An ID-based Authenticated Key Exchange Protocol based on Bilinear Diffie-Hellman Problem

Hai Huang and Zhenfu Cao

Department of Computer Science and Engineering, Shanghai Jiaotong University, 800 Dongchuan Road, Shanghai, 200240, People's Republic of China

Abstract

In recent years, a great deal of ID-based authenticated key exchange protocols have been proposed. However, many of them have been broken or have no security proof. The main issue is that without static private key it is difficult for simulator to fully support the SessionKeyReveal and EphemeralKeyReveal queries. Some proposals which have purported to be provably secure just hold in relatively weak model, which does not fully support above-mentioned two queries. For protocols to be proven secure in more desirable model, people must make use of the stronger gap [15] assumption, which means that the computational problem remains hard even in the presence of an effective decision oracle. However, the gap assumption may not be acceptable at all, since the decision oracle, which the proofs rely on, may not exist in real world.

Cash, Kiltz and Shoup [14] recently proposed a new computational problem called twin Diffie-Hellman problem, a nice feature of which not enjoyed by ordinary Diffie-Hellman problem is that the twin Diffie-Hellman problem remains hard, even with access to a decision oracle that recognizes solutions to the problem. At the heart of their method is the "trapdoor test" that allows us to implement an effective decision oracle for the twin Diffie-Hellman problem, without knowing the corresponding discrete logarithm.

In this paper,we present a new ID-based authenticated key exchange (ID-AKE) protocol based on the trapdoor test technique. Compared with previous ID-AKE protocols, our proposal is based on the Bilinear Diffie-Hellman (BDH) assumption, which is more standard than Gap Bilinear Diffie-Hellman (GBDH) assumption, on which previous protocols are based.

Moreover, our scheme is shown to be secure in the enhanced Canetti-Krawczyk (eCK) model, which is the currently strongest AKE security model.

Key words: ID-based, Authenticated key exchange, BDH problem, Provably secure, Trapdoor test

1 Introduction

Authenticated key exchange (AKE) is a traditional primitive of cryptography. It enables two parties, Alice (A) and Bob (B), to establish a shared session key via unsecured channels. Later, the shared session key can be used to efficiently ensure data confidentiality and integrity between A and B using efficient symmetric encryptions and message authentication codes.

In ID-based cryptography, a trusted key generator center (PKG) generates user's private key when given an identity. In ID-based authenticated key exchange, users use the ID-based public/private pairs to perform key exchange protocols instead of PKI public/private pairs.

It is desirable for ID-based authenticated key exchange protocols to possess the following attributes:

1.*Known-key security*: Each run of the protocol should result in a unique secret session key. It is reasonable to assume the adversary has the ability to learn the session keys except for one under attack. A protocol is said to be known-key secure if the compromise of one session key should not compromise other session keys.

2.*Forward security*: If the static private key of an entity is compromised, the adversary can arbitrarily masquerade as that entity in future. However, we want to guarantee that when the static private key is compromised, the adversary can not obtain the session keys that were accepted before the compromise. Protocols are said to provide perfect forward security if the static private keys of all parties involved have been compromised without compromising the previously established session keys by these entities. There is further notion of forward security in ID-based setting, which we call PKG-forward security (PKG-fs). The PKG means that even the compromise of PKG master private key does not compromise the preciously established session key. However, if the adversary is actively involved with the choice of the DH values X, Y at a session, no two-message AKE protocol can achieve forward security, according to the result of HMQV [6]. So we define weak from of forward security (wFS).

3.*Key compromise impersonation resistance*: When the static private key of an entity, say A, is compromised, the adversary can arbitrarily masquerade as A in future. However, we want to guarantee that in this case the adversary cannot masquerade as another entity, say B, to communicate with A.

4.*Ephemeral key reveal resistance*: The adversary can obtain the ephemeral

Email addresses: `chinesechess@sjtu.edu.cn` (Hai Huang), `zfcao@cs.sjtu.edu.cn` (Zhenfu Cao).

key of entities. Protocols are said to be ephemeral key reveal resistance if even when the adversary obtains the ephemeral key of entities the session key under attack still remains secure.

Protocols for AKE have been established to be surprisingly difficult to design. Bellare and Rogaway [1] firstly proposed a formal security model for authentication and key distribution. Since then, there have been several extensions to the model [2–4]. Among them, the Canetti-Krawczyk (CK) model [4] is regarded as possibly promising one. Choo, Boyd and Hitchcock [9] compared the most commonly used security models for key exchange. All these models attempt to cover these desirable properties listed above as much as possible. Recently, NAXOS [10] and CMQV [11] present a new security model named eCK which is currently strongest one. The desirable properties of eCK model include resistance to key-compromise impersonation (KCI), weak perfect forward security (wPFS) and resilience to the leakage of ephemeral private keys etc. In this paper, the eCK model is actually an adaption of eCK model from PKI-based setting to ID-based setting.

1.1 Related Work

In recent years, a great deal of ID-based authenticated key exchange protocols have been proposed. Some of them have been shown to be insecure or have no security proof, other are only proven secure in weak model, say, it does not fully support both the adversary's SessionKeyReveal and EphemeralKeyReveal queries [7–9,16].

Kudla and Paterson in [13] propose a modular proof approach, which makes use of gap assumption to keep the consistency of random oracle queries. While the approach is elegant and suitable for the security analysis of many key exchange protocols, the gap assumption may not acceptable at all, since there may not exist any polynomial time algorithms to construct such a decision oracle in real world.

Chen, Cheng and Smart in [12] propose a new approach to solve the reveal queries issue. Their approach incorporates a built-in decision function in key exchange protocols. The built-in decision function is designed to distinguish a Diffie-Hellman (DH) triple from a random element in group G. It is well known that in groups equipped with pairings such decision problem is available. So their approach does not make use of any oracle which may not exist in real world. However, although their modified Bellare and Rogaway (mBR) model fully support SessionKeyReveal queries, it does not deal with the EphemeralKeyReveal queries.

Chow and Choo in [17] propose a new ID-based authenticated key exchange protocol based on their challenge-response signature technique. They claim that their protocol allows SessionKeyReveal queries in all cases, and EphemeralKeyReveal queries in most cases, without employing any gap assumption. While this is certainly a contribution, as the simulator has no peer's static private key, their protocol cannot deal with the adversary's EphemeralKeyReveal queries to those sessions owned by the peer of Test session. In fact, this is a main issue of authenticated key agreement protocol. In this paper, we propose a better solution to this issue.

1.2 Our contributions

So far all ID-based authenticated key exchange protocols either base their security on GBDH assumption, which is a basic technique to deal with SessionKeyReveal and EphemeralKeyReveal queries or are proven secure in restricted security model, which does not fully support above-mentioned two queries.

In this paper, we present a new ID-based AKE protocol. Compared with previous ID-AKE protocols, our proposal is based on the BDH assumption, which is more standard than GBDH assumption, on which previous protocols are based.

Moreover, Our scheme is proved to be secure in the currently strongest eCK model, which is actually an adaption of eCK model from PKI-based setting to ID-based setting.

1.3 Organization

The paper is organized as follows. In section 2, we will review the related building techniques, for example, hardness assumption. In section 3 we review the security model eCK. Then we propose our scheme in section 4. In section 5, we will give the security proof of the new scheme in eCK model. In section 6 we compare the efficiency between previous ones and ours. Finally, concluding remarks are made in section 7.

2 Preliminaries

Let the value k be the security parameter. Let G be two cyclic groups of prime order q and $P \in G$ be the generator of group G. Define

$$CDH(X,Y) := Z, \text{ where } X = xP, Y = yP \text{ and } Z = xyP.$$

CDH Assumption. For any probabilistic polynomial time algorithm A,

$$Pr[A(q, G, P, X = xP, Y = yP) = CDH(X, Y)] \leq \epsilon(k).$$

where $x, y \in \mathbb{Z}_q$ and $\epsilon(k)$ is negligible. The probability is taken over the coin tosses of A, the choice of q, P and the random choices of x, y in $\mathbb{Z}_q$.

Let $e : G \times G \longrightarrow G_T$ be a bilinear pairing, where G, G_T be two cyclic groups of prime order q and $P \in G$ be the generator of group G. Define

$BDH(X, Y, W) := Z$, where $X = xP, Y = yP, W = wP$ and $Z = e(P, P)^{wxy}$.

BDH Assumption. For any probabilistic polynomial time algorithm A,

$$Pr[A(q, G, G_T, P, X = xP, Y = yP, W = wP) = BDH(X, Y, W)] \leq \epsilon(k).$$

where $x, y, z \in \mathbb{Z}_q$, and where $\epsilon(k)$ is negligible. The probability is taken over the coin tosses of A, the choice of q, P and the random choices of x, y and w in $\mathbb{Z}_q$.

The theorem below is a variant of trapdoor test theorem [14] in ID-based setting. As stated by authors of that paper, it is easy to check that both proofs are similar, so we omitted the details. The readers are referred to [14].

Theorem 1 (Trapdoor Test [14]) *Let $e : G \times G \longrightarrow G_T$ be a bilinear pairing, where G, G_T be two cyclic groups of prime order q and $P \in G$ be the generator of group G. Suppose W_1, r, s are mutually independent random variables where W_1 takes values in G, and each of r, s is uniformly distributed over $\mathbb{Z}_q$, and define the random variable $W_2 := sP - rW_1$. Further, suppose that $\hat{X}, \hat{Y}$ are random variables taking values in G and $\hat{Z}_1, \hat{Z}_2$ are random variables taking values in G_T, each of which is defined as some function of W_1 and W_2. Then we have:*

(i) W_2 is uniformly distributed over G;

(ii) W_1 and W_2 are independent;

(iii) if $W_1 = w_1P$ and $W_2 = w_2P$, then the probability that the truth value of

$$\hat{Z_1}^r \cdot \hat{Z_2} \stackrel{?}{=} e(\hat{X}, \hat{Y})^s \tag{1}$$

does not agree with the truth value of

$$\hat{Z}_1 \stackrel{?}{=} e(\hat{X}, \hat{Y})^{w_1} \bigwedge \hat{Z}_2 \stackrel{?}{=} e(\hat{X}, \hat{Y})^{w_2} \tag{2}$$

is at most $1/q$; moreover, if (2) holds, then (1) certainly holds.

Intuitionally, theorem 1 means that the simulator can use (1) to judge whether (2) holds (Knowing either the discrete logarithm $\hat{x}$ of $\hat{X}$ or the discrete logarithm $\hat{y}$ of $\hat{Y}$, the adversary can compute $\hat{Z}_1, \hat{Z}_2$ itself, while the simulator cannot). This technique is essential to implement the effective decision oracle without knowing the corresponding discrete logarithms w_1, w_2 of W_1, W_2.

3 Security Model

Our basic security model is the eCK model in ID-based setting. Further details of the original eCK model can be found in [10,11].

Participants. We model the protocol participants as a finite set P of fixed size with each ID_i being a probabilistic polynomial time (PPT) Turing machine. Each protocol participant $ID_i \in P$ may execute a polynomial number of protocol instances in parallel. We will refer to s-th instance of principal ID_i communicating with peer ID_j as $\Pi^s_{i,j}(i, j \in N)$ (*a session*).

Adversary Model. The adversary M is modeled as a PPT Turing machine and has full control of the communication network and may eavesdrop, delay, replay, alter and insert messages at will. We model the adversary's capability by providing it with oracle queries.

- **EphemeralKeyReveal($\Pi^s_{i,j}$)** The adversary obtains the ephemeral private key of $\Pi^s_{i,j}$. These queries are motivated by practical scenarios, such as if session-specific secret information is stored in insecure memory on device or if the random number generator of the party is corrupted.
- **SessionKeyReveal($\Pi^s_{i,j}$)** The adversary obtains the session key for a session s of ID_i, provided that the session holds a session key.
- **StaticKeyReveal(ID_i)** The adversary obtains the static private key of ID_i.
- **PKGStaticKeyReveal** The adversary obtains the PKG master private key, the query is used to model the PKG forward security (PKG-fs).
- **EstablishParty(ID_i)** The query models that the adversary can arbitrarily register a legal user on behalf of the party ID_i. In this way the adversary gets the party ID_i's static private key and totally controls the party ID_i. Parties against whom the adversary does not issue this query are called honest.
- **Send($\Pi^s_{i,j}, m$)** The adversary sends the message m to the session s executed by ID_i communicating with ID_j and get a response according to the protocol specification.
- **Test($\Pi^s_{i,j}$)** Only one query of this form is allowed for the adversary. Provided that session key is defined, the adversary M can execute this query at any time. Then with probability 1/2 the session key and with probability 1/2 a

uniformly chosen random value $\zeta \in \{0,1\}^k$ is returned.

Definition 1 (Matching Session) *Let $\Pi^s_{i,j}$ be a completed session with public output (ID_i, X, Y, ID_j), where ID_i is the owner of the session, ID_j is the peer, and X is ID_i' outgoing message, Y is ID_j' outgoing message. The session $\Pi^t_{j,i}$ is called the* matching session *of $\Pi^s_{i,j}$, if $\Pi^t_{j,i}$ is completed and its public output is (ID_j, Y, X, ID_i).*

Definition 2 (Freshness) *Let instance $\Pi^s_{i,j}$ be a completed session, which was executed by an honest party ID_i with another honest party ID_j. We define $\Pi^s_{i,j}$ to be* fresh *if none of the following three conditions hold:*

- *The adversary M reveals the session key of $\Pi^s_{i,j}$ or of its matching session (if latter exists).*
- *ID_j is engaged in session $\Pi^t_{j,i}$ matching to $\Pi^s_{i,j}$ and M either reveal:*
  *-both **StaticKey** of ID_i and **EphemeralKey** of $\Pi^s_{i,j}$; or*
  *-both **StaticKey** of ID_j and **EphemeralKey** of $\Pi^t_{j,i}$.*
- *No sessions matching to $\Pi^s_{i,j}$ exist and M either reveal:*
  *-both **StaticKey** of ID_i and **EphemeralKey** of $\Pi^s_{i,j}$; or*
  *-**StaticKey** of ID_j.*

Note that the adversary can reveal static key either by StaticKeyReveal queries or by PKGStaticKeyReveal query.

Definition 3 (AKE Security) *. As a function of the security parameter k, we define the advantage $Adv^{AKE}_{M,\Sigma}(k)$ of the PPT adversary M in attacking protocol Σ as*

$$Adv^{AKE}_{M,\Sigma}(k) \overset{def}{=} |Succ^{AKE}_{M,\Sigma}(k) - \tfrac{1}{2}|$$

*Here $Succ^{AKE}_{M,\Sigma}$ is the probability that the adversary queries **Test** oracle to a* fresh *instance $\Pi^s_{i,j}$, outputs a bit $\hat{b}$ such that $\hat{b} = b$, where the bit b is used by the **Test** oracle.*

We call the authenticated key exchange protocol Σ to be AKE secure if for any PPT adversary M the function is negligible.

4 An ID-based Authenticated Key Exchange Protocol based on Bilinear Diffie-Hellman Problem

Setup

Let the value k be the security parameter. Let $e: G \times G \longrightarrow G_T$ be a bilinear pairing, where G, G_T be two cyclic groups of prime order q and $P \in G$ be the generator of group G. We denote by G^* the non-identity elements set of G. Let $H_1, H_2 : \{0,1\}^* \rightarrow G^*$ and $H : \{0,1\}^* \rightarrow \{0,1\}^k$ be three hash functions. We randomly pick a value $z \in \mathbb{Z}_q$ and set $Z = zP$. We keep z as PKG master private key and publish params=$< q, G, G_T, e, k, P, Z, H_1, H_2, H >$.

Extract

For the given string $ID \in \{0,1\}^*$, PKG computes $Q_{ID_1} = H_1(ID), Q_{ID_2} = H_2(ID)$ and returns the corresponding private keys $d_{ID_1} = zQ_{ID_1}, d_{ID_2} = zQ_{ID_2}$ to the applicant, where z is the PKG master private key.

$$\begin{array}{lcl}
A & & B \\
x \leftarrow_R \mathbb{Z}_q & & y \leftarrow_R \mathbb{Z}_q \\
 & \xrightarrow{X = xP} & \\
 & \xleftarrow{Y = yP} & \\
sid = (X, Y, A, B) & & sid = (X, Y, A, B) \\
Z_1 = e(Y + Q_{B_1}, xZ + d_{A_1}) & & Z_1 = e(X + Q_{A_1}, yZ + d_{B_1}) \\
Z_2 = e(Y + Q_{B_2}, xZ + d_{A_2}) & & Z_2 = e(X + Q_{A_2}, yZ + d_{B_2}) \\
Z_3 = xY & & Z_3 = yX \\
SK = H(Z_1, Z_2, Z_3, sid) & & SK = H(Z_1, Z_2, Z_3, sid)
\end{array}$$

Fig. 1. Our proposed protocol

Protocol description

In the protocol below, A, B are two participants.

(1) A chooses an ephemeral private key $x \in \mathbb{Z}_q$ at random, computes ephemeral public key $X = xP$ and send X to B. Similarly, B randomly chooses $y \in \mathbb{Z}_q$, and send $Y = yP$ to A.

(2) Upon receiving X, party B verifies that $X \in G^*$. If so, B computes $Z_1 = e(X + Q_{A_1}, yZ + d_{B_1})$, $Z_2 = e(X + Q_{A_2}, yZ + d_{B_2})$, $Z_3 = yX$ and $SK = H(Z_1, Z_2, Z_3, sid)$, where $sid = (X, Y, A, B)$. B keeps SK as the established session key.

(3) Similarly, upon receiving Y, A checks if $Y \in G^*$. If so, A computes $Z_1 = e(Y + Q_{B_1}, xZ + d_{A_1})$, $Z_2 = e(Y + Q_{B_2}, xZ + d_{A_2})$, $Z_3 = xY$ and $SK = H(Z_1, Z_2, Z_3, sid)$. where $sid = (X, Y, A, B)$. A keeps SK as the established session key.

5 Security Proof

Theorem 2 *Suppose that the BDH assumption for (G, G_T, e, P) holds, CDH assumption for G, p holds and H_1, H_2, H are random oracles, then the proposed scheme is a secure ID-based authenticated key exchange protocol in eCK model.*

Proof. Let k denote the security parameter. Assume that the adversary M activates at most $n(k)$ honest parties and $s(k)$ sessions in each party. Assume that the adversary succeeds with non-negligible probability in the environment described in Section 3. Since $H(\cdot)$ is modeled as a random oracle, after the adversary queries Test oracle, it has only two possible ways to distinguish a session key from a random string.

CASE 1 Forging attack: At some point in its run, the adversary M queries H on the value $(Z_1, Z_2, Z_3, X, Y, A, B)$ in the Test session owned by A communicating with B. Clearly, in this case M computes the value Z_1, Z_2, Z_3 itself.

CASE 2 Key-replication attack: The adversary M forces a non-matching session to have the same session key with the Test session. In this case, the adversary M can simply learn the session key by querying the non-matching session.

The input to the key derivation function $H(\cdot)$ includes all information contained in *sid*. Since two non-matching sessions can not have same identities and same ephemeral public keys and H is modeled as random oracle, the success probability of key replication attack is negligible.

The rest of this section is mainly devoted to the analysis of the CASE 1 Forging attack. In this case, according to freshness definition, We consider separately two complementary subcases below:

CASE 1.1: No honest party owns a matching session to the Test session.

CASE 1.2: The Test session has a matching session owned by another honest party.

5.1 The analysis of CASE 1.1

Consider the following two subcase:

CASE 1.1.1: At some point, the static private key owned by the party A has been revealed by the adversary M (Note that in this case, according to the freshness definition, M is not permitted to reveal ephemeral private key of the Test session).

CASE 1.1.2: The static private key owned by the party A has never been revealed by the adversary M (Note that in this case, according to the freshness definition, M may reveal party A's ephemeral private key in the Test session).

CASE 1.1.1:

In this case, following the standard approach, we will show how to construct BDH problem solver S that uses an adversary M who succeeds with non-negligible probability in CASE 1.1.1. The solver S is given BDH problem instance($U = uP, Z = zP, W = wP$), where $u, z, w \in \mathbb{Z}_q$ and $U, Z, W \in G$. Its task is to compute $BDH(U, Z, W) = e(P, P)^{uzw}$. S sets PKG master public key to be Z. With probability at least $\frac{1}{n(k)^2}$, S guesses the adversary M will select one party denoted by A as the owner of the session $\hat{s}$ and the other party denoted by B as the peer. With probability at least $\frac{1}{s(k)}$, S guesses the adversary M will select the session $\hat{s}$ as Test session. Furthermore, S randomly chooses $s, r \in \mathbb{Z}_q$, assigns static public key $Q_{B_1} = W_1 = W, Q_{B_2} = W_2 = sP - rW$ for B, and random static public/private key pairs for the remaining $n(k) - 1$ parties (including A). When the adversary M activates a party whose static key S possesses, S follows the protocol description. We next discuss mainly the simulation action of S when the adversary M makes queries to party B (because S does not know B's static private key). Without loss of generality, we assume that B is the responder.

-$H_1(ID_i)$: S maintain an initially empty list H_1^{list} with entries of the form (ID_i, l_{i1}, Q_{i1}). The simulator S responds to these queries in the following way:

- If ID_i is already there, then S responds with stored value Q_{i1}.
- Otherwise,
  -If $ID_i = B$, S randomly chooses $s, r \in \mathbb{Z}_q$, computes $Q_{i1} = W, Q_{i2} = sP - rW$, then inserts $(ID_i, null, Q_{i1})$ into the H_1^{list} and inserts corresponding $(ID_i, null, Q_{i2})$ into the H_2^{list} (maintained in H_2 query).
  -Otherwise, S randomly chooses $l_{i1}, l_{i2} \in \mathbb{Z}_q$, computes $Q_{i1} = l_{i1}P, Q_{i2} = l_{i2}P$, inserts (ID_i, l_{i1}, Q_{i1}) into the H_1^{list} and inserts corresponding (ID_i, l_{i2}, Q_{i2}) into the H_2^{list} (maintained in H_2 query).

-$H_2(ID_i)$: S maintain an initially empty list H_2^{list} with entries of the form (ID_i, l_{i2}, Q_{i2}). The simulator S responds to these queries in the following way:

- If ID_i is already there, then S responds with stored value Q_{i2}.
- Otherwise,
  -If $ID_i = B$, S randomly chooses $s, r \in \mathbb{Z}_q$, computes $Q_{i1} = W, Q_{i2} = sP - rW$, then inserts $(ID_i, null, Q_{i2})$ into the H_2^{list} and inserts corresponding $(ID_i, null, Q_{i1})$ into the H_1^{list} (maintained in H_1 query).
  -Otherwise, S randomly chooses $l_{i1}, l_{i2} \in \mathbb{Z}_q$, computes $Q_{i1} = l_{i1}P, Q_{i2} = l_{i2}P$, inserts (ID_i, l_{i2}, Q_{i2}) into the H_2^{list} and inserts corresponding (ID_i, l_{i1}, Q_{i1}) into the H_1^{list} (maintained in H_1 query).

-$H(\hat{Z}_1, \hat{Z}_2, \hat{Z}_3, X, Y, ID_i, ID_j)$: S maintains an initially empty list H^{list} with entries of the form $(\hat{Z}_1, \hat{Z}_2, \hat{Z}_3, X, Y, ID_i, ID_j, h)$. S simulates the oracle in usual way except for queries of the form $(\hat{Z}_1, \hat{Z}_2, \hat{Z}_3, X, Y, C, B)$, where C is B' peer and may not be honest. The simulator S responds to these queries in the following way:

- If $(\hat{Z}_1, \hat{Z}_2, \hat{Z}_3, X, Y, C, B)$ is already there, then S responds with stored value h.
- Otherwise, S looks in L^{list} (maintained in the Send query) for the entry (X, Y, C, B). If finds it, S computes

$$\bar{Z}_1 = \hat{Z}_1/(e(Y, d_{C_1})e(X, yZ)e(Q_{B_1}, d_{C_1})) \tag{3}$$
$$\bar{Z}_2 = \hat{Z}_2/(e(Y, d_{C_2})e(X, yZ)e(Q_{B_2}, d_{C_2})) \tag{4}$$

  Then S checks if $\hat{Z}_1, \hat{Z}_2$ are correctly generated by computing(Theorem 1)

$$\bar{Z}_1^{\,r}\bar{Z}_2 \overset{?}{=} e(X, Z)^s \tag{5}$$

  Note that the values $\hat{Z}_1, \hat{Z}_2$ are correctly generated iff $\hat{Z}_i = e(Y + Q_{B_i}, xZ + d_{C_i})$, which is equivalent to $\bar{Z}_i = e(Q_{B_i}, xZ) = e(X, Z)^{w_i}$(i=1,2). S also computes

$$\hat{Z}_3 \overset{?}{=} yX \tag{6}$$

  -If both predicates evaluate to 1, S returns from L^{list} the stored value SK to the adversary M and stores the new tuple $(\hat{Z}_1, \hat{Z}_2, \hat{Z}_3, X, Y, C, B, SK)$ in H^{list}.

  -Otherwise, S chooses $h \in \{0,1\}^k$ at random, sends it to the adversary M and stores the new tuple $(\hat{Z}_1, \hat{Z}_2, \hat{Z}_3, X, Y, C, B, h)$ in H^{list}.
- Otherwise, S chooses $h \in \{0,1\}^k$ at random, sends it to M and stores the new tuple $(\hat{Z}_1, \hat{Z}_2, \hat{Z}_3, X, Y, C, B, h)$ in H^{list}.

-**EstablishParty(ID_i)**: The simulator S registers the ID_i on behalf of the adversary M. Concretely, S makes queries to H_1, H_2 oracle with ID_i and returns the $d_{Q_{ID_{i1}}} = l_{i1}Z, d_{Q_{ID_{i1}}} = l_{i2}Z$ to the adversary M.

-**PKGStaticKeyReveal**: The simulator S fails.

-**StaticKeyReveal(ID_i)**:

- If $ID_i = B$ then simulator fails (S do not know the corresponding static private key d_{B_1}, d_{B_2}).
- Otherwise, S returns the corresponding static private key to the adversary M.

-**EphemeralKeyReveal($\Pi^s_{i,j}$)**:

- If $ID_i = A$ and $\Pi^s_{i,j}$ is Test session, the simulator fails (The ephemeral key of Test session cannot be revealed).

- Otherwise, the simulator returns the stored ephemeral private key to the adversary M.

-Send$(\Pi^s_{i,j}, m)$: S maintains an initially empty list L^{list} with entries of the form (X, Y, ID_i, ID_j, SK).

- If $ID_i = A$ and $\Pi^s_{i,j}$ is Test session, then simulator returns U to the adversary M (We set the ephemeral public key of Test session owned by A to be U).
- If $ID_i = B$ (For convenience, we set $ID_j = C$ and $m = X$).
  -S chooses $y \leftarrow_R \mathbb{Z}_q$ and returns $Y = yP$ to the adversary M.
  -S looks in H^{list} for entry $(\hat{Z}_1, \hat{Z}_2, \hat{Z}_3, X, Y, C, B, h)$. If finds it, S computes

$$\bar{Z}_1 = \hat{Z}_1/(e(Y, d_{C_1})e(X, yZ)e(Q_{B_1}, d_{C_1})) \tag{7}$$

$$\bar{Z}_2 = \hat{Z}_2/(e(Y, d_{C_2})e(X, yZ)e(Q_{B_2}, d_{C_2})) \tag{8}$$

Then S checks if $\hat{Z}_1, \hat{Z}_2$ are correctly generated by computing (Theorem 1)

$$\bar{Z_1}^r \bar{Z_2} \overset{?}{=} e(X, Z)^s \tag{9}$$

Also, note that the values $\hat{Z}_1, \hat{Z}_2$ are correctly generated iff $\hat{Z}_i = e(Y + Q_{B_i}, xZ + d_{C_i})$, which is equivalent to $\bar{Z}_i = e(Q_{B_i}, xZ) = e(X, Z)^{w_i}$(i=1,2). S also computes

$$\hat{Z}_3 \overset{?}{=} yX \tag{10}$$

· If both predicates evaluates to 1, S stores the new tuple (X, Y, ID_i, ID_j, h) in L^{list} (The value h is from H^{list}).

·Otherwise, S chooses $SK \in \{0,1\}^k$ at random and stores the new tuple (X, Y, C, B, SK) in L^{list}.

-Otherwise, chooses $SK \in \{0,1\}^k$ at random and stores the new tuple (X, Y, C, B, SK) in L^{list}.

-SessionKeyReveal$(\Pi^s_{i,j})$:

- If $\Pi^s_{i,j}$ is Test session, S aborts.
- Otherwise, S returns the stored value SK in L^{list} to M.

-Test$(\Pi^s_{i,j})$:

- If $\Pi^s_{i,j}$ is not Test session, S aborts.
- Otherwise, S randomly chooses $\zeta \in \{0,1\}^k$ and returns it to the adversary M.

As the attack that adversary M mounts is Forging attack, if M succeeds, it must have queried oracle H on the first two inputs of this form $Z_1^* = e(Y + Q_{B_1}, xZ + d_{A_1}) = e(Y + Q_{B_1}, uZ + d_{A_1})$, $Z_2^* = e(Y + Q_{B_2}, xZ + d_{A_2}) = e(Y + Q_{B_2}, uZ + d_{A_2})$. To solve $BDH(U, Z, W)$ problem, for all entries in H^{list}, S randomly chooses one entry Z_1^*, Z_2^* and proceeds with following steps:

S computes

$$Z_1 = Z_1^*/(e(Y, d_{A_1})e(Q_{B_1}, d_{A_1})) = e(Y + Q_{B_1}, xZ) \tag{11}$$

and

$$Z_2 = Z_2^*/(e(Y, d_{A_2})e(Q_{B_1}, d_{A_2})) = e(Y + Q_{B_2}, xZ) \tag{12}$$

S divide (12) by (11),

$$\bar{Z} = \frac{Z_2}{Z_1} = e(Q_{B_2} - Q_{B_1}, xZ) = e(sP - rW_1 - W_1, xZ) \tag{13}$$

Again, S computes

$$Z = (\bar{Z}/e(X, Z)^s)^{\frac{-1}{r+1}} = e(X, Z)^{w_1} = e(U, Z)^w \tag{14}$$

This contradicts the BDH assumption.

The success probability of S is

$$Pr[S] \geq \frac{1}{s(k)n(k)^2t(k)}p_1(k) \tag{15}$$

where $p_1(k)$ is the probability of the event that CASE 1.1.1 occurs and the adversary M succeeds in this case, $t(k)$ is the polynomial bound on the number of distinct H calls made by the adversary M.

CASE 1.1.2:

In this case, given BDH problem instance ($U = uP, Z = zP, W = wP$), where $u, z, w \in \mathbb{Z}_q$ and $U, Z, W \in G$. The solver S's task is to compute $BDH(U, Z, W) = e(P, P)^{uzw}$. S sets PKG master public key to be Z. With probability at least $\frac{1}{n(k)^2}$, S guesses the adversary M will select one party denoted by A as the owner of the session $\hat{s}$ and the other party denoted by B as the peer. With probability at least $\frac{1}{s(k)}$, S guesses the adversary M will select the session $\hat{s}$ as Test session. The simulation performed by S is the same as that of CASE 1.1.1. except that S assigns $\hat{A}'$s static public key to be $Q_{A_1} = U_1 = U, Q_{A_2} = U_2 = s^*P - r^*U$, $\hat{B}'$s static public key to be $Q_{B_1} = W_1 = W, Q_{B_2} = W_2 = sP - rW$ and the ephemeral public key of the Test session owned by A to be $X = g^x$, where $x, s^*, r^*, s, r \in_R \mathbb{Z}_q$. Furthermore, S assigns random static key pairs for the remaining $n(k) - 2$ parties (except for A and B).

When the adversary M activates a party whose static key S possesses, S follows the protocol description. We next discuss the action of S when the adversary M makes queries related to party A and B (because S knows neither A's static private key nor B's static private key).

Without loss of generality, we assume that B is the responder. We assume that the message X is generated by the adversary M and Y by the simulator S. We claim below that the probability that the adversary M generates correctly these values $\hat{Z}_1 = e(X+Q_{A_1}, yZ+d_{B_1})$, $\hat{Z}_2 = e(X+Q_{A_2}, yZ+d_{B_2})$ is negligible. We show how to construct a BDH problem solver D if the adversary M queries these values correctly.

From

$$\hat{Z}_1 = e(X + Q_{A_1}, yZ + d_{B_1}) \tag{16}$$

and

$$\hat{Z}_2 = e(X + Q_{A_2}, yZ + d_{B_2}) \tag{17}$$

D computes

$$\bar{Z}_1 = \frac{\hat{Z}_1}{e(X + Q_{A_1}, yZ)} = e(X + Q_{A_1}, d_{B_1}) = e(X + U, zW) \tag{18}$$

and

$$\begin{aligned}
\bar{Z}_2 &= \left(\frac{\hat{Z}_2}{e(X + Q_{A_2}, yZ)e(X + s^*P - r^*U, sZ)e(s^*Z, -rW)}\right)^{\frac{-1}{r}} \\
&= \left(\frac{e(X + Q_{A_2}, d_{B_2})}{e(X + s^*P - r^*U, sZ)e(s^*Z, -rW)}\right)^{\frac{-1}{r}} \\
&= \left(\frac{e(X + s^*P - r^*U, -rzW)}{e(s^*Z, -rW)}\right)^{\frac{-1}{r}} \\
&= e(X - r^*U, -rzW)^{\frac{-1}{r}} \\
&= e(X - r^*U, zW)
\end{aligned}$$

D divides $\bar{Z}_1$ by $\bar{Z}_2$

$$Z = \frac{\bar{Z}_1}{\bar{Z}_2} = \frac{e(X + U, zW)}{e(X - r^*U, zW)} = e(U + r^*U, zW) = e(U, zW)^{r^*+1} \tag{19}$$

At last, from (19) D gets

$$Z^{\frac{1}{r^*+1}} = e(U, zW) = e(P, P)^{uwz} \tag{20}$$

This contradicts BDH assumption.

Now, having the conclusion above, we can deal with the H queries and send queries easily.

-$H_1(ID_i)$: S maintain an initially empty list H_1^{list} with entries of the form (ID_i, l_{i1}, Q_{i1}). S simulates the oracle in the same way as that of CASE 1.1.1 except for queries of the form $H_1(A)$.

- If $ID_i = A$, S randomly chooses $s^*, r^* \in \mathbb{Z}_q$, computes $Q_{i1} = U, Q_{i2} = s^*P - r^*U$, then inserts $(ID_i, null, Q_{i1})$ into the H_1^{list} and inserts corresponding $(ID_i, null, Q_{i2})$ into the H_2^{list} (maintained in H_2 query).

The $H_2(ID_i)$ queries can be dealt with similarly.

-**H**$(\hat{Z}_1, \hat{Z}_2, \hat{Z}_3, X, Y, ID_i, ID_j)$: S maintains an initially empty list H^{list} with entries of the form $(\hat{Z}_1, \hat{Z}_2, \hat{Z}_3, X, Y, ID_i, ID_j, h)$. S simulates the oracle in the same way as that of CASE 1.1.1 except for queries of the form $(\hat{Z}_1, \hat{Z}_2, \hat{Z}_3, X, Y, A, B)$. The simulator S responds to these queries in the following way:

- If $(\hat{Z}_1, \hat{Z}_2, \hat{Z}_3, X, Y, A, B)$ is already there, then S responds with stored value h.
- Otherwise, S chooses $h \in \{0,1\}^k$ at random, sends it to M and stores the new tuple $(\hat{Z}_1, \hat{Z}_2, \hat{Z}_3, X, Y, A, B, h)$ in H^{list}.

Send$(\Pi_{i,j}^s, m)$: S maintains an initially empty list L^{list} with entries of the form (X, Y, ID_i, ID_j, SK).

- In the case that $ID_i = B$ and $ID_j = A$ (we set $m = X$. The case that $ID_i = A$ and $ID_j = B$ also can be deal with similarly.)
  -S chooses $y \leftarrow_R \mathbb{Z}_q$ and returns $Y = yP$ to the adversary M.
  -S chooses simply $SK \in \{0,1\}^k$ at random and stores the new tuple (X, Y, A, B, SK) in L^{list}.
- In the case that $ID_i = B$ and $ID_j = C$, where S knows C's private key, The simulation action of S is similar to that of case 1.1.1.

As the attack that adversary M mounts is Forging attack, if M succeeds, it must have queried oracle H on the first two inputs of this form $Z_1^* = e(Y + Q_{B_1}, xZ + d_{A_1})$, $Z_2^* = e(Y + Q_{B_2}, xZ + d_{A_2})$, where X is chosen by S and Y by the adversary. To solve $BDH(U, Z, W)$ problem, for all entries in H^{list}, S randomly chooses one entry Z_1^*, Z_2^* and proceeds with following steps:

S compute

$$Z_1 = \frac{Z_1^*}{e(Y + Q_{B_1}, xZ)} = e(Y + Q_{B_1}, d_{A_1}) = e(Y + W, zU) \tag{21}$$

and

$$\begin{aligned}
Z_2 &= \left(\frac{Z_2^*}{(e(Y + Q_{B_2}, xZ))e(Y + sP - rW, s^*Z)e(sZ, -r^*U)} \right)^{\frac{-1}{r^*}} \\
&= \left(\frac{e(Y + Q_{B_2}, d_{A_2})}{e(Y + sP - rW, s^*Z)e(sZ, -r^*U)} \right)^{\frac{-1}{r^*}} \\
&= \left(\frac{e(Y + sP - rW, z(s^*P - r^*U))}{e(Y + sP - rW, s^*Z)e(sZ, -r^*U)} \right)^{\frac{-1}{r^*}} \\
&= \left(\frac{e(Y + sP - rW, -r^*zU)}{e(sZ, -r^*U)} \right)^{\frac{-1}{r^*}} \\
&= e(Y - rW, -r^*zU)^{\frac{-1}{r^*}} \\
&= e(Y - rW, zU)
\end{aligned}$$

S divide Z_1 by Z_2,

$$\bar{Z} = \frac{Z_1}{Z_2} = \frac{e(Y + W, zU)}{e(Y - rW, zU)} = e(W + rW, zU) \tag{22}$$

Then, S computes

$$Z = \bar{Z}^{\frac{1}{r+1}} = e(W, zU) = e(P, P)^{uzw} \tag{23}$$

This contradicts the BDH assumption.

The success probability of S is

$$Pr[S] \geq \frac{1}{s(k)n(k)^2t(k)} p_2(k) \tag{24}$$

where $p_2(k)$ is the probability of the event that CASE 1.1.2 occurs and the adversary M succeeds in this case, $t(k)$ is the polynomial bound on the number of distinct H calls made by the adversary M.

5.2 The Analysis of CASE 1.2

In this case, according to the freshness definition, the adversary M has four ways to mount the attacks.

CASE 1.2.1. The adversary M makes ephemeral key query to both the Test session and the matching session of the Test session (The adversary does not reveal their corresponding static private key).

CASE 1.2.2. The adversary learns the static private key of both the owner of Test session and its peer.

CASE 1.2.3. The adversary makes queries to the static private key of the owner of Test session and its peer's ephemeral static key.

CASE 1.2.4. The adversary makes queries to the ephemeral private key of the owner of Test session and its peer's static private key.

For CASE 1.2.1, given the BDH instance U, Z, W, where $U, Z, W \in G$, the task of solver S is to solve the BDH problem. With probability at least $\frac{1}{n(k)^2}$, S guesses the adversary M will select one party denoted by A as the owner of the session $\hat{s}$ and the other party denoted by B as the peer. With probability at least $\frac{1}{s(k)}$, S guesses the adversary M will select the session $\hat{s}$ as Test session. We assume that the owner of Test session is A and owner of matching session is B. S randomly choose $s, r, s^*, r^* \in_R \mathbb{Z}_q$ and sets A's public key to be $Q_{A_1} = U, Q_{A_2} = s^*P - r^*U$, B's public key to be $Q_{B_1} = W, Q_{B_2} = sP - rW$. S assigns the static public/private pairs for the remaining $n(k) - 2$ parties. S sets the PKG master public key to be Z. The simulation of A and B is similar to that of CASE 1.1.2. If the adversary M succeeds in a Test session for which the exponents x, y is chosen by S on behalf of A,B then M must has queried H oracle with the values $Z_1^* = e(Y + Q_{B_1}, xZ + d_{A_1})$, $Z_2^* = e(Y + Q_{B_2}, xZ + d_{A_2})$. For all entries in H^{list}, S randomly chooses one entry Z_1^*, Z_2^*. From one of these values, say Z_1^*, knowing x, y, S can compute $BDH(U, Z, W) = Z_1^*/(e(Y + Q_{B_1}, xZ)e(Z, yU)) = e(Q_{B_1}, d_{A_1}) = e(W, zU) = e(P, P)^{wuz}$. This contradicts the BDH assumption.

The success probability of S in this case is

$$Pr[S] \geq \frac{2}{s(k)n(k)^2t(k)} p_3(k) \tag{25}$$

where $p_3(k)$ is the probability of the event that CASE 1.2.1 occurs and the adversary M succeeds in this case. $t(k)$ is the polynomial bound on the number of distinct H calls made by the adversary M.

For CASE 1.2.2, given the CDH instance U, V, where $U, V \in G$, we construct a solver F of $CDH(U, V)$ problem. With probability at least $\frac{2}{s(k)^2}$, F guesses that the adversary M will select one of two sessions as Test session and other as matching session. We assume that the owner of Test session is A and owner of matching session is B. F sets ephemeral public key of Test session owned by A to be U and of its matching session to be V. F sets PKG master private key itself and assign all static public/private key pairs for $n(k)$ parties. As F knows PKC master private key and all paries' static private key, the simulation of all queries is easy. If the adversary M succeeds in a Test session then M must has queried H oracle with the values $Z_3^* = uV = vU$. From the value, F can directly output $CDH(U,V) = Z_3^*$. This contradicts the CDH assumption.

The success probability of F in this case is

$$Pr[F] \geq \frac{2}{s(k)^2 t(k)} p_4(k) \tag{26}$$

where $p_4(k)$ is the probability of the event that CASE 1.2.2 occurs and the adversary M succeeds in this case. $t(k)$ is the polynomial bound on the number of distinct H calls made by the adversary M.

For CASE 1.2.3 and CASE 1.2.4, the simulation of A and B is similar to that of CASE 1.1.1. The details are omitted.

Together with (15), (24) and (25), the success probability of S is

$$Pr[S] \geq \max_{i=1,2,3,5,6}\{\tfrac{1}{s(k)n(k)^2 t(k)} p_i(k)\}$$

where $p_5(k), p_6(k)$ are defined in CASE 1.2.3 and CASE 1.2.4 respectively.

The success probability of F is

$$Pr[F] \geq \frac{2}{s(k)^2 t(k)} p_4(k) \tag{27}$$

where $p_4(k)$ are defined in (26).

If the adversary M succeeds with non-negligible probability in any case above, we can also solve the BDH or CDH problem with non-negligible probability, which contradicts the assumed security of BDH, CDH problem. In addition, note that the BDH problem can be reduced to CDH problem. So we can conclude that our scheme is based its security on BDH problem.

6 Protocol Comparison

The table 1 shows the comparison between ID-based AKE protocols in terms of efficiency, security model and underlying hardness assumptions. We do not take into account subgroup validation and off-line computation that may be applicable.

We use the following symbols to explain the computational performance of each scheme. For simplicity, we just consider expensive operations:

-P: pairing.

-E: exponentiation in G.
-T: exponentiation in G_T.

We denote by ECK the enhanced Chen-Kudla protocol [12], by EMB the enhanced McCullagh-Barreto protocol [12]. We also denote by MBDH the Modified Bilinear Diffie-Hellman, on which protocol 2 of paper [17] is based. The notation ℓ-BCAA1 means Bilinear Collision Attack Assumption, on which EMB scheme is based.

As stated in the introduction, both protocol 1 and protocol 2 from [17] does not support the adversary's EphemeralKeyReveal queries to those sessions owned by the peer of Test session, so both protocols does not achieve requirements of CK model (The CK model allows adversaries to make EphemeralKeyReveal queries to all session except for Test session and its matching session). We denote by CK^* their security model below. The mBR model does not cover EphemeralKeyReveal queries at all. Also, note that neither mBR nor CK^* models cover PKG-fs, while our eCK model covers it. If we want those protocols proven secure in two models above to be PKG-fs, extra computation must be added.

Protocol	Efficiency	Security model	Assumption
CK[13]	1P+2E	mBR	GBDH
Smart[13]	2P+2E	mBR	GBDH
SCK-1[12]	2P+3E	mBR,PKG-fs	BDH
SYL[12]	1P+3E	mBR,PKG-fs	BDH
ECK[12]	3P+3E	mBR	BDH
EMB[12]	4P+2E+1T	mBR	ℓ-BCAA1
CC07_1[17]	1P+3E	CK^*,KCI	BDH
CC07_2[17]	1P+5E	CK^*,KCI,PKG-fs	MBDH
Our scheme	2P+3E	eCK	BDH

Table 1: protocol comparison

As shown in table 1 above, compared with all previous ID-based AKE protocols, the security model of our protocol is strongest. Meanwhile, Our protocol has a security proof under BDH assumption, which is more standard than GBDH assumption.

7 Conclusion

It is well known that the ID-based authenticated key exchange is surprisingly difficult to design and prove. The main issue is that without corresponding static private key, it is difficult for the simulator to deal with SessionKeyReveal or EphemeralKeyReveal queries.

One approach is to rely security proof on GBDH assumption. Another approach is to prove security in weaker model, which does not fully support above-mentioned two queries.

In this paper, we present a new ID-based AKE protocol. As compared with previous ID-AKE protocols, our proposal is based on the BDH assumption, which is more standard than GBDH assumption, on which previous protocols are based.

Moreover, The eCK model is the currently strongest security model which better supports SessionKeyReveal or EphemeralKeyReveal queries. To the best of our knowledge, our scheme is the first ID-based AKE protocol provably secure in the eCK model.

References

[1] M.Bellare, P.Rogaway, Entity authentication and key distribution, Advances in Cryptology-CRYPTO'93, volume 773 of Lecture Notes in Computer Science, pages 232-249. Springer, 1993.

[2] Mihir Bellare, Phillip Rogaway, Provably Secure Session Key Distribution: The Three Party Case, In Proceedings of the Twenty-Seventh Annual ACM Symposium on Theory of Computing (STOC95), pages 57-66. ACM Press, 1995.

[3] Mihir Bellare, David Pointcheval, and Phillip Rogaway, Authenticated Key Exchange Secure Against Dictionary Attacks, In Advances in Cryptology-EUROCRYPT'00, volume 1807 of Lecture Notes in Computer Science, pages 139-155. Springer, May 2000.

[4] Ran Canetti and Hugo Krawczyk, Analysis of Key-Exchange Protocols and Their Use for Building Secure Channels, In Advances in Cryptology-EUROCRYPT'01, volume 2045 of Lecture Notes in Computer Science, pages 453-474. Springer, 2001.

[5] Kim-Kwang Raymond Choo, Colin Boyd, and Yvonne Hitchcock, Examining Indistinguishability-Based Proof Models for Key Establishment Protocols, In Advances in Cryptology-ASIACRYPT'05, volume 3788 of Lecture Notes in Computer Science, pages 585-604. Springer, 2005.

[6] Hugo Krawczyk,HMQV: A High-Performance Secure Diffie-Hellman Protocol. In: Shoup, V. (ed.) CRYPTO 2005. LNCS, vol. 3621, pp. 546-566. Springer, Heidelberg (2005).

[7] N.P. Smart. An identity based authenticated key agreement protocol based on the Weil pairing. Electronics Letters, 38, 630-632, 2002.

[8] L. Chen and C. Kudla. Identity based authenticated key agreement from pairings. In IEEE Computer Security Foundations Workshop, 219-233, 2003. The modified version of this paper is available at Cryptology ePrint Archive, Report 2002/184.

[9] K. Choo, C. Boyd and Y. Hitchcock. On session key construction in provably-secure key establishment protocols: revisiting Chen Kudla (2003) and McCullagh Barreto (2005) ID-based protocols. Cryptology ePrint Archive, Report 2005/206. Also available at the Proceedings of Mycrypt 2005, Springer-Verlag LNCS 3715, 116-131, 2005.

[10] B.LaMacchia,K.Lauter,A.Mityagin, Stronger Security of Authenticated Key Exchange, avaliable at http://eprint.iacr.org/2006/073.

[11] B.Ustaoglu, Obtaining a secure and efficient key agreement protocol from (H)MQV and NAXOS, available at http://eprint.iacr.org/2007/123.

[12] L.Chen, Z.Cheng and N.P. Smart, Identity-based Key Agreement Protocols From Pairings, available at http://eprint.iacr.org/2006/119.

[13] C. Kudla and K. Paterson. Modular security proofs for key agreement protocols. In Advances in Cryptology -Asiacrypt 2005, Springer-Verlag LNCS 3788, 549-565, 2005.

[14] D.Cash,E.Kiltz,V.Shoup, The Twin Diffie-Hellman Problem and Applications, Advances in Cryptlogy-EUROCRYPT 2008, Full version available at http://eprint.iacr.org/2008/067.

[15] T.Okamoto and D.Pointcheval, The Gap-Problems: A new class of problems for the security of cryptographic schems, Public Key Cryptography-PKC2001,LNCS 1992(2001),104-118.

[16] N. McCullagh and P.S.L.M. Barreto. A new two-party identity-based authenticated key agreement. In Topics in Cryptology-CT-RSA 2005, Springer-Verlag LNCS 3376, 262-274, 2005.

[17] S.S.M.Chow and K.R.Choo. Strongly-Secure Identity-Based Key Agreement and Anonymous Extension. Information Security, Volume 4779/2007,Springer Berlin Heidelberg,203-220, 2007. Cryptology ePrint Archive, Report 2007/018. Full version of this paper (2007).