,i} = (\prod_{j=1}^{i} a_{j,m_j})P : 1 \leq i \leq n\}.$$

– *Verify* : Let $s_{m,0} = 1$. Given the public key P_{pub}, verify that, for all $i, 1 \leq i \leq n$, $e(P, s_{m,i}) = e(s_{m,i-1}, A_{i,m_i})$.

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 P. 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 $l_c = a_{i,0}l_p$, while if it is the right child, its label will be $l_c = a_{i,1}l_p$. 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.

• Assumption :
Existence of GDH group, Many-DH problem is hard.

• 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.

• Efficiency :
– *KeyGen* : $2n$ scalar multiplications in G_1.
– *Sign* : n scalar multiplications in G_1, $(n-1)$ multiplications in Z_q^*.
– *Verify* : $2n$ pairing computations.

4.12 An Authentication-Tree Based Secure Signature Scheme Without Random Oracle

(Boneh, Mironov, Shoup [14], 2003)

In an authentication-tree based scheme, signatures are produced that represent paths connecting messages and the root of the tree. Messages are usually placed in the very bottom level of the tree. The authentication mechanism works inductively : the root authenticates its children, they authenticate their children, and so on, down to the message authenticated by its parent.

• Protocol Description :
– *KeyGen* : Consider a keyed family of collision resistant hash functions $\mathcal{H}_k : \mathcal{M} \rightarrow \{0,1\}^s$ where $\mathcal{M}$ is the message space. 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.

1. Pick randomly $\alpha_i \in Z_q^*, 1 \leq i \leq n$ and $Q \in_R G_1$. Choose a random key k for the collision resistant hash function $\mathcal{H}_k$. Compute $Q_1 = (1/\alpha_1)Q, \ldots, Q_n = (1/\alpha_n)Q \in G_1$.
2. Pick randomly $R \in G_1$. Compute $y = e(R, Q)$.
3. Pick randomly $\beta_0 \in Z_q$. Compute $x_0 = y^{\beta_0}$.
4. The public key for a signer is $(k, Q, Q_1, \ldots, Q_n, y, x_0)$ and the corresponding private key is $(\alpha_1, \alpha_2, \ldots, \alpha_n, \beta_0, R)$.

– *Sign* : Each node in the tree is authenticated with respect to its parent; messages to be signed are authenticated with respect to the leaves, which are selected in sequential order and never reused. To sign i-th message $m \in \mathcal{M}$, the signer generates the i-th 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_j-th child of $x_{j-1} (i_j \in \{1, \ldots, n\})$.

1. $x_j = y^{\beta_j}$ for some $\beta_j \in_R Z_q^*, 1 \leq j \leq l$. The secret β_j is stored for as long as node x_j is an anscestor of the current signing leaf.
2. Compute $f_j = \alpha_{i_j}(\beta_{j-1} + \mathcal{H}_k(x_j))R$. This is the authenticated value of x_j, the i_j-th child of x_{j-1}.
3. Compute $f = (\beta_l + \mathcal{H}_k(m))R$. 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 $\hat{x}_l = e(\hat{f}, Q)y^{-\mathcal{H}_k(m)}$.
2. Compute $\hat{x}_{j-1} = e(\hat{f}_j, Q_{i_j})y^{-\mathcal{H}_k(\hat{x}_j)}$ for $l \leq j \leq 1$.
3. Accept the signature if $\hat{x}_0 = x_0$.

• Assumption :
CDH problem is hard.

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

• Efficiency :
– *KeyGen* : n scalar multiplications in G_1; 1 pairing computation; 1 exponentiation in G_2.
– *Sign* : l exponentiations in G_2; $(l+1)$ hash function $(\mathcal{H}_k)$ evaluations; $(l+1)$ additions in Z_q^*; l multiplications in Z_q^*, l scalar multiplications in G_1.
– *Verify* : $(l+1)$ pairing computations; $(l+1)$ hash function $(\mathcal{H}_k)$ evaluations; $(l+1)$ exponentiations in G_2; $(l+1)$ multiplications in G_2.

4.13 Short Signature Scheme Without Random Oracle

(Boneh, Boyen [8], 2004)

• Protocol Description :
– *KeyGen* : The secret key is $(x, y) \in_R Z_q^* \times Z_q^*$ and the public key is (P, U, V) where $U = xP$ and $V = yP$ for a signer. The messages are assumed to be elements of Z_q^*.

– *Sign :* Given a secret key (x, y), a message $m \in Z_q^*$, choose a random $r \in Z_q^*$ and compute $\sigma = \frac{1}{x+m+yr}P$. Here $\frac{1}{x+m+yr}$ is computed modulo q and the unlikely event $x + m + yr = 0$ is avoided by choosing a different r. The signature is (σ, r).

– *Verify :* Given a public key (P, U, V), a message $m \in Z_q^*$ and a signature (σ, r), verify $e(\sigma, U + mP + rV) = e(P, P)$.

• Assumption :
q-SDH problem is hard.

• Security :
Secure against existential forgery under chosen message attack under SDH assumption and without using the random oracle model.

• Efficiency :
– *KeyGen* : 2 scalar multiplications in G_1.
– *Sign* : 1 inversion in Z_q^*; 1 scalar multiplication in G_1.
– *Verify* : 2 scalar multiplication in G_1; 2 additions in G_1; 2 pairing computations one of which, $e(P, P)$ can be precomputed.

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 [27] 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 large number of cryptographic protocols some of which have been previously mentioned. 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. Authenticated key agreement protocols are the basic tools for group-oriented and collaborative applications such as, distributed simulation, multi-user games, audio or video-conferencing, and also peer-to-peer application that are likely to involve a large number of users. These are used to construct secure channels which are the base for desiging, analyzing and implimenting higher-level protocols in a modular approach. A formal

model of security for group authenticated key agreement can be found in [16]. Much research work remaines to be done in this area.

5.1 Joux's One Round Three Party Key Agreement Protocol

(Joux [27], 2000)

• Protocol Description :
Consider three parties 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 $K_{\mathsf{ABC}} = K_A = K_B = K_C = e(P, P)^{abc}$.

• Assumption :
BDH problem is hard.

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

• Efficiency :
– *Communication* : Round required is 1; group elements (of G_1) sent are 3.
– *Computation* : 3 scalar multiplications in G_1; 3 pairing computations; 3 exponentiations in G_2.

5.2 Extending Joux's Protocol to Multi Party Key Agreement

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

• Protocol Description :
Let $H : G_2 \rightarrow Z_q^*$ be a hash function. Consider the set of n users $U = \{1, 2, \ldots, n\}$. Let $p = \lfloor \frac{n}{3} \rfloor$ and $r = n \bmod 3$. The set U 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 the procedure CombineTwo for user sets of two users and the procedure CombineThree for user sets of three users in the key tree as described below. With this tree structure, CombineTwo is never invoked above level 1.

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 $H(e(P, P)^{s_1 s_2 \overline{s}})$;

• Assumption :
DHBDH problem is hard.

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

• Efficiency :
– *Communication* : Rounds required is $\lceil \log_3 n \rceil$; group elements (of G_1) sent are $n\lceil \log_3 n \rceil$.
– *Computation* : $< \frac{5}{2}(n-1)$ scalar multiplications in G_1; $n\lceil \log_3 n \rceil$ pairing computations; $n\lceil \log_3 n \rceil$ exponentiations in G_2; $n\lceil \log_3 n \rceil$ hash function (H) evaluation.

6 Threshold Schemes

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.

6.1 Threshold Signature Scheme

(Boldyreva, [6], 2003)

• Protocol Description :
– *KeyGen* : Let $H : \{0,1\}^* \rightarrow G_1$ be a Map-to-point hash function. Suppose 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 :

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

where $L_i = \prod_{j \in S} \frac{-x_j}{(x_i - x_j)}$ is the Lagrange co-efficient, x_i is the private key share and $P_{pub_i} = x_i P$ is the public key share of user u_i.

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

– *Signature Share Verification :* Given m, σ_i, P_{pub_i}, anyone can check whether user u_i is honestly behaving in giving it's share σ_i of signature by checking

$$e(P, \sigma_i) = e(P_{pub_i}, H(m)).$$

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 $\sigma = \sum_{i \in S} L_i \sigma_i$.
The correctness of the scheme is easy to verify since

$$e(P, \sigma) = e(H(m), xP).$$

• Assumption :
Existence of GDH group.

• 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.

• Efficiency :
– *KeyGen* : n scalar multiplications in G_1.
– *Signature Share Generation* : For each user, 1 Map-to-point hash operation; 1 scalar multiplication in G_1.
– *Signature Share Verification* : 2 pairing computations; 1 Map-to-point hash operation.
– *Signature Reconstruction* : $(t+1)$ scalar multiplications in G_1; t additions in G_1; $(t+1)$ Lagrange co-efficient (L_i) computations.

6.2 Pairing Based (t, n)-Threshold Decryption

(Libert, Quisquater [31], 2003)

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

• Protocol Description:
– *KeyGen* : Choose a $(t-1)$-degree polynomial $f(x) = s + a_1 x + \cdots + a_{t-1}x^{t-1}$ for random $a_1, \ldots, a_{t-1} \in Z_q^*$. For $i = 1, 2, \ldots, n$, compute $P_{pub}^{(i)} = f(i)P \in G_1$ and $P_{pub} = sP$.
Let $H_1 : \{0,1\}^* \rightarrow G_1$ be a Map-to-point hash function and $H_2 : G_2 \rightarrow \{0,1\}^l$ be another hash function.
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 co-efficient explicitely given by the formula

$$L_i = \prod_{j \in S} \frac{-x_j}{(x_i - x_j)}.$$

Given a user's identity $\mathsf{ID} \in \{0,1\}^*$, 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 which then issues a new share.

– *Encrypt :* Given message $m \in \{0,1\}^l$ and identity ID, compute $Q_{\mathsf{ID}} = H_1(\mathsf{ID})$. Choose a random $r \in Z_q^*$ and 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 whether 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 $h = H(e(U, d_{\mathsf{ID}_i}), e(P_{pub}, Q_{\mathsf{ID}}), w_1, w_2)$. Next, player i computes $V = R + h d_{\mathsf{ID}_i} \in G_1$ 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.

• Assumption :
BDH problem is hard.

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

• Efficiency :
– *KeyGen* : n function (f) evaluation; ($2n$+1) scalar multiplications in G_1; $2n$ pairing computations.
– *Encrypt* : 1 Map-to-point hash operation; 1 hash function (H_2) evaluation; 1 XOR operation; 1 exponentiation in G_2; 1 scalar multiplication in G_1.
– *Decryption Share Generation* : For each share holder, 1 pairing computation.
– *Recombination* : $|S|$ pairing computations; ($|S|-1$) multiplications in G_2; $|S|$ Lagrange co-efficient computations.

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

(Baek, Zheng [1], 2003)

Consider the following scenario : Alice wishes to send a confidential message to a committee in an organization. She can first encrypt the message using the identity of the committee and then send over the ciphertext. Suppose Bob who is the committee's president has created the identity and 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 provides 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 in random oracle model.

• Protocol Description :
– *KeyGen :* PKG chooses $x \in_R Z_q^*$ and computes $P_{pub} = xP$. The master key of PKG is x and the public key is P_{pub}. Consider four hash functions : $H_1 : G_2 \to \{0,1\}^l$, $H_2 : G_1 \times \{0,1\}^l \to G_1$, $H_3 : \{0,1\}^* \to G_1$, $H_4 : G_2^3 \to Z_q^*$. H_3 is a Map-to-point hash function.

– *Extract :* Given an identity $\mathsf{ID} \in \{0,1\}^*$, 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 $\mathsf{ID} \in \{0,1\}^*$, choose $r \in Z_q^*$ at random and set $U = rP$. Compute $Q_{\mathsf{ID}} = H_3(\mathsf{ID}), d = e(Q_{\mathsf{ID}}, P_{pub})$, $\kappa = d^r$, $V = H_1(\kappa) \oplus m$, $W = rH_2(U, V)$ and 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

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

• Assumption :
BDH problem is hard.

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

• Efficiency :
– *KeyGen* : 1 scalar multiplication.
– *Extract* : 1 Map-to-point hash operation; 1 scalar multiplication in G_1.

– *Private Key Distribution* : For each share holder, $(t-1)$ scalar multiplications in G_1; $(t-1)$ additions in G_1; 1 pairing computation.
– *Encrypt* : 1 scalar multiplication; 1 Map-to-point hash operation; 1 pairing computation; 1 exponentiation in G_2.
– *Decryption Share Generation* : For each share holder, 1 hash function H_2 evaluation, 5 pairing computations; 1 hash function H_4 evaluation; 1 scalar multiplication in G_1; 1 addition in G_1.
– *Decryption Share Verification* : 1 hash function H_4 evaluation; 2 pairing computations; 2 exponentiations in G_2.
– *Share Combining* : 1 hash function H_2 evaluation; 2 pairing computations; $|S|$ Lagrange co-efficient computations; $(|S|-1)$ multiplications in G_2; 1 hash function H_1 evaluation; 1 XOR operation.

7 Miscellaneous Applications

7.1 Key Sharing Scheme :

(Sakai, Ohgishi, Kasahara [38], 2000)

• Protocol Description :
Let $H_{:}\{0,1\}^*$ be a Map-to-point hash function.
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 $\mathsf{ID}_y \in \{0,1\}^*$ 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.

• Assumption :
BDH problem is hard.

• Efficiency :
For each party, 1 pairing computation for key sharing; 1 scalar multiplication in G_1; 1 Map-to-point hash operation for private key extraction.

7.2 ID-Based Chameleon Hashes from Bilinear Pairings :

(Zhang, Safavi- Naini, Susilo [45], 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.
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.

Scheme 1 :

• Protocol Description :
– *Setup* : PKG chooses a random number $s \in Z_q^*$ and sets $P_{pub} = sP$. The master key of PKG is s and the public key is P_{pub}. Consider a Map-to-point hash function $H_0 : \{0,1\}^* \rightarrow G_1$ and another hash function $H_1 : \{0,1\}^l \rightarrow Z_q^*$.

– *Extract* : A user submits his identity $\mathsf{ID} \in \{0,1\}^*$ 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 message $m \in \{0,1\}^l$, 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 correct because

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

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

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

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

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

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

• Assumption :
BLS signature scheme is secure.

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

• Efficiency :
– *Setup* : 1 scalar multipication.

– *Extract* : 1 Map-to-point hash operation; 1 scalar multiplication in G_1.
– *Hash* : 2 pairing computations; 1 scalar multiplication in G_1; 1 Map-to-point hash operation; 1 hash function H_1 evaluation. 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.* $R = rP, \mathsf{Hash}(\mathsf{ID}, m, R) = a^r b^{H_1(m)}$.
– *Forge* : 2 hash function H_1 evaluation; 1 scalar multiplication in G_1; 1 subtraction in Z_q^*; 1 addition in G_1.

Scheme 2 :

• Protocol Description:
– *Setup* : PKG chooses a random number $s \in Z_q^*$ and sets $P_{pub} = sP$. The master key of PKG is s and the public key is P_{pub}. Consider two hash functions $H_0 : \{0,1\}^* \to Z_q^*$ and $H_1 : \{0,1\}^l \to Z_q^*$.

– *Extract* : Given an identity $\mathsf{ID} \in \{0,1\}^*$, compute $S_{\mathsf{ID}} = \frac{1}{s+H_0(\mathsf{ID})} P$. S_{ID} is the private key corresponding to the public identity ID.

– *Hash* : Given a message $m \in \{0,1\}^l$, an identity $\mathsf{ID} \in \{0,1\}^*$ and a random element $R \in G_1$, define

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

– *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

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

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

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

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

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

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

• Assumption :
ZSS signature scheme is secure.

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

• Efficiency :
– *Setup* : 1 scalar multiplication in G_1.
– *Extract* : 1 hash function H_0 evaluation; 1 addition in Z_q^*; 1 multiplicative inverse in Z_q^*; 1 scalar multiplication in G_1.
– *Hash* : 2 pairing computations; 1 hash function H_0 evaluation; 2 exponentiations in G_2; 1 addition in G_1. 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.* $R = rS_{\mathsf{ID}}, \mathsf{Hash}(\mathsf{ID}, m, R) = a^{(r+1)H_1(m)}$.
– *Forge* : 2 hash function H_1 evaluations; 2 scalar multiplications in G_1; 1 subtraction in Z_q^*; 1 multiplicative inverse in Z_q^*.

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 [34] defines extended security notions for ID-based signcryption schemes.

7.3.1 Identity-Based Signcryption

(Malone-Lee [34], 2003)

• Protocol Description
– *Setup* : Choose $s \leftarrow^R Z_q^*$ and set $P_{\mathsf{Pub}} = sP$. The master key generated by the trusted party is s and the public key is P_{pub}. Consider three hash functions : $H_1 : \{0,1\}^* \rightarrow G_1$, $H_2 : \{0,1\}^* \rightarrow Z_q^*$ and $H_3 : G_2 \rightarrow \{0,1\}^l$.

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

– *Signcrypt*($S_{\mathsf{ID}_a}, \mathsf{ID}_b, m$) : For a message $m \in \{0,1,\}^l$, compute $Q_{\mathsf{ID}_b} = H_1(\mathsf{ID}_b)$. Choose $x \leftarrow^R Z_q^*$ and set $U = xP$. Compute $r = H_2(U||m)$, $W = xP_{pub}$, $V = rS_{\mathsf{ID}_a} + W$, $y = e(W, Q_{\mathsf{ID}_b})$, $\kappa = H_3(y)$, $c = \kappa \oplus m$. Send $\sigma = (c, U, V)$

– *Unsigncrypt* ($\mathsf{ID}_a, S_{\mathsf{ID}_b}, \sigma$) : Compute $Q_{\mathsf{ID}_a} = H_1(\mathsf{ID}_a)$. Parse σ as (c, U, V). Compute $y = e(S_{\mathsf{ID}_b}, U)$, $\kappa = H_3(y)$, $m = \kappa \oplus c$, $r = H_2(U||m)$. Return m if and only if $e(V, P) = e(Q_{\mathsf{ID}_a}, P_{pub})^r\, e(U, P_{pub})$.

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.

• Assumption :
BDH problem is hard.

• 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 :
– *Setup* : 1 scalar multiplication in G_1.
– *Extract* : 1 Map-to-point hash operation; 1 scalar multiplication in G_1.
– *Signcrypt* : 1 Map-to-point hash operation; 3 scalar multiplications in G_1; 1 hash function H_2 evaluation; 1 pairing computation; 1 hash function H_3 evaluation; 1 XOR operation, 1 addition in G_1.
– *Unsigncrypt* : 1 Map-to-point hash operation; 4 pairing computations; 1 hash function H_3 evaluation; 1 XOR operation; 1 hash function H_2 evaluation; 1 exponentiation in G_2.
The size of the cryptogram is $n + 2|G_1|$ when a message of n-bit is sent.

7.3.2 A New Identity-Based Signcryption :

(Libert, Quisquater [32], 2003)

• Protocol Description :
– *Setup* : Choose $s \leftarrow^R Z_q^*$ and set $P_{pub} \leftarrow sP$. The secret key is s and the public key is P_{pub}. Choose a secure symmetric cipher (E, D) with keyspace K_s and ciphertext space C_s. Also consider three hash functions : $H_1 : \{0,1\}^* \rightarrow G_1$, $H_2 : G_2 \rightarrow K_s$ and $H_3 : C_s \times G_2 \rightarrow Z_q^*$. H_1 is a Map-to-point hash function.

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

– *Signcrypt*($S_{\mathsf{ID}_a}, \mathsf{ID}_b, m$) : For a message $m \in \{0,1\}^l$, compute $Q_{\mathsf{ID}_b} = H_1(\mathsf{ID}_b)$. Choose $x \leftarrow^R Z_q^*$ and set $\kappa_1 = e(P, P_{pub})^x$, $\kappa_2 = H_2(e(P_{pub}, Q_{\mathsf{ID}_b})^x)$. Compute $c = E_{\kappa_2}(m)$, $r = H_3(c, \kappa_1)$, $S = xP_{pub} - rS_{\mathsf{ID}_a}$. Send $\sigma = (c, r, S)$.

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

• Assumption :
DBDH problem is hard.

• 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.

• Efficiency :
– *Setup* : 1 scalar multiplication in G_1.

– *Extract* : 1 Map-to-point hash operation; 1 scalar multiplication in G_1.
– *Signcrypt* : 1 Map-to-point hash operation; 2 pairing computations; 1 hash function H_2 evaluation; 1 exponentiation in G_2; 1 symmetric key encryption; 1 hash function H_3 evaluation; 2 scalar multiplications in G_1; 1 inversion in G_1.
– *Unsigncrypt* : 1 Map-to-point hash operation; 4 pairing computations; 2 exponentiations in G_2; 1 hash function H_2 evaluation; 1 symmetric key decryption; 1 hash function H_3 evaluation.

7.4 Identification Scheme based on GDH

(Kim, Kim, [29], 2002)

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.

• Protocol Description :
– *KeyGen* : Choose randomly $a, b, c \in Z_q^*$ and compute aP, bP, cP, $v = e(P, P)^{abc}$. The secret key is (a, b, c) and make aP, bP, cP, v public.

– *Protocol actions between* $\mathcal{P}$ *and* $\mathcal{V}$: This scheme consists of several rounds, each of which 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.

• Assumption :
Existence of GDH group.

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

• Efficiency :
– *KeyGen* : 3 scalar multiplications in G_1; 1 pairing computations.
– *Protocol actions between* $\mathcal{P}$ *and* $\mathcal{V}$: 3 pairing computations and 4 scalar multiplications in G_1 for $\mathcal{P}$; 1 exponentiation in G_2 and 1 multiplication in G_2 for $\mathcal{V}$.

7.5 Other Signature Schemes

There are a large number of cryptographic protocols that uses pairings. Discussing every prototocol is beyond the scope of the paper. This subsection includes a list of few other interesting signature schemes that have various cryptographic applications in digital world. For details, see the references [3].

1. Optimistic Fair Exchange [21].
2. Non-Interactive Deniable Ring Authentication [44].
3. A New Varifiably Encrypted Signature Scheme [43].
4. Partially Blind Signature Scheme [43].
5. ID-Based Group Signature Scheme [18].
6. Delegation-By-Certificate Proxy Signature Scheme [7].
7. Hierarchical ID-Based Signatures (HIDS) Scheme [25].

8 Conclusion

Several cryptographic primitives using pairings have been described in this survey. Some others have been left out, mainly due to the non-availability of proper security proofs. The area is still growing and almost each conference proceedings include some new proposals. On the other hand, we have covered the basic schemes which will continue to be referred in the future. Thus we believe that our survey will provide both an introduction to the area as well as serve as a ready reference to the area in the next few years.

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] R. Dutta, r. Barua, P. Sarkar. *Pairing Based Cryptography : A survey*. Cryptology ePrint Archive, Report 2004/064, available at http://eprint.iacr.org/2004/064.

[4] P. S. L. M. Berreto. *The Pairing-Based Crypto Lounge*. Available at http://planeta.terra.com.br/informatica/paulobarreto/pblounge.html.

[5] 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.

[6] 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.

[7] 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.

[8] D. Boneh, X. Boyen. *Short Signatures Without Random Oracles*. In Christian Cachin and Jan Camenisch, editors. Proceedings of Eurocrypt 2004, LNCS, Springer-Verlag, 2004.

[9] D. Boneh, X. Boyen. *Efficient Selective-ID Secure Identity Based Encryption Without Random Oracles*. In Proceedings of Eurocrypt 2004.

[10] 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.

[11] 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.

[12] 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.

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

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

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

[16] 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.

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

[18] 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.

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

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

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

[22] 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.

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

[24] 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.

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

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

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

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

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

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

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

[32] 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.

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

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

[35] 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.

[36] 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.

[37] 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.

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

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

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

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

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

[43] 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.

[44] W. Susilo, Y. Mu. *Non-Interactive Deniable Ring Authentication*, ICISC 2003, LNCS, Springer-Verlag, pp. 397 - 412 (preproceedings), 2003.

[45] 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.