On Security Proof of McCullagh-Barreto's Key Agreement Protocol and its Variants

Zhaohui Cheng[1] and Liqun Chen[2]

[1] School of Computing Science, Middlesex University
White Hart Lane, London N17 8HR, UK
m.z.cheng@mdx.ac.uk
[2] Hewlett-Packard Laboratories, Bristol, UK
liqun.chen@hp.com

Abstract. McCullagh and Barreto presented an identity-based authenticated key agreement protocol in CT-RSA 2005. Their protocol was found to be vulnerable to a key-compromise impersonation attack. In order to recover the weakness, McCullagh and Barreto, and Xie proposed two variants of the protocol respectively. In each of these works, a security proof of the proposed protocol was presented. In this paper, we revisit these three security proofs and show that all the reductions in these proofs are invalid, because the property of indistinguishability between their simulation and the real world was not held. As a replacement, we slightly modify the McCullagh and Barreto's second protocol and then formally analyse the security of the modified scheme in the Bellare-Rogaway key agreement model.

1 Introduction

An identity-based authenticated key agreement protocol is a key agreement protocol where each of two (or more) parties uses an identity-based asymmetric key pair instead of a traditional public/private key pair for authentication and determination of the established key, which at the end of the protocol is shared by these parties.

The concept of identity-based cryptography was first formulated by Shamir in 1984 [23] in which a public key is the identity (an arbitrary string) of a user, and the corresponding private key is created by binding the identity string with a master secret of a trusted authority (called key generation center). In the same paper, Shamir provided the first identity-based key construction that was based on the RSA problem, and presented an identity-based signature scheme. By using varieties of the Shamir key construction, a number of identity-based key agreement schemes were proposed (e.g., [15, 21, 28]).

In 2000, Sakai et al. introduced an identity-based key agreement scheme based on bilinear pairings over elliptic curves [26]. Their protocol made use of an interesting identity-based key construction with pairings, in which an identity string is mapped to a point on an elliptic curve and then the corresponding private key is computed by multiplying the mapped point with the master private key that is a random integer. A similar key construction is also used by Boneh and Franklin in their well-known provable identity-based encryption scheme [3]. After that, many other identity-based key agreement schemes using this key construction were presented, such as [13, 22, 24, 29]. The security of these key agreement schemes were scrutinized (although some errors in a few reductions have been pointed out recently but fixed as well, e.g., [14]).

In 2003, Sakai and Kasahara presented a new identity-based key construction using pairings (SK key construction for short) [25], which can be tracked back to the work in [20]. This key construction has the potential to improve performance, where an identity string is mapped to an element h of the cyclic group $\mathbb{Z}_q^*$ instead of a point on an elliptic curve directly. The corresponding private key is generated by first computing an inverse of the sum of the master secret (a random integer from $\mathbb{Z}_q^*$) and the mapped value h, and then multiplying a point of the elliptic curve (which is the generator of an order q subgroup of the group of points on the curve) with the inverse.

Based on the SK key construction, McCullagh and Barreto (MB) presented an identity-based authenticated key agreement protocol on CT-RSA 2005 [17], which appears to be more efficient on computation than the above mentioned schemes [13, 22, 24, 29]. However, as pointed out by Cheng [7] and Xie [30], the scheme is vulnerable to a key-compromise impersonation attack, i.e., if an adversary knows a party A's long-term private key, the adversary can impersonate any other party to A. In order to recover this security weakness, McCullagh and Barreto [18], and Xie [31] proposed two fixes respectively. Meanwhile, they provided a security reduction for each protocol in the Bellare-Rogaway's key agreement formulation [5, 6].

In this paper, we revisit the security proofs in [17, 18, 31] and show that all these three proofs are problematic. More specifically, in their security reductions, the property of indistinguishability between their simulation and the real world was not held. We observe an interesting feature when simulating an identity-based cryptographic world. In any identity-based cryptographic scheme, given a certain identity string, system parameters and key generation center's master public key, it is universally verifiable whether the private key corresponding to the identity string is correctly constructed or not. Therefore, if a simulator is not able to offer an adversary necessary evidence, which allows the adversary to verify the correction of a simulated key construction, the simulation fails, because the adversary can immediately notice the inconsistency between the simulation and the simulated real world. All the three proofs failed to provide this feature.

As pointed out in [16], Xie's scheme still suffers from the key-compromise impersonation attack, hence here we do not formally analyse it. Instead, we slightly modify McCullagh and Barreto's second protocol [18] and then present a formal reduction for the scheme in the Bellare-Rogaway key agreement model. The new reduction demonstrates the security strength of the scheme.

The paper is organized as follows. First, we recall the existing primitives, some related assumptions and the security model of a key agreement scheme in next section. Then, in Section 3 we revisit the three protocols. In our specification, we refer to these protocols as the MB protocol and its variants. We give a sketch of the three security proofs and point out the flaws in the proofs in Section 4. After that, in Section 5 we further modify McCullagh and Barreto's second protocol and present a new reduction for the modified scheme. At the end, we conclude the paper.

2 Preliminaries

Before revisiting the protocols and their proofs, we recall some related pairing primitives, assumptions and the security model of an authenticated key agreement (AK) scheme.

2.1 Bilinear Groups and Some Assumptions

Definition 1 *A pairing is a bilinear map $\hat{e} : \mathbb{G}_1 \times \mathbb{G}_1 \rightarrow \mathbb{G}_2$ with two cyclic groups $\mathbb{G}_1$ and $\mathbb{G}_2$ of prime order q, which has the following properties [3]:*

1. *Bilinear: $\hat{e}(sP, tR) = \hat{e}(P, R)^{st}$ for all $P, R \in \mathbb{G}_1$ and $s, t \in \mathbb{Z}_q$.*
2. *Non-degenerate: for a given point $Q \in \mathbb{G}_1$, $\hat{e}(Q, R) = 1_{\mathbb{G}_2}$ for all $R \in \mathbb{G}_1$ if and only if $Q = 1_{\mathbb{G}_1}$.*
3. *Computable: $\hat{e}(P, Q)$ is efficiently computable for any $P, Q \in \mathbb{G}_1$.*

Some researchers have recently worked on varieties of pairings, such as asymmetric pairings [27], where two inputs from two (possibly) different groups are mapped into an element in the third group, i.e., $\hat{e} : \mathbb{G}_1 \times \mathbb{G}_2 \rightarrow \mathbb{G}_3$. For the purpose of analyzing security of the key agreement protocols based on the SK key construction, in the remaining of this paper, we will focus on a symmetric pairing, i.e., $\hat{e} : \mathbb{G}_1 \times \mathbb{G}_1 \rightarrow \mathbb{G}_2$.

The following Bilinear Diffie-Hellman assumption has been used to construct many exciting cryptography schemes.

Assumption 1 (BDH [3]) *For $x, y, z \in_R \mathbb{Z}_q^*$, $P \in \mathbb{G}_1^*$, $\hat{e} : \mathbb{G}_1 \times \mathbb{G}_1 \rightarrow \mathbb{G}_2$, given (P, xP, yP, zP), computing $\hat{e}(P, P)^{xyz}$ is hard.*

In [17, 18, 31], the authors reduced the security of their protocols to the following Bilinear Inverse Diffie-Hellman assumption. The assumption was proved to be equivalent to the BDH assumption [32].

Assumption 2 (BIDH [32]) *For $x, y \in_R \mathbb{Z}_q^*$, $P \in \mathbb{G}_1^*$, $\hat{e} : \mathbb{G}_1 \times \mathbb{G}_1 \rightarrow \mathbb{G}_2$, given (P, xP, yP), computing $\hat{e}(P, P)^{y/x}$ is hard.*

There are a few related assumptions which have been used in the literature to construct cryptography systems (see [10] for a summary and relations among these assumptions). The following decision k-BDHI assumption was used in [1] to construct a selective identity-based encryption without random oracles, and in [10, 12] to prove the security of identity-based encryption schemes using SK key construction [25].

Assumption 3 (k-BDHI [1]) *For an integer k, and $x \in_R \mathbb{Z}_q^*$, $P \in \mathbb{G}_1^*$, $\hat{e} : \mathbb{G}_1 \times \mathbb{G}_1 \to \mathbb{G}_2$, given $(P, xP, x^2P, \ldots, x^kP)$, computing $\hat{e}(P,P)^{1/x}$ is hard.*

Assumption 4 (k-DBDHI [1]) *For an integer k, and $x, r \in_R \mathbb{Z}_q^*$, $P \in \mathbb{G}_1^*$, $\hat{e} : \mathbb{G}_1 \times \mathbb{G}_1 \to \mathbb{G}_2$, distinguishing between the distributions $(P, xP, x^2P, \ldots, x^kP, \hat{e}(P,P)^{1/x})$ and $(P, xP, x^2P, \ldots, x^kP, \hat{e}(P,P)^r)$ is hard.*

Assumption 5 (k-BCAA1 [10]) *For an integer k, and $x \in_R \mathbb{Z}_q^*$, $P \in \mathbb{G}_1^*, \hat{e} : \mathbb{G}_1 \times \mathbb{G}_1 \to \mathbb{G}_2$, given $(P, xP, h_0, (h_1, \frac{1}{h_1+x}P), \ldots, (h_k, \frac{1}{h_k+x}P))$ where $h_i \in_R \mathbb{Z}_q^*$ and different from each other for $0 \le i \le k$, computing $\hat{e}(P,P)^{1/(x+h_0)}$ is hard.*

The relationship between k-BDHI and k-BCAA1 has been proved in [10] by the following theorem.

Theorem 1 ([10]) *If there exists a polynomial time algorithm to solve (k-1)-BDHI, then there exists a polynomial time algorithm for k-BCAA1. If there exists a polynomial time algorithm to solve (k-1)-BCAA1, then there exists a polynomial time algorithm for k-BDHI.*

We shall use the gap variant of k-BCAA1 in this work. That is

Assumption 6 (k-Gap-BCAA1) *For an integer k, and $x \in_R \mathbb{Z}_q^*$, $P \in \mathbb{G}_1^*, \hat{e} : \mathbb{G}_1 \times \mathbb{G}_1 \to \mathbb{G}_2$, given $(P, xP, h_0, (h_1, \frac{1}{h_1+x}P), \ldots, (h_k, \frac{1}{h_k+x}P))$ where $h_i \in_R \mathbb{Z}_q^*$ and different from each other for $0 \le i \le k$ and the access to a decision BIDH oracle (DBIDH) which given (P, aP, bP, rP) returns 1 if $\hat{e}(P,P)^r = \hat{e}(P,P)^{b/a}$, else returns 0, computing $\hat{e}(P,P)^{1/(x+h_0)}$ is hard.*

2.2 Security Model of Key Agreement

In this paper, we use Blake-Wilson et al.'s key agreement formulation which extends the Bellare-Rogaway model [5] to public key construction to test the security strength of a protocol.

In the Bellare-Rogaway model [5], each party involved in a session is treated as an oracle. An adversary can access the oracle by issuing some specified queries. An oracle $\Pi_{i,j}^s$ denotes an instance s of a party i involved with a partner party j in a session where the instance of the party j is $\Pi_{j,i}^t$ for some t. The oracle $\Pi_{i,j}^s$ executes the prescribed protocol Π and produces the output as $\Pi(1^k, i, j, S_i, P_i, P_j, conv_{i,j}^s, r_{i,j}^s, x) = (m, \delta_{i,j}^s, \sigma_{i,j}^s)$ where x is the input message; m is the outgoing message; S_i and P_i are the private/public key pair of party i; P_j is the public key of j; $\delta_{i,j}^s$ is the decision of the oracle (accept or reject the session or no decision yet) and $\sigma_{i,j}^s$ is the generated session key (please see [5, 6] for the details). At the end of Π, the conversation transcript $conv_{i,j}^s$ is updated as $conv_{i,j}^s.x.m$ (where "$a.b$" denotes the result of the concatenation of two strings, a and b).

The security of a protocol is tested by a game with two phases. In the first phase, an adversary E is allowed to issue queries as follows in any order.

1. Send a message query: $Send(\Pi^s_{i,j}, x)$. $\Pi^s_{i,j}$ executes $\Pi(1^k, i, j, S_i, P_i, P_j, conv^s_{i,j}, r^s_{i,j}, x)$ and responds with m and $\delta^s_{i,j}$. If the oracle $\Pi^s_{i,j}$ does not exist, it will be created. Note that x can be λ in the query which causes an oracle to be generated as an initiator, otherwise as a responder.
2. Reveal a session key: $Reveal(\Pi^s_{i,j})$. $\Pi^s_{i,j}$ reveals the private output of the session $\sigma^s_{i,j}$ if the oracle accepts.
3. Corrupt a party: $Corrupt(i)$. The party i responds with the private key S_i.

Once the adversary decides that the first phase is over, it starts the second phase by choosing a *fresh oracle* $\Pi^s_{i,j}$ and issuing another query: $Test(\Pi^s_{i,j})$. Oracle $\Pi^s_{i,j}$, as a challenger, randomly chooses $b \in \{0, 1\}$ and responds with $\sigma^s_{i,j}$, if $b = 0$; otherwise it returns a random sample generated according to the distribution of the session secret $\sigma^s_{i,j}$. If the adversary guesses the correct b, we say that it wins. Define

$$Advantage^E(k) = \max\{0, \Pr[E \text{ wins}] - \tfrac{1}{2}\}.$$

The fresh oracle in the game is defined as below, which is particularly defined to address the key-compromise impersonation resilience property [14].

Definition 2 (fresh oracle) *An oracle $\Pi^s_{i,j}$ is fresh if (1) $\Pi^s_{i,j}$ has accepted; (2) $\Pi^s_{i,j}$ is unopened (not being issued the* Reveal *query); (3) j is not corrupted (not being issued the* Corrupt *query); (4) there is no opened oracle $\Pi^t_{j,i}$, which has had a matching conversation to $\Pi^s_{i,j}$.*

We stress that in this work, it is required that $i \neq j$ for the chosen fresh oracle in the game (note that the model allows a party to engage in a session with itself).

We use session ID [4], which is the concatenation of the messages in a session (the transcript of an oracle), to define matching conversations. Two oracles $\Pi^s_{i,j}$ and $\Pi^t_{j,i}$ have the matching conversations if both of them derive the same session ID from (each own) conversation transcript.

A secure authenticated key (AK) agreement protocol is defined as below.

Definition 3 [6] *Protocol Π is a secure AK if:*

1. *In the presence of the benign adversary, which faithfully conveys the messages, on $\Pi^s_{i,j}$ and $\Pi^t_{j,i}$, both oracles always accept holding the same session key σ, and this key is distributed uniformly at random on $\{0,1\}^k$;*

and if for every adversary E:

2. *If two oracles $\Pi^s_{i,j}$ and $\Pi^t_{j,i}$ have matching conversations and both i and j are uncorrupted, then both accept and hold the same session key σ;*
3. *$Advantage^E(k)$ is negligible.*

3 The MB Protocol and its Variants

In this section, we recall McCullagh and Barreto's protocol and its variants. These protocols use the same key construction (the SK key construction) and exchange the same message flows. However, in the last step of the protocols, each protocol has a different scheme to compute an established session key.

Setup. Given the security parameter k, the algorithm randomly chooses $s \in \mathbb{Z}_q^*$ and generates the system **params** $(\mathbb{G}_1, \mathbb{G}_2, \hat{e}, q, P, sP, H_1, H_2)$ where $P \in \mathbb{G}_1^*$ and $H_1 : \{0,1\}^* \to \mathbb{Z}_q^*$, $H_2 : \mathbb{G}_2 \to \{0,1\}^n$ for some integer n. The **master key** is s which is kept secret by the center.

Extract. The schemes employs the SK key construction [25]. Given an identity ID_A, the **master key** s, and the system **params**, the algorithm computes $H_1(ID_A) = \alpha \in \mathbb{Z}_q^*$ and the corresponding private key $d_A = \frac{1}{s+\alpha}P$ for ID_A. $\alpha P + sP$ will be treated as the real public key corresponding to ID_A.

Protocol. Suppose $H_1(A) = \alpha$ and $H_1(B) = \beta$. Party A and B randomly choose x and y from $\mathbb{Z}_q^*$ respectively. The protocol proceeds as follow.

$$A \to B : T_1 = x(\beta P + sP)$$
$$B \to A : T_2 = y(\alpha P + sP)$$

On completion of the protocol, there are three ways to compute the agreed secret which have different security strength (here we slightly change the protocols in [17, 18, 31] by employing an extra hash function on the agreed secret to generate the session keys).

Scheme 1 (McCullagh-Barreto's first scheme [17]). A computes $K = \hat{e}(T_2, d_A)^x = \hat{e}(P,P)^{xy}$ and B computes $K = \hat{e}(T_1, d_B)^y = \hat{e}(P,P)^{xy}$. The agreed session key is $SK = H_2(\hat{e}(P,P)^{xy})$. This scheme *appears* to provide an interesting security property: the perfect forward secrecy (i.e., if the private keys of both parties are compromised, the agreed session keys between these two parties cannot be recovered by the adversary). However, this scheme does not achieve the key-compromise impersonation resilience [7, 30]. To defeat this attack, Xie and McCullagh-Barreto attempted the following two variants respectively.

Scheme 2 (McCullagh-Barreto's second scheme [18]). A computes $K = \hat{e}(T_2, d_A) \cdot \hat{e}(P,P)^x = \hat{e}(P,P)^{x+y}$ and B computes $K = \hat{e}(T_1, d_B) \cdot \hat{e}(P,P)^y = \hat{e}(P,P)^{x+y}$. The agreed session key is $SK = H_2(\hat{e}(P,P)^{x+y})$. Although now the protocol achieves the key-compromise impersonation resilience property, this scheme looses another desirable security attribution: the perfect forward secrecy.

Scheme 3 (Xie's scheme [31]). A computes $K = \hat{e}(T_2, d_A)^{x+1} \cdot \hat{e}(P,P)^x = \hat{e}(P,P)^{xy+x+y}$ and B computes $K = \hat{e}(T_1, d_B)^{y+1} \cdot \hat{e}(P,P)^y = \hat{e}(P,P)^{xy+x+y}$. The agreed session key is $SK = H_2(\hat{e}(P,P)^{xy+x+y})$. Unfortunately this modification is still vulnerable to the key-compromise impersonation attack and further suffers from a trivial man-in-the-middle attack as pointed out in [16].

Note that in the original schemes described in [17, 18, 31], the use of H_2 is not explicitly required. It will result in a potential security problem, which will be discussed in the security proof of Section 4.2.

4 Their Security Proofs

4.1 A Sketch of Their Proofs

In this subsection, we give a sketch of three security proofs from [17, 18, 31] respectively, each for one variant of the MB protocol as described in Section 3. The proofs were intended to adopt the security model proposed by Bellare and Rogaway [5] and extended by Blake-Wilson et al. [6].

All of the three proofs are based on the bilinear inverse Diffie-Hellman (BIDH) problem, described in Assumption 2, i.e., given $(P, \alpha P, \beta P)$, computing $\hat{e}(P, P)^{\beta/\alpha}$ is computationally infeasible. Each proof involves two algorithms: an adversary $\mathcal{A}$ and a challenger (i.e., a simulator of the real world) $\mathcal{B}$. $\mathcal{A}$'s goal is to break a specified protocol, and $\mathcal{B}$'s goal is to solve the BIDH problem with the help of $\mathcal{A}$.

Each proof includes a set of parties, each modelled by an oracle. The notation $\Pi^s_{i,j}$ denotes an oracle i believing that it is participating in the s-th run of the protocol with another oracle j. $\mathcal{A}$ can access any oracle by issuing the queries of Create, Corrupt, Send, and Test. All queries by $\mathcal{A}$ pass through $\mathcal{B}$. Before the game starts, $\mathcal{B}$ randomly selects a pair of oracles, $\Pi^s_{i,j}$ and $\Pi^t_{j,i}$. $\mathcal{B}$ expects that $\mathcal{A}$ is going to attack the oracle $\Pi^s_{i,j}$ by playing the role of $\Pi^t_{j,i}$. In the three proofs, $\mathcal{A}$ and $\mathcal{B}$ play the game described in Section 2.2 in the following three slightly different ways.

Proof 1 (for Scheme 1 [17]). To answer a Create/Corrupt query for any oracle m where $m \neq j$, $\mathcal{B}$ chooses a random integer $y_m \in \mathbb{Z}^*_q$, and answers $y_m P$ as m's public key and $y_m^{-1} P$ as m's private key. $\mathcal{B}$ answers αP as j's public key and does not know j's private key $\alpha^{-1} P$. To answer a Send query for any oracle except $\Pi^s_{i,j}$, $\mathcal{B}$ follows the protocol properly. To answer a Send query for $\Pi^s_{i,j}$, $\mathcal{B}$ chooses a random integer $x_i \in \mathbb{Z}^*_q$ and answers $x_i P$. The proof relies on that an input from $\mathcal{A}$ as $\Pi^t_{j,i}$'s response is exactly the value of βP. After a Test query for $\Pi^s_{i,j}$, if $\mathcal{A}$ successfully breaks Scheme 1 by distinguishing the established key, $K = \hat{e}(P, P)^{x_i \beta / y_i \alpha}$, from a random number, $\mathcal{B}$ can get $\hat{e}(P, P)^{\beta/\alpha}$ by computing K^{y_i / x_i}.

Proof 2 (for Scheme 2 [18]). $\mathcal{B}$ answers Create/Corrupt queries in the same way as it did in Proof 1. To answer a Send query for any oracle except $\Pi^s_{i,j}$, $\mathcal{B}$ follows the protocol properly. To answer a Send query for $\Pi^s_{i,j}$, $\mathcal{B}$ answers βP. The input from $\mathcal{A}$ as $\Pi^t_{j,i}$'s response is an arbitrary value δP. After a Test query for $\Pi^s_{i,j}$, if $\mathcal{A}$ successfully breaks Scheme 2 by distinguishing the established key, $K = \hat{e}(P, P)^{\beta/\alpha + \delta/y_i}$, from a random number, $\mathcal{B}$ can get $\hat{e}(P, P)^{\beta/\alpha}$ by computing $K / \hat{e}(P, P)^{\delta/y_i}$.

Proof 3 (for Scheme 3 [31]). To answer a Create/Corrupt query for any oracle m where $m \notin \{i, j\}$, $\mathcal{B}$ chooses a random integer $y_m \in \mathbb{Z}^*_q$, and answers $y_m P$ as m's public key and $y_m^{-1} P$ as m's private key. $\mathcal{B}$ answers αP as i's public key and does not know i's private key $\alpha^{-1} P$. $\mathcal{B}$ answers βP as j's public key and does not know j's private key $\beta^{-1}P$. To answer a Send query for any oracle except $\Pi^s_{i,j}$, $\mathcal{B}$ follows the protocol properly. To answer a Send query for $\Pi^s_{i,j}$, $\mathcal{B}$ chooses a random integer $x_i \in \mathbb{Z}^*_q$ and answers $x_i\beta P$. The proof relies on that an input from $\mathcal{A}$ as $\Pi^t_{j,i}$'s response is exactly the value of βP (i.e., $y_j\alpha P = \beta P$ for some y_j). After a Test query for $\Pi^s_{i,j}$, if $\mathcal{A}$ successfully breaks Scheme 3 by distinguishing the established key, $K = \hat{e}(P,P)^{(x_i+1)\beta/\alpha} \cdot \hat{e}(P,P)^{x_i}$, from a random number, $\mathcal{B}$ can get $\hat{e}(P,P)^{\beta/\alpha}$ by computing $(K/\hat{e}(P,P)^{x_i})^{1/(x_i+1)}$.

4.2 Analysis of Their Proofs

We now show that all the three reductions described in the last subsection are invalid. More specifically, the reductions have following three problems (independently Choo et al. [9] pointed out some similar errors of [17]).

Problem 1: From $\mathcal{A}$'s point view, the simulation offered by $\mathcal{B}$ is distinguishable from the real world of an identity-based authenticated key agreement protocol.

In any identity-based cryptographic world, the correctness of a public/private key pair derived from a chosen identity string, ID, is verifiable, given system parameters and the key generation center's master public key. In those security reductions based on a standard model (e.g., [1, 2]), an adversary can use ID directly as the public key to verify the result of a private key generation query (i.e., Corrupt query). In those security reductions based on a random oracle model, such as [3, 13], to verify a correct key deriving can be done with a query of ID to the random oracle. This is acceptable in the random oracle model based reductions, although it is not as ideal as the first case.

It is addressed in [17, 18, 31] that the identity map function H_1 in the MB protocol and its variants is by means of the random oracle model. However, how to respond to the H_1 query is not specified in these three proofs. Another related missing part is that these three proofs do not specify either which entity has the access to the value of s, or what the system parameters that $\mathcal{A}$ would get access to should be. We can see that $\mathcal{B}$ is not able to answer the H_1 query by following the Create and Corrupt queries specified in these three proofs. As a result, $\mathcal{A}$ cannot verify correctness of either Create or Corrupt query result from ID and the system parameters. $\mathcal{A}$ can then immediately notice that $\mathcal{B}$ is a simulator, instead of the real world. We discuss this issue in the following two cases, dependent on whether or not $\mathcal{B}$ knows the value of s.

1. The value s is not known to $\mathcal{B}$. Following the three proofs, to answer the Create/Corrupt query to an oracle with the identity ID_m, $\mathcal{B}$ assigns a random element pair, y_mP, $y_m^{-1}P \in \mathbb{G}^*_1$, as the public/private key pair. However, $\mathcal{B}$ is not able to give the value of $u_m = H_1(ID_m)$, satisfying $u_mP = y_mP - sP$, because to solve the discrete logarithm problem in $\mathbb{G}_1$ is computational infeasible, which is implied by the used BIDH assumption. Therefore, $\mathcal{B}$ is not able to answer the oracle query $H_1(ID_m)$, and $\mathcal{A}$ then cannot verify correctness of the received y_mP and $y_m^{-1}P$ from ID_m and sP.

2. The value s is chosen by $\mathcal{B}$. Following the reductions, to answer the Create query to an oracle with the identity ID_j in Proof 1 and Proof 2 (or ID_i in Proof 3), $\mathcal{B}$ assigns $\alpha P \in \mathbb{G}_1^*$, as the public key to the party j (or i). However, again $\mathcal{B}$ is not able to give the value of u_j (or u_i), satisfying $u_j P$ (or $u_i P$) $= \alpha P - sP$, because to solve the discrete logarithm problem in $\mathbb{G}_1$ is supposed to be computational infeasible. Therefore $\mathcal{B}$ is not able to answer the oracle query $H_1(ID_j) = u_j$ (or $H_1(ID_i) = u_i$) and $\mathcal{A}$ then cannot verify correctness of the received public key αP for ID_j (or ID_i) under the master public key sP.

In conclusion, since $\mathcal{B}$ cannot answer some H_1 queries, $\mathcal{A}$ can immediately notice the inconsistency between the simulation and the real world.

Problem 2: $\mathcal{B}$ in Proof 1 and Proof 3 requires that $\mathcal{A}$ provides an expected value as a response to a specific oracle. This is not a reasonable requirement.

This is because $\mathcal{A}$ is not controlled by $\mathcal{B}$. Even the assumption that the adversary would follow the protocol strictly to generate messages is too strong to cover many dangerous attacks. A sound reduction can only require that messages from the adversary are in the specified message space at most.

Problem 3: H_2 is not clearly required in the MB protocol and its variants. This results in that the reduction to the computational BIDH assumption does not follow.

The simulation $\mathcal{B}$ cannot be created based on the BIDH assumption for the original protocols in [17, 18, 31]. Instead, even if both Problem 1 and 2 are solved, a simulation could only be created based on the decision BIDH for the original protocols if they are secure. Otherwise, the adversary can win the game with the probability to differentiate a random element of $\mathbb{G}_2$ from the true value. This is the reason why we employ an extra hash function on the agreed secret to generate a session key SK.

5 The Modified Scheme and Its Security Analysis

As pointed in [8], Scheme 2 does not achieve the known-session key security (i.e., the compromise of one session key would not affect other session keys' security), and so cannot be proved in the Bellare-Rogaway key agreement model with the Reveal query allowed (note that the reduction in [18] proceeds in the model with the Reveal query disallowed, though it is invalid). However, we can slightly tweak the session key generation method in Scheme 2 to recover from the attack, furthermore the modification enables us to reduce the security of the scheme to the k-Gap-BCAA1 assumption in the full Bellare-Rogaway model. The modified scheme, which we refer to as **Scheme** $2'$, is as follow:

$$SK = H_3(A, B, T_1, T_2, \hat{e}(P, P)^{x+y}),$$

where $H_3 : \{0,1\}^* \times \{0,1\}^* \times \mathbb{G}_1 \times \mathbb{G}_1 \times \mathbb{G}_2 \to \{0,1\}^n$ is a hash function.

Before presenting the details of the proof, we should note that as pointed out in [11], when receiving a message T the party in general should check that T is in the specified message space, in this case, $\mathbb{G}_1^*$, then to accept the session. Otherwise, potential attacks are available, for example, if T is with small order, so could K be, and the adversary can enumerate the result group to compromise K.

Theorem 2 *Scheme 2′ is a secure AK, provided that H_1, H_3 are random oracles and the k-Gap-BCAA1 assumption is sound.*

Proof: The conditions 1 and 2 directly follow from the protocol specification. In the sequel we prove that the protocol satisfies the condition 3.

Suppose that there is an adversary $\mathcal{A}$ against the protocol with non-negligible probability $n(k)$. Let q_1 and q_2 be the number of the distinct queries to H_1 and H_3 respectively (note that H_1 could be queried directly by an H_1-query or indirectly by a *Corrupt* query or a *Send* query). With the help of $\mathcal{A}$, we can construct an algorithm $\mathcal{B}$ to solve a $(q_1 - 1)$-Gap-BCAA1 problem with non-negligible probability.

Given an instance of the $(q_1 - 1)$-Gap-BCAA1 problem $(\mathbb{G}_1, \mathbb{G}_2, \hat{e}, q, (P, sP, h_0, (h_1, \frac{1}{h_1+s}P), \ldots, (h_{q_1-1}, \frac{1}{h_{q_1-1}+s}P))$ where $h_i \in_R \mathbb{Z}_q^*$ for $0 \leq i \leq q_1 - 1$ and the DBIDH oracle $\mathcal{O}_{\texttt{DBIDH}}$, $\mathcal{B}$ simulates the **Setup** algorithm to generate the system **params** $(\mathbb{G}_1, \mathbb{G}_2, \hat{e}, q, P, sP, H_1, H_3)$ (i.e., using s as the **master key** which it does not know). H_1 and H_3 are two random oracles controlled by $\mathcal{B}$. Suppose, in the game, there are T_1 oracles created by $\mathcal{A}$. Here, we slightly abuse the notation $\Pi_{i,j}^s$ as the s-th oracle among all the oracles created during the attack, instead of the s-th instance of i. This change does not affect the soundness of the model because s originally is just used to uniquely identify an instance of party i. $\mathcal{B}$ randomly chooses $u \in_R \{1, \ldots, T_1\}$ and $I \in_R \{1, \ldots, q_1\}$ and interacts with $\mathcal{A}$ in the following way:

H_1**-queries** (ID_i)**:** $\mathcal{B}$ maintains a list of tuples (ID_i, h_i, d_i) as explained below. We refer to this list as H_1^{list}. The list is initially empty. When $\mathcal{A}$ queries the oracle H_1 at a point on ID_i, $\mathcal{B}$ responds as follows:

1. If ID_i already appears on the H_1^{list} in a tuple (ID_i, h_i, d_i), then $\mathcal{B}$ responds with $H_1(ID_i) = h_i$.
2. Otherwise, if the query is on the I-th distinct ID, then $\mathcal{B}$ stores $(ID_I, h_0, \perp)$ into the tuple list and responds with $H_1(ID_I) = h_0$.
3. Otherwise, $\mathcal{B}$ selects a random integer $h_i (i > 0)$ from the $(q_1 - 1)$-Gap-BCAA1 instance which has not been chosen by $\mathcal{B}$ and stores $(ID_i, h_i, \frac{1}{h_i+s}P)$ into the tuple list. $\mathcal{B}$ responds with $H_1(ID_i) = h_i$.

H_3**-queries** $(ID_i, ID_j, T_i, T_j, K_t)$**:** At any time $\mathcal{A}$ can issue queries to the random oracle H_3. To respond to these queries $\mathcal{B}$ maintains a list of tuples called H_3^{list}. Each entry in the list is a tuple of the form $(ID_i, ID_j, T_i, T_j, K_t, \zeta_t)$ indexed by $(ID_i, ID_j, T_i, T_j, K_t)$. To respond to a query, $\mathcal{B}$ does the following operations:

1. If on the list there is a tuple indexed by $(ID_i, ID_j, T_i, T_j, K_t)$, then $\mathcal{B}$ responds with ζ_t.
2. Otherwise, $\mathcal{B}$ goes through the list Λ built in the Reveal query to find tuples of the form $(ID_i, ID_j, T_i, T_j,\ r_i^t, \zeta_i^t)$ and proceeds as follow:
   - Compute $D = K_t/(P,P)^{r_i^t}$.
   - Access $\mathcal{O}_{\texttt{DBIDH}}(P,(h_0{+}s)P, T_j, D)$. If $\mathcal{O}_{\texttt{DBIDH}}$ returns 1, $\mathcal{B}$ inserts $(ID_i, ID_j, T_i, T_j, K_t, \zeta_i^t)$ into H_3^{list} and responds with ζ_i^t to the query and removes the tuple from Λ.
3. Otherwise, $\mathcal{B}$ randomly chooses a string $\zeta_t \in \{0,1\}^n$ and inserts a new tuple $(ID_i, ID_j, T_i, T_j, K_t, \zeta_t)$ into the list H_3^{list}. It responds to $\mathcal{A}$ with ζ_t.

Corrupt(ID$_i$): $\mathcal{B}$ looks through list H_1^{list}. If ID_i is not on the list, $\mathcal{B}$ queries $H_1(ID_i)$. $\mathcal{B}$ checks the value of d_i: if $d_i \neq \perp$, then $\mathcal{B}$ responds with d_i; otherwise, $\mathcal{B}$ aborts the game (**Event 1**).

Reveal$(\Pi_{j,i}^t)$**:** $\mathcal{B}$ maintains an initially empty list Λ with tuples $(ID_i, ID_j, T_i, T_j, r_i^t, \zeta_i^t)$. $\mathcal{B}$ responds to the query as follow:

- Go through $H_1^{list}(ID_j)$ to find the private key d_j of party j with identity ID_j.
- If $d_j \neq \perp$, compute $K = \hat{e}(T_i, d_j) \cdot \hat{e}(P,P)^{r_{j,i}^t}$ where T_i is the incoming message and $r_{j,i}^t$ is the random flips of the oracle $\Pi_{j,i}^t$. $\mathcal{B}$ responds with $H_3(ID_j, ID_i, T_j, T_i, K)$ if the oracle is the initiator, otherwise $H_3(ID_i,\ ID_j,\ T_i,\ T_j,\ K)$.
- Otherwise, $\mathcal{B}$ goes through H_3^{list} to find tuples indexed by (ID_j, ID_i, T_j, T_i) (if $\Pi_{j,i}^t$ is the initiator) or by (ID_i, ID_j, T_i, T_j) (if $\Pi_{j,i}^t$ is the responder). For each (K_t, ζ_t) in the found tuples,
  - Compute $D = K_t/\hat{e}(P,P)^{r_{j,i}^t}$.
  - Access $\mathcal{O}_{\texttt{DBIDH}}(P,(h_0+s)P, T_i, D)$. If $\mathcal{O}_{\texttt{DBIDH}}$ returns 1, then $\mathcal{B}$ responds to the query with ζ_t. Note that there can be at most one K_t meeting the test.
- Otherwise (no K_t is found in the last step), randomly choose $\zeta_t \in \{0,1\}^n$ and insert $(ID_j, ID_i, T_j, T_i, r_{j,i}^t, \zeta_t)$ if the oracle is the initiator, or $(ID_i,\ ID_j,\ T_i,\ T_j,\ r_{j,i}^t,\ \zeta_t)$ into Λ. $\mathcal{B}$ responds with ζ_t.

Send$(\Pi_{j,i}^t, M)$**:** $\mathcal{B}$ first looks through the list H_1^{list}. If ID_i is not on the list, $\mathcal{B}$ queries $H_1(ID_i)$. After that, $\mathcal{B}$ checks the value of t. If $t \neq u$, $\mathcal{B}$ responds the query by honestly following the protocol. If $t = u$, $\mathcal{B}$ further checks the value of d_i, and then responds the query differently as below depending on this value.

- If $d_i \neq \perp$, $\mathcal{B}$ aborts the game (**Event 2**). We note that there is only one party's private key is represented as $\perp$ in the whole simulation.
- Otherwise, $\mathcal{B}$ randomly chooses $y \in \mathbb{Z}_q^*$ and responds with yP. Note that $\Pi_{j,i}^t$ can be the initiator (if $M = \lambda$) or the responder (if $M \neq \lambda$).

Test$(\Pi_{j,i}^t)$**:** If $t \neq u$, $\mathcal{B}$ aborts the game (**Event 3**). Otherwise, $\mathcal{B}$ randomly chooses a number $\zeta \in \{0,1\}^n$ and gives it to $\mathcal{A}$ as the response. When $\mathcal{A}$ responds, $\mathcal{B}$ goes through H_3^{list} and for each K_ℓ,

- Compute $D = (K_\ell / \hat{e}(d_j, M))^{1/y}$, where M is the incoming message to oracle $\Pi^t_{j,i}$.
- Access $\mathcal{O}_{\texttt{DBIDH}}(P, (h_0 + s)P, P, D)$. If $\mathcal{O}_{\texttt{DBIDH}}$ returns 1, $\mathcal{B}$ returns D as the response of the $(q_1 - 1)$-Gap-BCAA1 challenge.
- If no K_ℓ meets the test, fail the game.

Claim 1 *If algorithm $\mathcal{B}$ does not abort during the simulation, then algorithm $\mathcal{A}$'s view is identical to its view in the real attack.*

Proof: $\mathcal{B}$'s responses to H_1 queries are uniformly and independently distributed in $\mathbb{Z}_q^*$ as in the real attack. H_3 is modelled as a random oracle which requires that for each unique input, there should be only one response. We note that the simulation substantially makes use of the programmability of random oracle and the access to the DBIDH oracle to guarantee the unique response for every H_3 query. The responses in other types of query are valid as well. Hence the claim follows. □

Now let us evaluate the probability that $\mathcal{B}$ did not abort the game. $\mathcal{B}$ aborts the game only when at least one of following events happens: (1) **Event 1**, denoted as $\mathcal{H}_1$: $\mathcal{A}$ corrupted party i whose private key is represented by $\perp$ at some point; (2) **Event 2**, denoted as $\mathcal{H}_2$: in the u-th session, if $\mathcal{A}$ impersonated a party, it did not impersonate party i whose private key is represented by $\perp$ (recall that oracle $\Pi^u_{j,i}$ was simulated by $\mathcal{B}$ in the game. It is not important who sent the message M to $\Pi^u_{j,i}$. It could be the adversary who impersonated party i or an oracle $\Pi^v_{i,j}$ for some v); (3) **Event 3**, denoted as $\mathcal{H}_3$: $\mathcal{A}$ did not choose the u-th oracle as the challenge fresh oracle.

Note that according to the rules of the game, the adversary would not corrupt party i if it chose $\Pi^t_{j,i}$ as the fresh oracle. Hence $\neg\mathcal{H}_3 \wedge \neg\mathcal{H}_2$ implies $\neg\mathcal{H}_1$ (recall that in this work, we require that $j \neq i$ for the chosen fresh oracle $\Pi^t_{j,i}$, i.e., in the attacked session the victim party did not establish the session with itself. This is not an unusual requirement in the real practice). Let $\mathcal{F}$ be the event that $\mathcal{B}$ did not abort the game. Then, we have

$$\Pr[\mathcal{F}] = \Pr[\neg\mathcal{H}_1 \wedge \neg\mathcal{H}_2 \wedge \neg\mathcal{H}_3] = \Pr[\neg\mathcal{H}_2 \wedge \neg\mathcal{H}_3] \geq \frac{1}{q_1} \cdot \frac{1}{T_1}.$$

Claim 2 *The adversary has computed the agreed secret K in the challenge session with probability $n(k)$ if the game does not abort.*

Proof: Suppose in the game, the chosen fresh oracle for the Test query is $\Pi^t_{j,i}$ and party j is the initiator (the result holds if party i is the responder for the similar reason). Let $\mathcal{H}$ be the event that $(ID_j, ID_i, T^t_j, T^t_i, K^t_{j,i})$ has been queried to H_3 where T^t_j is the message from j and T^t_i is the message from party i or the adversary $\mathcal{A}$ and $K^t_{j,i}$ is the agreed secret. Now let us consider the following situations:

- $\mathcal{A}$ reveals $\Pi^w_{u,v}$ where $u \notin \{i, j\}$ or $v \notin \{i, j\}$ or $u = v = i$ or $u = v = j$. Because ID_u and ID_v, instead of ID_j and ID_i, will be queried as the identifier to H_3, event $\mathcal{H}$ will not be caused by such Reveal queries. Recall that for the chosen fresh oracle as the challenge, it is required that $i \neq j$.
- $\mathcal{A}$ reveals $\Pi^w_{i,j}$ where i is the initiator (if j is the responder for the chosen fresh oracle, then i is the responder). The Reveal query will only query in the form of $H_3(ID_i, ID_j, *, *, *)$, so event $\mathcal{H}$ will not be caused by this type of Reveal query.
- $\mathcal{A}$ reveals $\Pi^w_{i,j}$ where i is the responder (if j is the responder for the chosen fresh oracle, then i is the initiator). Because of the rule of the game, $\Pi^w_{i,j}$ has no matching conversation to $\Pi^t_{j,i}$, which means either $T^t_j \neq T^w_j$ or $T^t_i \neq T^w_i$ (recall that the session ID which is the message transcript of the session, is used to define matching conversations. Two oracles have matching conversation to each other if both have the same message transcript). Hence event $\mathcal{H}$ will not be caused by this type of Reveal query.
- $\mathcal{A}$ reveals $\Pi^w_{j,i}$ for $w \neq t$. Because for both oracles $\Pi^t_{j,i}$ and $\Pi^w_{j,i}$, j is not controlled by the adversary, it is only negligibly likely that $T^t_j = T^w_j$ if the used random flips are generated uniformly. Hence event $\mathcal{H}$ will not (or with only negligible probability) be caused by this type of Reveal query.

From the above analysis, we know that the Reveal queries won't cause $\mathcal{H}$. In other words, $\mathcal{H}$ happens only if $\mathcal{A}$ has queried $H_3(ID_j, ID_i, T^t_j, T^t_i, K^t_{j,i})$. While as H_3 is a random oracle, we have $\Pr[\mathcal{A} \text{ wins}|\neg\mathcal{H}] = \frac{1}{2}$. Then we have

$$\begin{aligned} n(k) + \tfrac{1}{2} &= \Pr[\mathcal{A} \text{ wins}] \\ &= \Pr[\mathcal{A} \text{ wins}|\mathcal{H}] \Pr[\mathcal{H}] + \Pr[\mathcal{A} \text{ wins}|\neg\mathcal{H}] \Pr[\neg\mathcal{H}] \\ &\leq Pr[\mathcal{H}] + \tfrac{1}{2}. \end{aligned}$$

So, $\Pr[\mathcal{H}] \geq n(k)$, which means, $\mathcal{A}$ has computed $K^t_{j,i}$ with probability $n(k)$. □

Note the agreed secret in oracle $\Pi^t_{j,i}$ should be $K = \hat{e}(d_j, M) \cdot \hat{e}(P, P)^r$ where $r(h_0P + sP) = yP$ (recall that party i's public key is $h_0P + sP$ and the private key is unknown to $\mathcal{B}$ and represented by $\perp$), i.e., $r = \frac{y}{h_0+s}$ and $K = \hat{e}(d_j, M) \cdot \hat{e}(yP, \frac{1}{h_0+s}P)$. From Claim 2 we know that the event, denoted by $\mathcal{H}'$, that $\mathcal{A}$ has computed K and queried it on H_3, happens with probability $n(k)$ ($\Pr[\mathcal{H}'] \geq n(k)$).

Overall, we have

$$\Pr[\mathcal{B} \text{ wins}] = \Pr[\mathcal{F} \wedge \mathcal{H}'] \geq \frac{1}{q_1 \cdot T_1} \cdot n(k).$$

This completes the security analysis of the protocol. □

6 Conclusion

In this paper, we have revisited the security proofs of three identity-based authenticated key agreement schemes using the Sakai and Kasahara key construction and pointed out the flaws in the reductions. We have also slightly modified McCullagh and Barreto's second protocol and provided a new reduction for the modified scheme based on the k-Gap-BCAA1 assumption in the Bellare-Rogaway's key agreement formulation.

References

1. D. Boneh and X. Boyen. Efficient selective-ID secure identity based encryption without random oracles. In *Proceedings of Advances in Cryptology - Eurocrypt'2004*, LNCS 3027, pp. 223-238, 2004.
2. D. Boneh and X. Boyen. Secure identity-based encryption without random oracles. In *Proceedings of Advances in Cryptology - Crypto 2004*, LNCS 3152, pp. 443-459, 2004.
3. D. Boneh and M. Franklin. Identity based encryption from the Weil pairing. In *Proceedings of Advances in Cryptology - Crypto 2001*, LNCS 2139, pp. 213-229, 2001.
4. M. Bellare, D. Pointcheval and P. Rogaway. Authenticated key exchange secure against dictionary attacks. In *Proceedings of Proceedings of Advances in Cryptology - Eurocrypt 2000*, LNCS 1807, pp. 139-155, 2000.
5. M. Bellare and P. Rogaway. Entity authentication and key distribution. In *Proceedings of Advances in Cryptology - Crypto '93*, LNCS 773, pp. 232-249, 1993.
6. S. Blake-Wilson, D. Johnson and A. Menezes. Key agreement protocols and their security analysis. In *Proceedings of the Sixth IMA International Conference on Cryptography and Coding*, LNCS 1355, pp. 30-45, 1997.
7. Z. Cheng. Private communication. 2004.
8. K.-K. R. Choo. Revisit of McCullagh–Barreto two-party ID-based authenticated key agreement protocols. Cryptology ePrint Archive, Report 2004/343.
9. K.-K. R. 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. In *Proceedings of Mycrypt 2005*, LNCS 3715, pp. 116-131, 2005.
10. L. Chen and Z. Cheng. Security proof of the Sakai-Kasahara's identity-based encryption scheme. In *Proceedings of Cryptography and Coding 2005*, LNCS 3706, pp. 442-459, 2005.
11. Z. Cheng, L. Chen, R. Comley and Q. Tang. Identity-based key agreement with unilateral identity privacy using pairings. To appear in ISPEC 2006. Full version avaialbe on Cryptology ePrint Archive, Report 2005/339.
12. L. Chen, Z. Cheng, J. Malone-Lee and N. Smart. An efficient ID-KEM based on the Sakai-Kasahara key construction. To appear in *IEE proceedings Information Security.*
13. L. Chen and C. Kudla. Identity based authenticated key agreement from pairings. In *Proceedings of the 16th IEEE Computer Security Foundations Workshop*, pp. 219-233, June 2003.
14. Z. Cheng, M. Nistazakis, R. Comley and L. Vasiu. On the indistinguishability-based security model of key agreement protocols-simple cases. In *Proceedings of ACNS 2004 (technical track).* The full paper available on Cryptology ePrint Archive, Report 2005/129.
15. ISO/IEC 11770-3:1999. Information technology - Security techniques - Key management - Part 3: Mechanisms using asymmetric techniques.

16. S. Li, Q. Yuan and J. Li. Towards security two-party authenticated key agreement protocols. Cryptology ePrint Archive, Report 2005/300.
17. N. McCullagh and P.S.L.M. Barreto. A new two-party identity-based authenticated key agreement. In *Proceedings of CT-RSA 2005*, LNCS 3376, pp. 262-274, 2005.
18. N. McCullagh and P. S. L. M. Barreto. A new two-party identity-based authenticated key agreement. Cryptology ePrint Archive, Report 2004/122.
19. A. Menezes, P. van Oorschot and S. Vanstone. Handbook of applied cryptography. CRC Press, 2001.
20. S. Mitsunari, R. Sakai and M. Kasahara. A new traitor tracing. *IEICE Trans. Fundamentals*, E85-A(2):481–484, 2002.
21. E. Okamoto. Proposal for identity-based key distribution system. *Electronics Letters*, 22:1283-1284, 1986.
22. M. Scott. Authenticated ID-based key exchange and remote log-in with insecure token and PIN number. Cryptology ePrint Archive, Report 2002/164, 2002. http://eprint.iacr.org/2002/164.
23. A. Shamir. Identity-based cryptosystems and signature schemes. In *Proceedings of Advances in Cryptology - Crypto '84*, LNCS 196, pp.47-53, 1985.
24. N. P. Smart. An Identity Based Authenticated Key Agreement Protocol Based on the Weil Pairing. *Electronics Letters* 38 (2002), pp. 630-632. See also Cryptology ePrint Archive, Report 2001/111.
25. R. Sakai and M. Kasahara. ID based cryptosystems with pairing on elliptic curve. Cryptology ePrint Archive, Report 2003/054.
26. R. Sakai, K. Ohgishi and M. Kasahara. Cryptosystems based on pairing. In *Proceedings of 2000 Symposium on Cryptography and Information Security*, Okinawa, Japan, 2000.
27. N. Smart and F. Vercauteren. On computable isomorphisms in efficient piaring based systems. Cryptology ePrint Archive, Report 2005/116.
28. K. Tanaka and E. Okamoto. Key distribution system for mail systems using ID-related information directory. *Computers & Security*, 10:25-33, 1991.
29. Y. Wang. Efficient identity-based and authenticated key agreement protocol. Cryptology ePrint Archive, Report 2005/108.
30. G. Xie. Cryptanalysis of Noel McCullagh and Paulo S. L. M. Barreto's two-party identity-based key agreement. Cryptology ePrint Archive, Report 2004/308.
31. G. Xie. An ID-based key agreement scheme from pairing. Cryptology ePrint Archive, Report 2005/093.
32. F. Zhang, R. Safavi-Naini and W. Susilo. An efficient signature scheme from bilinear pairings and its applications. In *Proceedings of International Workshop on Practice and Theory in Public Key Cryptography - PKC 2004*, LNCS 2947, pp. 277-290, 2004.