Perfect Forward Secure Identity-Based Authenticated Key Agreement Protocol in the Escrow Mode

Shengbao Wang[1], Zhenfu Cao[1], Zhaohui Cheng[2], and Kim-Kwang Raymond Choo[3]

[1] Department of Computer Science and Engineering,
Shanghai Jiao Tong University,
800 Dongchuan Road, Shanghai 200240, China
[2] School of Computing Science, Middlesex University
The Burroughs, Hendon, London, UK
[3] Canberra, Australian Capital Territory, Australia
{shengbao-wang, cao-zf}@cs.sjtu.edu.cn
m.z.cheng@mdx.ac.uk
raymond.choo.au@gmail.com

Abstract. There are several essential features in key agreement protocols such as key escrow (essential when confidentiality, audit trail and legal interception are required) and *perfect* forward secrecy (i.e., the security of a session key established between two or more entities is guaranteed even when the private keys of the entities are compromised). Majority of the existing escrowable identity-based key agreement protocols, however, only provide *partial* forward secrecy. Therefore, such protocols are unsuitable for real-word applications that require a stronger sense of forward secrecy — *perfect* forward secrecy. In this paper, we propose an efficient perfect forward secure identity-based key agreement protocol in the escrow mode. We prove the security of our protocol in the random oracle model, assuming the intractability of the Gap Bilinear Diffie-Hellman (GBDH) problem. Security proofs are invaluable tools in assuring protocol implementers about the security properties of protocols. We note, however, that many existing security proofs of previously published identity-based protocols entail lengthy and complicated mathematical proofs. In this paper, our proof adopts a modular approach and, hence, simpler to follow.

Key words: Authenticated key agreement; Perfect forward secrecy; Bilinear pairing; Provable security; Modular security proof

1 Introduction

Key agreement protocols are fundamental to establishing secure communications between two or more parties over an insecure network. A key establishment protocol (including key agreement protocol) allows two or more communicating parties to establish a common secret (session) key via public communication channels (e.g., internet). The established session key can then be used to create a confidential or integrity-protected communication channel between the parties. *Authenticated key agreement (AK)* protocols not only allow parties to compute the session key but also ensure the authenticity of the involved parties [5].

Authenticated key agreement protocols can be built based on both secret-key cryptography and public-key cryptography. If secret-key cryptography is used, then either a symmetric secret key or a shared password should be distributed before the key agreement protocol is executed. If traditional (certificate-based) public-key cryptography is used, a public key infrastructure (PKI) will typically be required to be deployed that allows authentication of registered users' public keys. In 1984, Shamir proposed the concept of identity-based cryptography [21] whereby each party's public key can be an arbitrary string (typically an identity string) and, hence, removes the need for certificates. This, therefore, greatly simplified the management of public keys in the identity-based cryptosystems. Following the work of Boneh and Franklin on identity-based encryption [1], several identity-based two-party key agreement schemes using bilinear pairings on elliptic curves have been proposed (see [6]).

Motivation 1. Secure key agreement protocol in the escrow mode

ID-based authenticated key agreement protocols may either work in the escrowed mode (i.e., the private key generator (PKG) is able to recover the session keys established by its users) or escrowless mode (i.e., the PKG is unable to recover the session keys established by its users). Majority of the ID-based key agreement protocols are designed to be escrowless for privacy concerns. However, as noted in [4], key escrow is desirable under certain circumstances especially in certain closed groups applications. For example, escrow is *essential* in situations where confidentiality as well as audit trail are legal requirements, such as secure communications in the health care profession.

One example of an ID-based authenticated key agreement protocol in the escrow mode is the protocol of Chen and Kudla [12]. A drawback with this protocol (and also of Smart's ID-based authenticated key agreement protocol [20]) is that it does not provide perfect forward secrecy (i.e., compromise of the long-term secret keys belonging to any party will expose previously established corresponding session keys). Although Shim [22] proposed a protocol that claimed to provide such a (perfect forward secrecy) property, it was later found to be vulnerable to the man-in-the-middle attack [23].

Our first motivation is, therefore, to propose a new protocol that is secure in the escrow mode and achieves perfect forward secrecy.

Motivation 2. Simplifying the proof — a modular approach

It is by now standard practice for protocol designers to provide security proof in widely accepted security models in order to assure protocol implementors of their security properties (see [8]). As pointed out in a recent survey of two-party ID-based authenticated key agreement protocols [6], many existing protocols are not proven secure in the modular approach (e.g., the model of Canetti and Krawczyk [11]) and their proofs of security are often complicated and error-prone [8,17].

Kudla and Paterson [17,16] then developed a modular technique for constructing security proofs for a large class of key agreement protocols using a slightly modified Smart's protocol as an example. Informally, their modular technique works in the following sequence.

1. Prove that a protocol Π has the property of strong partnering.
2. Prove that a related protocol π is secure in a highly reduced security model.
3. The security proof for π in the reduced model is then translated into a security proof for Π in the full model using a Gap assumption [19].

The overall proof technique using the modular approach is far easier than the conventional approach and this forms the second motivation of our paper.

Our contributions

1. We propose an efficient ID-based AK protocol (with only a single online pairing computation) that works in the escrow mode. With comparable performance, our new protocol achieves perfect forward secrecy.

2. Using the modular technique of Kudla and Paterson, we prove that our protocols is secure in the random oracle model, provided that the Gap Bilinear Diffie-Hellman (GBDH) problem is hard. Our protocol is the first Chen–Kudla-type key agreement protocol to be proven secure in a modular approach.

Paper organization The remainder of this paper is structured as follows. In Section 2, we briefly describe bilinear pairings, the computational problems and the corresponding complexity assumptions, the security model, and the Kudla–Paterson modular proof approach [16] required in this paper. Section 3 revisits the protocol due to Chen and Kudla [12]. We present our proposed ID-based authenticated key agreement protocol (hereafter referred to as E-IBAK) in Section 4. In Section 5, a detailed security proof in the random oracle model [3] of our proposed E-IBAK protocol is provided. Section 6 provides a performance comparison between several related ID-based protocols. We draw our conclusions in Section 7.

2 Preliminaries

2.1 Bilinear Pairings and GBDH Probolem

Let $\mathbb{G}_1$ denotes an additive group of prime order q and $\mathbb{G}_2$ a multiplicative group of the same order. We let P denote a generator of $\mathbb{G}_1$. For us, an admissible pairing is a map $\hat{e} : \mathbb{G}_1 \times \mathbb{G}_1 \rightarrow \mathbb{G}_2$ with the following properties:

1. The map $\hat{e}$ is bilinear: given $Q, R \in \mathbb{G}_1$ and $a, b \in \mathbb{Z}_q^*$, we have $\hat{e}(aQ, bR) = \hat{e}(Q, R)^{ab}$.
2. The map $\hat{e}$ is non-degenerate: $\hat{e}(P, P) \neq 1_{\mathbb{G}_2}$.
3. The map $\hat{e}$ is efficiently computable.

Typically, the map $\hat{e}$ will be derived from either the Weil or Tate pairing on an elliptic curve over a finite field. We refer to [2,1,15] for a more comprehensive description of how these groups, pairings and other parameters should be selected in practice for efficiency and security.

Definition 1 (Bilinear Diffie-Hellman (BDH) Parameter Generator). *As in [1], we say that a randomized algorithm $\mathcal{IG}$ is a BDH parameter generator if $\mathcal{IG}$ takes a security parameter $l > 0$, runs in time polynomial in l, and outputs the description of two groups $\mathbb{G}_1$ and $\mathbb{G}_2$ of the same prime order q and the description of an admissible pairing $\hat{e} : \mathbb{G}_1 \times \mathbb{G}_1 \rightarrow \mathbb{G}_2$.*

The security of the ID-based key agreement protocols in this paper are based on the difficulty of the following problems:

Definition 2 (Bilinear Diffie-Hellman (BDH) Problem). *Let $\mathbb{G}_1$, $\mathbb{G}_2$, P and $\hat{e}$ be as above. The BDH problem in $\langle \mathbb{G}_1, \mathbb{G}_2, \hat{e} \rangle$ is as follows: Given $\langle P, aP, bP, cP \rangle$ with uniformly random choices of a, b, $c \in \mathbb{Z}_q^*$, compute $\hat{e}(P, P)^{abc} \in \mathbb{G}_2$.*

We say that a *probabilistic polynomial time* (PPT) algorithm B has advantage ϵ in solving the BDH problem in $\langle \mathbb{G}_1, \mathbb{G}_2, \hat{e} \rangle$ if

$$\Pr[\mathcal{B}(P, aP, bP, cP) = \hat{e}(P, P)^{abc}] \geq \epsilon$$

where the probability is measured over the random choices of $a, b, c \in \mathbb{Z}_q^*$ and the random bits of $\mathcal{B}$.

The above BDH problem has a decisional counterpart called the decisional bilinear Diffie-Hellman (DBDH) problem which is defined as follows.

Definition 3 (Decisional Bilinear Diffie-Hellman (DBDH) Problem). *Let $\mathbb{G}_1$, $\mathbb{G}_2$, P and $\hat{e}$ be as above. The* DBDH *problem in $\langle \mathbb{G}_1, \mathbb{G}_2, \hat{e} \rangle$ is as follows: Given $\langle P, aP, bP, cP \rangle$ with uniformly random choices of a, b, $c \in \mathbb{Z}_q^*$, as well as $W \in \mathbb{G}_2$, determine if $\hat{e}(P,P)^{abc} = W$ (if it holds, then the tuple $\langle P, aP, bP, cP, W \rangle$ is called a* BDH *tuple).*

The BDH and DBDH problems can be used to define a related Gap problem [19].

Definition 4 (Gap Bilinear Diffie-Hellman GBDH Problem). *Let $\mathbb{G}_1$, $\mathbb{G}_2$, P and $\hat{e}$ be as above. The* GBDH *problem in $\langle \mathbb{G}_1, \mathbb{G}_2, \hat{e} \rangle$ is as follows: Given $\langle P, aP, bP, cP \rangle$ with uniformly random choices of a, b, $c \in \mathbb{Z}_q^*$, as well as an oracle the solves the* DBDH *problem in $\langle \mathbb{G}_1, \mathbb{G}_2, \hat{e} \rangle$, compute $\hat{e}(P,P)^{abc}$.*

Informally, the BDH, DBDH and GBDH assumptions are that no PPT adversary has non-negligible advantage in solving the BDH, DBDH and GBDH problems, respectively.

2.2 Desirable Security Attributes

Let Alice (A) and Bob (B) be two honest entities, i.e., legitimate entities who execute the steps of a protocol correctly. Here we list up a number of desirable attributes of AK protocols which referred to [5,12].

- *Known-key secrecy (K-KS).* Each run of a key agreement between A and B should produce a unique secret session key. The compromise of one session key should not compromise the keys established in other sessions.
- *Perfect forward secrecy (PFS).* If long-term private keys of *all* entities are compromised, the secrecy of previous session keys established by honest entities is not affected.
- *Key-compromise impersonation (K-CI) resilience.* Suppose $A's$ private key is disclosed. Obviously, an adversary who knows this key can impersonate A to other entities (e.g. B). However, it is desired that this disclosure does not allow the adversary to impersonate any *other* entity (e.g. B) to A.
- *Unknown key-share (UK-S) resilience.* Entity A cannot be coerced into sharing a key with entity B without $A's$ knowledge, i.e., when A believes that the key is shared with some entity $C \neq B$, and B (correctly) believes the key is shared with A.
- *No key control.* Neither entity should be able to force the session key (or any portion of the session key) to a preselected value.

In addition to these security attributes, it would be desirable for a protocol to have low computational cost (the computing operations needed for A and B to finish a run of the protocol) and low communication overhead (which means that only a small amount of data is exchanged) for its practical use.

2.3 Security Model for ID-Based AK Protocols — ID-mBJM

In this subsection, we present our refined formal security model for ID-based authenticated key agreement protocols. Kudla [17] proposed the so called ID-BJM model, which is an extension of the model of Blake-Wilson *et al.* [7] (known as the BJM model). In this paper, we extend a modified version of the BJM model to the ID-based setting which we call the ID-mBJM model. Following the approach of Choo *et al.* [9], we use the notion of session identifier SID (instead of *matching conversation* used in the BJM model and Kudla's ID-BJM model) in our partnership definition.

The model includes a set U of participants modeled by a collection of *oracles* (e.g., oracle $\Pi^n_{I,J}$ represents the n-th instance of participant I carrying out a protocol session in the belief that it is communicating with another participant J. Each participant has a long-term ID-based public/private key pair, in which the public key is generated using her identity information and the private one is computed and issued secretly by a private key generator.

There is an active adversary (denoted by E) in the model modeled by a PPT Turing Machine which has access to all the participants' oracles[4]. Participant oracles only respond to queries by the adversary and do not communicate directly among themselves, i.e., there exists *at least* a benign adversary who simply passes messages between participants faithfully.

Definition of security in the model depends on the notion of the *partner* oracles to any oracle being tested. We define partners by having the same session identifier (SID). Concretely, we define $\mathrm{SID}(\Pi^n_{I,J})$ as the concatenation of all messages that oracle $\Pi^n_{I,J}$ has sent and received.

Definition 5 (Partner). *Two oracles $\Pi^n_{I,J}$ and $\Pi^{n'}_{J,I}$ are said to be partner oracles if they have accepted with the same SID.*

The security of a protocol is defined via a two-phase adaptive game (called the ID-mBJM game) between a *challenger* $\mathcal{C}$ that simulates a set of participant oracles running the protocol and the adversary E. $\mathcal{C}$ also simulates the PKG in this environment, and therefore generates the public parameters of the PKG and gives these to E. $\mathcal{C}$ also generates a master secret s from which it can generate a private key d_I from any given identity I.

In the first phase, the adversary E is allowed to issue the following queries in any order.

Send(I, J, n, M): E can send message M to oracle $\Pi^n_{I,J}$. The oracle executes the protocol and responds with an outgoing message m or a decision to indicate accepting or rejecting the session. Any incoming and outgoing message is recorded on its transcript. If $M = \lambda$ (denotes the null message), then the oracle initiates a protocol run.

Reveal$(\Pi^n_{I,J})$: To respond to the query, oracle $\Pi^n_{I,J}$ returns the session key if the session has been accepted. Otherwise, a symbol $\perp$ is returned. Such an oracle, $\Pi^n_{I,J}$, is then considered *opened*.

Corrupt(I): Upon receiving this query, $\mathcal{C}$ outputs the private key d_I of the participant I. A participant is called *corrupted* if a Corrupt query has been issued to it.

Test$(\Pi^n_{I,J})$: At some point, E can make a Test query to some *fresh oracle* $\Pi^n_{I,J}$ (see Definition 6 below). To answer the query $\mathcal{C}$ flips a fair coin $b \in \{0, 1\}$; if the answer is 0, then $\mathcal{C}$ outputs the agreed session key of the test oracle, otherwise outputs a randomly chosen value from the session key space.

In the second phase, E is allow to continue asking Send, Reveal and Corrupt queries to the oracles, except that E is not allowed to reveal the target test oracle or its partner oracle (if any), and E cannot corrupt participant J (assuming $\Pi^n_{I,J}$ is the test oracle).

[4] If the proof is given in the random oracle model (ROM), then the adversary also has access to all the existing random oracles.

Output: Finally, E outputs a prediction (b') on b. E wins the game if $b' = b$, and we define E's advantage (l is the security parameter) in winning the game as

$$Adv^E(l) = |\Pr[b' = b] - 1/2|.$$

Definition 6 (Fresh Oracle). *An oracle $\Pi^n_{I,J}$ $(I \neq J)$ is called* fresh *if it has accepted (and therefore holds a session key sk_i), it is not opened, J has not been corrupted, and there is no opened oracle $\Pi^{n'}_{J,I}$ which is a partner oracle of $\Pi^n_{I,J}$.*

Remark 1. The above definition of fresh oracle is particularly defined to cover the security attribute of *key-compromise impersonation resilience* since it implies that the participant I could have been issued a Corrupt query [13].

Definition 7 (ID-mBJM Secure Protocol). *A protocol is a secure AK protocol in the ID-mBJM model if:*

1. *In the presence of the* benign adversary *(who faithfully relays messages between parties) on $\Pi^n_{I,J}$ and $\Pi^{n'}_{J,I}$, both oracles always accept holding the same session key, and this key is distributed uniformly on session key space.*
2. *$Adv^E(l)$ is negligible.*

It is easy to see that the above definition of security does not include the property of perfect forward secrecy (PFS). To model PFS, the definition of fresh oracle (refer to Definition 6) should be modified so that the the participants associated with the Test (fresh) oracle can also be corrupted. We define PFS as follows.

Definition 8 (Perfect Forward Secrecy (PFS)). *A protocol is said to have perfect forward secrecy (PFS) if any PPT adversary wins the ID-mBJM game with negligible advantage when it chooses an unopened oracle $\Pi^n_{I,J}$ which has a an unopened partner oracle $\Pi^{n'}_{J,I}$ as the test oracle, and both oracles $\Pi^n_{J,I}$ and $\Pi^{n'}_{J,I}$ accepted and both participants I and J can be corrupted.*

Note that as in [18], here we refer to the practical notion of perfect forward secrecy that involves a benign adversary eavesdropping on a session of the protocol and then attempting to expose the key.

2.4 Modular Proof Technique for ID-Based AK Protocols

Here we briefly review the modular technique for proving the security for key agreement protocols [17]. Note that here we mainly focused on ID-based protocols.

The modular proof technique only works on key agreement protocols that produce hashed session keys on completion of the protocol. This reliance on hashing to produce a session key is reasonable since it is fairly common to use a *key derivation function* (KDF) to derive a session key from a secret value established during a key agreement protocol, and this key derivation function is usually implemented via a hash function. In the security proof, the key derivation function will be modeled as a random oracle.

Definition 9 (Session String). *Suppose Π is a protocol that produces a hashed session key using the cryptographic hash function H. Then the session string for a particular oracle $\Pi^i_{I,J}$ is denoted $ss_{\Pi^i_{I,J}}$, and is defined to be the string which is hashed to produce the session key $sk_{\Pi^i_{I,J}}$. So we have that $sk_{\Pi^i_{I,J}} = H(ss_{\Pi^i_{I,J}})$.*

Strong Partnering. Suppose Π is a key agreement protocol. If there exists an adversary E, which when attacking Π in an ID-mBJM game defined in Section 2.3 and with non-negligible probability in the security parameter l, can make some two oracles $\Pi^i_{I,J}$ and $\Pi^n_{J,I}$ accept holding the same session key when they are not partners, then we say that Π has *weak partnering*. If Π does not have weak partnering, then we say that it has *strong partnering*.

As shown in [17], for a protocol Π to be ID-mBJM secure, it must have strong partnering. Since H is modeled as a random oracle, strong partnering can be ensured by including appropriate "*partnering information*" in the session string $ss_{\Pi^i_{I,J}}$, where partnering information is used to decide whether the two oracles are partners or not. In Section 4, we will use the session identifier SID and the identities of the two parties as the partnering information of our new protocol.

Reduced Games. A highly reduced game (called th *cNR-ID-mBJM* game) is used in the modular security proof. The reduced game is identical as the full ID-mBJM game defined in Section 2.3 except that the adversary E is not allowed to make Reveal queries and to win the game, E must select an accepted fresh oracle on which to make a modified Test query at the end of its attack and output the session key held by this oracle. We define E's advantage, denoted $Adv^E(l)$, in the cNR-ID-mBJM game to be the probability that E outputs a session key sk such that $sk = sk_{\Pi^i_{I,J}}$ where $\Pi^i_{I,J}$ is the oracle selected by the adversary for the modified Test query. We define security in the reduced game as follows:

Definition 10 (cNR-ID-mBJM Secure Protocol [17]). *A protocol Π is a secure key agreement protocol in the cNR-ID-mBJM model if:*

1. *In the presence of the benign adversary, two oracles running the protocol both accept holding the same session key, and the session key is distributed uniformly at random on session key space.*
2. *For any adversary E, $Adv^E(l)$ in the reduced game is negligible.*

As part of the the proof technique, it will be necessary to prove that a related protocol π of protocol Π is secure in the above reduced game.

Related Protocol π. The related protocol π of protocol Π is defied in the same way as Π except that the session key generated by π is defined to be the session string of Π rather than the hash of this string (i.e., $sk_{\pi^n_{I,J}} = ss_{\Pi^n_{I,J}}$). It is usually quite easy to establish a related protocol's security in the reduced game.

Definition 11 (Session String Decisional Problem). *Given the public parameters, the transcript $T_{\Pi^n_{I,J}}$ of oracle $\Pi^n_{I,J}$, as well as the public keys of I and J and a string s, decide whether $s = ss_{\Pi^n_{I,J}}$, where $ss_{\Pi^n_{I,J}}$ is the session string of oracle $\Pi^n_{I,J}$.*

The following result is at the heart of the modular proof technique that translates the weak security of a related weaker protocol into the security of the protocol in the full model.

Theorem 1 (Theorem 8.2 in [17]). *Suppose that key agreement protocol Π produces a hashed session key on completion of the protocol (via hash function H) and that Π has strong partnering. If the security of the related protocol π in the reduced game is probabilistic polynomial time reducible to the hardness of the computational problem of some relation f, and the session string decisional problem for Π is polynomial time reducible to the decisional problem of f, then the security of Π in the full model is probabilistic polynomial time reducible to the hardness of the Gap problem of f, assuming that H is a random oracle.*

3 Review of the CK Protocol

In this section, we briefly review the ID-based authenticated key agreement protocol due to Chen and Kudla (the CK protocol) [12], which requires only 1 pairing evaluation for each party. The CK protocol is illustrated in Figure 1.

Alice		**Bob**
$a \in_R \mathbb{Z}_q^*$, $T_A = aQ_A$		$b \in_R \mathbb{Z}_q^*$, $T_B = bQ_B$
	$\xrightarrow{T_A}$	
	$\xleftarrow{T_B}$	
		$K_{BA} = \hat{e}(d_B,\ bQ_A + T_A)$ $sk_{BA} = H(A, B, T_A, T_B, K_{BA})$
$K_{AB} = \hat{e}(d_A,\ aQ_B + T_B)$ $sk_{AB} = H(A, B, T_A, T_B, K_{AB})$		

$$sk_{AB} = H(A, B, T_A, T_B, \hat{e}(Q_A, Q_B)^{s(a+b)}) = sk_{BA}$$

Fig. 1. The CK Protocol [12]

There are three entities in the protocol: two users Alice and Bob who wish to establish a shared secret session key, and a PKG from whom they each acquire their own private keys. The protocol consists of two stages: Setup and Key Agreement.

Setup: Suppose we have an admissible pairing $\hat{e} : \mathbb{G}_1 \times \mathbb{G}_1 \rightarrow \mathbb{G}_2$ as described in Section 2, where $\mathbb{G}_1$ and $\mathbb{G}_2$ are two groups with the same prime order q. The PKG follows the following steps:

1. picks an arbitrary generator $P \in \mathbb{G}_1$, a secret master key $s \in \mathbb{Z}_q^*$;
2. chooses a cryptographic hash function $H_1 : \{0, 1\}^* \rightarrow \mathbb{G}_1$;
3. publishes the system parameters $params = \langle \mathbb{G}_1, \mathbb{G}_2, e, q, P, H_1 \rangle$;
4. computes the private key $d_{ID} = sQ_{ID}$ for a user with the identity information ID, in which the user's public key is $Q_{ID} = H_1(ID)$;
5. distributes the private key d_{ID} to the user with the identity information ID via a secure channel.

Thus, each user's identity-based public/private key pair is defined as (Q_{ID}, d_{ID}) where Q_{ID}, $d_{ID} \in \mathbb{G}_1$.

Key Agreement: To agree on a common session key, both Alice and Bob randomly choose an ephemeral private key, $a, b \in \mathbb{Z}_q^*$ respectively, and compute the corresponding ephemeral public keys

$$T_A = aQ_A$$

and

$$T_B = bQ_B.$$

They then exchange the ephemeral public keys as depicted in Figure 1.

Alice and Bob respectively computes K_{AB} and K_{BA} as follows,

$$K_{AB} = \hat{e}(d_A,\ aQ_B + T_B)\ ,\quad K_{BA} = \hat{e}(d_B,\ bQ_A + T_A).$$

If Alice and Bob follow the protocol, they will compute shared secrets of equal value, K_{AB} and K_{BA} respectively:

$$K = K_{AB} = K_{BA} = \hat{e}(Q_A,\ Q_B)^{s(a+b)}.$$

The value of the established shared secret, K, is therefore suitable to be used to derive a shared session key. We then use a key derivation function $H : \{0,1\}^* \times \{0,1\}^* \times \mathbb{G}_1 \times \mathbb{G}_1 \times \mathbb{G}_2 \rightarrow \{0,1\}^k$ to generate the shared session key. $|sk|$ denotes the length of the binary representation of sk (i.e., k=$|sk|$):

$$sk = H(A, B, T_A, T_B, K).$$

The security proof of the CK protocol in the ID-BJM model [17] illustrated that the protocol has the following security attributes: known-key secrecy, key-compromise impersonation resilience, unknown key-share resilience, and no key control.

Drawbacks of the CK protocol, however, include the following:

1. *Lack of PFS:* Suppose the private keys of Alice and Bob (d_A and d_B) are compromised, an adversary can easily recover previously established session keys between Alice and Bob as shown below:

$$K = \hat{e}(Q_A, Q_B)^{s(a+b)} = \hat{e}(d_A, T_B)\hat{e}(d_B, T_A).$$

2. *Unable to adopt the modular proof approach:* As pointed out by Kudla [17], the CK protocol cannot be proven using the modular approach as the session string decisional problem cannot be reduced to a decisional problem. We demonstrate that our proposed protocol (in the next section) can use the modular technique to prove its security.

4 Proposed Identity-Based AK Protocol

Our scheme employs the ID-based non-interactive key sharing protocol due to Sakai *et al.* [24] (hereafter referred to as the SOK protocol). The SOK protocol works as follows:

- The Setup stage of the SOK protocol is similar to that of the CK protocol.
- Once two users, Alice and Bob, have established their respective public/private key pairs, (Q_A, d_A) and (Q_B, d_B), they can establish a common secret F (called the SOK secret) statically (i.e., without any message exchange):

$$F = \hat{e}(d_A, Q_B) = \hat{e}(d_B, Q_A) = \hat{e}(Q_A, Q_B)^s.$$

We now describe our proposed escrowable identity-based authenticated key agreement protocol with perfect forward secrecy (hereafter referred to as E-IBAK).

As with the CK protocol and all other identity-based cryptosystems we assume the existence of a trusted PKG that is responsible for the generation and secure distribution of users' private keys. Our key agreement protocol can be implemented using either the modified Weil or Tate pairing [1,4]. The new protocol consists of the following two stages:
Setup: This stage is identical to that of the CK protocol (refer to Section 3).

Alice		**Bob**
$F = \hat{e}(d_A, Q_B) = \hat{e}(Q_A, Q_B)^s$		$F = \hat{e}(d_B, Q_A) = \hat{e}(Q_A, Q_B)^s$
$a \in_R \mathbb{Z}_q^*$, $T_A = aQ_A$		$b \in_R \mathbb{Z}_q^*$, $T_B = bQ_B$
	$\xrightarrow{T_A}$	
	$\xleftarrow{T_B}$	
		$K_{BA_1} = \hat{e}(d_B,\ T_A)$
		$K_{BA_2} = K_{BA_1}^b$
		$sk_{BA} = H(A, B, T_A, T_B, K_{BA_1}, F^b, K_{BA_2})$
$K_{AB_1} = \hat{e}(d_A,\ T_B)$		
$K_{AB_2} = K_{AB_1}^a$		
$sk_{AB} = H(A, B, T_A, T_B, F^a, K_{AB_1}, K_{AB_2})$		

$$sk_{AB} = H(A, B, T_A, T_B, F^a, F^b, F^{ab}) = sk_{BA}$$

Fig. 2. Proposed Protocol E-IBAK

Key Agreement: We denote user Alice and Bob's public/private key pairs as (Q_A, d_A) and (Q_B, d_B), respectively. We assume that Alice and Bob both pre-compute and store the above mentioned SOK (non-interactively shared) secret F.

To establish a shared session key, Alice and Bob each firstly generate an ephemeral private key (say a and $b \in \mathbb{Z}_q^*$), and compute the corresponding ephemeral public keys $T_A = aQ_A$ and $T_B = bQ_B$, respectively. They then exchange T_A, T_B and compute the session key as described in Figure 2. As in the CK protocol, $H : \{0,1\}^* \times \{0,1\}^* \times \mathbb{G}_1 \times \mathbb{G}_1 \times \mathbb{G}_2 \times \mathbb{G}_2 \times \mathbb{G}_2 \rightarrow \{0,1\}^k$ is a key derivation function (in which $k = |sk|$).

Protocol Correctness. By the bilinearity of the pairing, we can easily get the following equations:

$$K_{AB_1} = \hat{e}(d_A,\ T_B) = \hat{e}(d_A, bQ_B) = F^b,$$
$$K_{BA_1} = \hat{e}(d_B,\ T_A) = \hat{e}(d_B, aQ_A) = F^a$$

and $K_{AB_2} = K_{BA_2} = F^{ab}$.

Thus, the two session keys computed by Alice and Bob are

$$sk_{AB} = sk_{BA} = H(A, B, T_A, T_B, F^a, F^b, F^{ab}).$$

Escrow. The protocol E-IBAK has the escrow function, namely the PKG can recover all the session keys using the master secret key s and other public data such as T_A and T_B. We prove this as follows.

$$\begin{aligned} F^a &= \hat{e}(Q_A, Q_B)^{sa} = \hat{e}(T_A, Q_B)^s, \\ F^b &= \hat{e}(Q_A, Q_B)^{sb} = \hat{e}(Q_A, T_B)^s, \\ F^{ab} &= \hat{e}(Q_A, Q_B)^{sab} = \hat{e}(T_A, T_B)^s. \end{aligned}$$

The protocol is message independent and *role symmetric*, which means that each party performing the same operations and thus incurring the same computational cost. In the next section we will prove that our protocol E-IBAK achieves the ID-mBJM security (see Definition 10) as well as perfect forward secrecy.

5 Security Proof

We prove the security (i.e. ID-mBJM security plus PFS) of our new protocol $\mathsf{E\text{-}IBAK}$ in stages. We first give a basic identity-based protocol, $\mathsf{E\text{-}IBAK}'$, which does not provide perfect forward secrecy, and prove that it is ID-mBJM secure using the Kudla–Paterson modular technique. We then prove that the protocol $\mathsf{E\text{-}IBAK}$ is also secure in the ID-mBJM model and provides perfect forward secrecy. The only reason for describing the protocol $\mathsf{E\text{-}IBAK}'$ is to make the presentation easier to follow.

Protocol $\mathsf{E\text{-}IBAK}'$ is almost identical to protocol $\mathsf{E\text{-}IBAK}$ except that the final session key is computed as

$$sk_{AB} = H'(A, B, T_A, T_B, F^a, F^b),$$

where $H' : \{0,1\}^* \times \{0,1\}^* \times \mathbb{G}_1 \times \mathbb{G}_1 \times \mathbb{G}_2 \times \mathbb{G}_2 \rightarrow \{0,1\}^k$ is a key derivation function. In other words, without the value F^{ab} being part of the session string. With the description of the ID-mBJM model in Section 2.3, we now state:

Theorem 2 (ID-mBJM Security of $\mathsf{E\text{-}IBAK}'$). *If H' and H_1 are random oracles and the GBDH problem (for the pair of groups $\mathbb{G}_1$ and $\mathbb{G}_2$) is hard, then* $\mathsf{E\text{-}IBAK}'$ *is a secure key agreement protocol.*

We now prove Theorem 2 in three steps. We first show that protocol $\mathsf{E\text{-}IBAK}'$ has strong partnering. Secondly, we prove that the related protocol π of $\mathsf{E\text{-}IBAK}'$ is secure in the cNR-ID-mBJM model (see Definition 10). Lastly, we show that the session string decisional problem (see Definition 11) of $\mathsf{E\text{-}IBAK}'$ is reducible to the DBDH problem.

Lemma 1 (Strong Partnering of $\mathsf{E\text{-}IBAK}'$). *Protocol* $\mathsf{E\text{-}IBAK}'$ *has strong partnering in the random oracle model.*

Proof. It is easy to verify that this condition holds because the partnering information, namely the protocol transcript and participant IDs are included in the session string. Recall that we model H' as a random oracle, thus if two oracles end up holding the same session key, then they are not partners with only negligible probability. □

Lemma 2 (cNR-ID-mBJM Security of π). *The related protocol π is secure in the cNR-ID-mBJM model, assuming the* BDH *problem is hard (for the pair of groups $\mathbb{G}_1$ and $\mathbb{G}_2$) and provided that H_1 is a random oracle.*

Proof. Condition 1 follows from the correctness of the protocol π. Since H' is a random oracle, sk is distributed uniformly at random on $\{0,1\}^k$. In the following, we show that Condition 2 is also satisfied.

For a contradiction, assume that the adversary E has non-negligible advantage ϵ in winning the cNR-ID-mBJM game, making at most q_1 queries to H_1. Let q_S be the total number of the oracles that E creates, i.e., for any oracle Π^n_{AB}, $n \in \{1, ..., q_S\}$. We shall slightly abuse the notation Π^n_{AB} to refer to the n-th one among all the q_S participant instances in the game, instead of the n-th instance of participant A. As n is only used to help identify oracles, this notation change will not affect the soundness of the model.

We show how to construct a simulator S that uses E as a sub-routine to solves the BDH problem with non-negligible probability. Given input of the two groups $\mathbb{G}_1$, $\mathbb{G}_2$, the bilinear map $\hat{e}$, a generator P of $\mathbb{G}_1$, and a triple of elements xP, yP, $zP \in \mathbb{G}_1$ with $x, y, z \in \mathbb{Z}^*_q$ where q is the prime order of $\mathbb{G}_1$ and $\mathbb{G}_2$, $S's$ task is to compute and output the value $\hat{e}(P,P)^{xyz}$.

The algorithm S selects a random integer v from $\{1, ..., q_1\}$ and a random integer w from $\{1, ..., q_S\}$ and works by interacting with E as follows:

Setup: S sets the PKG's master key to be xP. S will also simulate all oracles required during the game. S controls the H_1 random oracle. S starts E, and answers all E's queries as follows.

$H_1(ID_i)$: S simulates the random oracle H_1 by keeping a list of tuples $\langle r_i, ID_i, Q_i\rangle$ which is called the H_1-List. When the H_1 oracle is queried with an input $ID_i \in \{0,1\}^*$, S responds as follows.

- If ID_i is already on the H_1-List in the tuple $\langle r_i, ID_i, Q_i\rangle$, then S outputs Q_i.
- Otherwise, if ID_i is the v-th distinct H_1 query, then the oracle outputs $Q_i = yP$ and adds the tuple $\langle \perp, ID_i, Q_i\rangle$ to the H_1-List.
- Otherwise S selects a random $r_i \in \mathbb{Z}_q^*$ and outputs $Q_i = r_iP$, and then adds the tuple $\langle r_i, ID_i, Q_i\rangle$ to the H_1-List.

We assume that J is the v-th distinct participant created in the game.

Corrupt(ID_i): S simulates the Corrupt query on input ID_i as follows.

- If $ID_i \neq J$, S outputs the corresponding long-term private key d_i.
- Otherwise abort the game (**Event 1**).

Send(A, B, t, M): S answers the queries as follows.

– If $t \neq w$, S randomly samples $\xi_t \in \mathbb{Z}_q^*$ and responds with $\xi_t Q_A$ where $Q_A = H_1(A)$.
– Otherwise,
  - If $B \neq J$, abort the game (**Event 2**).
  - Otherwise answer zP.

Test$(\Pi^t_{A,B})$: At some point in the simulation, E will ask a single Test query of some oracle. If E does not choose the guessed oracle $\Pi^w_{A,B}$ to ask the Test query, then S aborts (**Event 3**).

Output: At the end of the game, the algorithm E outputs a session key of the form (U, V, a, b, c, d) where $U, V \in \{0,1\}^*$, $a, b \in \mathbb{G}_1$ and $c, d \in \mathbb{G}_2$.

Solving the BDH Problem: If $\Pi^w_{A,B}$ was an initiator oracle, then S outputs c as its guess for the value $\hat{e}(P,P)^{xyz}$, otherwise S outputs d as its guess.

Now we evaluate the probability that the simulation does not abort. If the adversary indeed has chosen the w-th oracle as the test oracle and that oracle supposes to establish a session key with party J, then by the rules of the game **Event 1, 2** and **3** would not happen. We have

$$\Pr[S \text{ does not abort}] \geq \frac{1}{q_S q_1}.$$

Note that participant J has the public key $Q_J = yP$ and private key $d_J(= xyP)$. Given a message zP, part of the agreed secret is $\hat{e}(xyP, zP)$. So if the adversary computes the correct session key with non-negligible probability ϵ, then S answers the BDH problem correctly with probability with $\epsilon/(q_S q_1)$ (which is non-negligible in the security parameter l), contradicting to the hardness of the BDH problem. $\square$

Lemma 3 (Session String Decisional Problem of E-IBAK′). *The session string decisional problem of protocol* E-IBAK′ *is reducible to the* DBDH *problem.*

Proof. Recall the session string of protocol E-IBAK′ is of the form $(A, B, T_A, T_B, F^a, F^b)$ with $T_A = aQ_A$, $F^a = \hat{e}(d_B, T_A) = \hat{e}(sQ_B, T_A)$, s being the master secret key and P, sP being the public parameters, then we see $\langle P, sP, Q_B, T_A, F^a \rangle$ (similarly, $\langle P, sP, Q_A, T_B, F^b \rangle$) is a BDH tuple. This implies that the session string decisional problem of protocol E-IBAK′ is reducible to the DBDH problem (in constant time). □

Proof of Theorem 2. The theorem follows directly from Lemma 1, 2, 3 and Theorem 1. □

Now we are ready to prove our main result — the security of our newly proposed protocol E-IBAK.

Theorem 3 (Security of Protocol E-IBAK). *Protocol* E-IBAK

i) is secure in the ID-mBJM model, assuming the GBDH *problem (for the pair of groups* $\mathbb{G}_1$ *and* $\mathbb{G}_2$*) is hard and provided that* H *and* H_1 *are random oracles, and*
ii) has the property of perfect forward secrecy (PFS)*, assuming the* BDH *problem (for the pair of groups* $\mathbb{G}_1$ *and* $\mathbb{G}_2$*) is hard and provided that* H *and* H_1 *are random oracles.*

Proof. i) This follows directly from Theorem 2, since it is easy to see that any successful attack on protocol E-IBAK can be immediately converted to a successful attack on protocol E-IBAK′.

ii) According to our definition of perfect forward secrecy (see Definition 8), we require that when E chooses an oracle $\Pi^n_{I,J}$ as the test oracle, this oracle must indeed have a partner oracle $\Pi^{n'}_{J,I}$. As before, we let Π^n_{AB} denote the n-th oracle among all the oracles created in the game.

The proof follows along similar lines to the proof of Lemma 2. For a contradiction, we assume that the adversary E can win the game with non-negligible advantage ϵ by creating at most q_S oracles and making q_H queries to the H random oracle. We show how to construct a simulator S that uses E as the sub-routine to solves the BDH problem with non-negligible probability. Identical to the proof of Lemma 2, the input of S are the two groups $\mathbb{G}_1$, $\mathbb{G}_2$, the bilinear map $\hat{e}$, a generator P of $\mathbb{G}_1$, and a triple of elements xP, yP, $zP \in \mathbb{G}_1$ with $x, y, z \in \mathbb{Z}_q^*$ where q is the prime order of $\mathbb{G}_1$ and $\mathbb{G}_2$, its task is to compute and output the value $\hat{e}(P, P)^{xyz}$.

The algorithm S selects two random integers u, v from $\{1, ..., q_S\}$ (assuming $u < v$) and works by interacting with E as follows:

Setup: S sets the PKG's master key to be xP. S will also simulate all oracles required during the game. S controls two random oracles H_1 and H. S starts E, and answers all E's queries as follows.

$H_1(ID_i)$**:** S simulates the oracle H_1 by keeping a list of tuples $\langle r_i,\ ID_i,\ Q_i \rangle$ which is called the H_1-List. When the H_1 oracle is queried with an input $ID_i \in \{0,1\}^*$, S responds as follows.

- If ID_i is already on the H_1-List in the tuple $\langle r_i, ID_i, Q_i \rangle$, then S outputs Q_i.
- Otherwise S selects a random $r_i \in \mathbb{Z}_q^*$ and outputs $Q_i = r_i P$, and then adds the tuple $\langle r_i, ID_i, Q_i \rangle$ to the H_1-List.

$H(ID_i, ID_j, T_i, T_j, A_i, B_i, C_i)$**:** S simulates the random oracle H by keeping an H-List with tuples of the form $\langle ID_i, ID_j, T_i, T_j, A_i, B_i, C_i, k_i \rangle$. If the requested input is already on the list, then the corresponding k_i is returned, otherwise a random $k_i \in \{0,1\}^k$ is responded and a new entry is inserted into the list.

Corrupt(ID_i): Upon receiving the $\mathsf{Corrupt}$ query on input ID_i, S outputs the corresponding long-term private key $d_i = r_i xP$.

Send(A, B, t, M): S answers all Send queries as follows

- When $t = u$, if oracle $M \neq \lambda$, then abort (**Event 1**), otherwise return yP.
- When $t = v$, if $M \neq yP$, then abort (**Event 2**), otherwise return zP.
- When $t \neq u, v$, randomly sample $\xi_t \in \mathbb{Z}_q^*$, return $\xi_t H_1(A)$.

Reveal($\Pi^t_{A,B}$): Upon receiving a Reveal query, S outputs the appropriate session key, except if E asks the oracle $\Pi^u_{A,B}$ or $\Pi^v_{A,B}$, then S aborts (**Event 3**). Note that S can compute the agreed session secret given ξ_t, the input message and the private key d_A.

Test($\Pi^t_{A,B}$): If E does not choose the guessed oracle $\Pi^u_{A,B}$ or $\Pi^v_{A,B}$ to ask the Test query, then S aborts (**Event 4**). To answer the Test query, S randomly picks a value β from the session key space and responds to E with β.

Output: At the end of the game, the algorithm E outputs its guess.

Solving the BDH Problem: S picks a tuple of the form $\langle I, J, T_I, T_J, A_h, B_h, C_h\rangle$ (for some h) from the H-List and returns C_h as the response to the BDH challenge.

Now we evaluate the probability that S does not abort, namely **Event 1, 2, 3** and **4** do not happen. By the rule of the game, if the test session is between the u-th and v-th oracle, then the simulation goes through. The probability that the simulator has chosen the right session is $1/q_S^2$, because a randomly chosen oracle is the initiator of the test session is $1/q_S$ and similarly another randomly chosen oracle is the responder of the test session is also $1/q_S$. We have

$$\Pr[S \text{ does not abort}] \geq 1/q_S^2.$$

According to the simulation of the Send query, the test oracle $\Pi^u_{I,J}$ must have obtained the value $T_J = zP$ from its partner oracle $\Pi^v_{J,I}$. The oracle should hold a session key of the form $H(I, J, T_I, T_J, A_h, B_h, \hat{e}(d_I, T_J)^{y/r_I})$, in which $\hat{e}(d_I, T_J)^{y/r_I} = \hat{e}(xr_I P, zP)^{y/r_I} = \hat{e}(P, P)^{xyz}$.

Let Q be the event that the session string of the test oracle has been queried to H. Because of the construction of the session string and the use of session identifier to define partner oracles, we can easily prove that if Q happens with non-negligible probability in the random oracle model, it must be caused by a query issued by the adversary (see the detailed argument in [10]). Because H is a random oracle, we have $\Pr[E \text{ wins}|\bar{\mathsf{Q}}] = 1/2$. Then

$$\begin{aligned}
\Pr[E \text{ wins}] &= \Pr[E \text{ wins}|\bar{\mathsf{Q}}]\Pr[\bar{\mathsf{Q}}] + \Pr[E \text{ wins}|\mathsf{Q}]\Pr[\mathsf{Q}] \\
&\leq \Pr[E \text{ wins}|\bar{\mathsf{Q}}]\Pr[\bar{\mathsf{Q}}] + \Pr[\mathsf{Q}] \\
&= \tfrac{1}{2}\Pr[\bar{\mathsf{Q}}] + \Pr[\mathsf{Q}] \\
&= \tfrac{1}{2} + \tfrac{1}{2}\Pr[\mathsf{Q}].
\end{aligned}$$

$$\begin{aligned}
\Pr[E \text{ wins}] &= \Pr[E \text{ wins}|\bar{\mathsf{Q}}]\Pr[\bar{\mathsf{Q}}] + \Pr[E \text{ wins}|\mathsf{Q}]\Pr[\mathsf{Q}] \\
&\geq \Pr[E \text{ wins}|\bar{\mathsf{Q}}]\Pr[\bar{\mathsf{Q}}] \\
&= \tfrac{1}{2}\Pr[\bar{\mathsf{Q}}] \\
&= \tfrac{1}{2} - \tfrac{1}{2}\Pr[\mathsf{Q}].
\end{aligned}$$

It follows that $\Pr[\mathsf{Q}] \geq 2|\Pr[E \text{ wins} - 1/2]| = 2\epsilon$.

Combining all the above results, we have that S solves the BDH problem with probability at least $2\epsilon/(q_S^2 q_H)$ (which is non-negligible in the security parameter l), contradicting to the hardness of the BDH problem. $\square$

6 Comparison with Existing Escrowable Protocols

Here we summarize the security properties and performances of our E-IBAK protocol and several other previously published escrowable protocols in Table 1. Note that:

- Since all listed protocols offer the basic security properties (i.e., known-key secrecy, unknown key-share resilience, key-compromise impersonation resilience and no key control), we will restrict our comparison to only the forward secrecy property.
- In practice, pre-computation is often carried out prior to the execution of the protocol for better performance. We, therefore, compare only the on-line computation complexity of these protocols.

In the table, ✓ and × denote that the property holds and does not hold in the protocol respectively. We also use the following symbols to explain the computation complexity of each protocol. For simplicity, we only count these computationally expensive operations:

- $\mathbb{P}$: pairing.
- $\mathbb{M}$: scalar point multiplication in $\mathbb{G}_1$.
- $\mathbb{E}$: exponentiation in $\mathbb{G}_2$.
- $\mathbb{A}$: point addition in in $\mathbb{G}_1$.

Table 1. Comparisons of escrowable key agreement protocols (with pre-computation)

↓Protocols / Items→	$\mathbb{P}$	$\mathbb{M}$	$\mathbb{E}$	$\mathbb{A}$	Bandwidth	PFS	Extensible*
Protocol CK [12]	1	0	0	1	1 point	×	✓
Smart's [20]	1	0	0	0	1 point	×	✓
Wang's [25]	1	1	0	2	1 point	✓	✓
E-IBAK	1	0	1	0	1 point	✓	✓

* This indicates that if the protocol is extensible to work in the escrowless mode.

From the table, we observe that only Wang's protocol [25] and our proposed protocol E-IBAK achieve perfect forward secrecy (PFS) in the escrow mode. Our protocol is, however, more efficient than that of Wang's especially when we take into consideration that certain computations can be performed off-line.

Finally, it is worth noting that the escrowable protocols listed in Table 1 can be extended to work in the escrowless mode, using the simple idea due to Chen and Kudla [12] by embedding a raw Diffie–Hellman protocol [14] (interested readers are referred to [12] for the details). This reflects the flexibility of these key agreement protocols.

7 Conclusions

Perfect forward secrecy (PFS) is an important security property for authenticated key agreement protocols (in both escrow and escrowless modes). We presented an identity-based authenticated key agreement protocol that is provably secure in the escrow mode. We demonstrated that our proposed protocol provides *perfect* forward secrecy without compromising on computational efficiency. We also proved the security of our proposed protocol in a widely accepted model yet with a simpler modular proof (using the modular technique due to Kudla and Paterson [16]).

Acknowledgment

The first author would like to thank Liqun Chen for many valuable discussions. This work was partially supported by the National High Technology Development Program of China under Grant No. 2006AA01Z424 and the National Natural Science Foundation of China (Nos. 60572155 and 60673079).

References

1. D. Boneh, M. Franklin. Identity-based encryption from the Weil pairing. In *Proc. of CRYPTO 2001*, LNCS vol. 2139, pp. 213-229, Springer-Verlag, New York, 2001.
2. P.S.L.M. Barreto, H.Y. Kim and B. Lynn. Efficient algorithms for pairing-based cryptosystems. In *Proc. CRYPTO 2002*, LNCS vol. 2442, pp. 354-368. Springer-Verlag, New York,2002.
3. M. Bellare, P. Rogaway. Entity authentication and key distribution. In *Proc. of CRYPTO 1993*, LNCS vol. 773, pp. 110-125, Springer-Verlag, New York, 1993.
4. N. McCullagh, P.S.L.M. Barreto. A new two-party identity-based authenticated key agreement. In *Proc. of CT-RSA 2005*, LNCS vol. 3376, pp. 262-274, Springer-Verlag, New York, 2005.
5. S. Blake-Wilson, A. Menezes. Authenticated Diffie-Hellman key agreement protocols. In *Proc. of SAC 1998*, LNCS vol. 1556, pp. 339-361, Springer-Verlag, New York, 1999.
6. C. Boyd and K.-K. R. Choo. Security of two-party identity-based key agreement. In *Proc. of MYCRYPT 2005*, LNCS vol. 3715, pp. 229-243, Springer-Verlag, New York, 2005.
7. S. Blake-Wilson, C. Johnson and A. Menezes. Key agreement protocols and their security analysis. In *Proc. of the sixth IMA International Conference on Cryptography and Coding*, LNCS vol. 1355, pp. 30-45, Springer-Verlag, New York, 1997.
8. K.-K. R. Choo, C. Boyd, and Y. Hitchcock. Errors in computational complexity proofs for protocols. In *Proc. of ASIACRYPT 2005*, LNCS vol. 3788, pp. 624-643, Springer-Verlag, New York, 2005.
9. K.-K. R. Choo, C. Boyd, Y. Hitchcock, and G. Maitland. On session identifiers in provably secure protocols: The Bellare-Rogaway three-party key distribution protocol revisited. In *Proc. of SCN 2004*, LNCS vol. 3352, pp. 351-366, Springer-Verlag, New York, 2005.
10. Z. Cheng and L. Chen. On security proof of McCullagh-Barreto's key agreement protocol and its variants. *International Journal of Security and Networks* 2(3/4), pp. 251-259, 2007.
11. R. Canetti and H. Krawczyk. Analysis of key-exchange protocols and their use for building secure channels. In *Proc. of EUROCRYPT'01*, LNCS vol. 2045, pp. 453-474, Springer-Verlag, New York, 2001.
12. L. Chen, C. Kudla. Identity based key agreement protocols from pairings. In *Proc. of the 16^{th} IEEE Computer Security Foundations Workshop*, pp. 219-213, IEEE Computer Society, 2002. See also Cryptology ePrint Archive, Report 2002/184.
13. Z. Cheng, M. Nistazakis, R. Comley and L. Vasiu. On the indistinguishability-based security model of key agreement protocols - simple cases. In *Proc. of ACNS 2004 (technical track)*. The full paper available on Cryptology ePrint Archive, Report 2005/129
14. W. Diffie, M.E. Hellman. New directions in cryptography. *IEEE Trans. Inf. Theory* 22(6), pp. 644 - 654, 1976.
15. S.D. Galbraith, K. Harrison, and D. Soldera. Implementing the Tate pairing. In *Proc. of ANTS-V*, LNCS vol. 2369, pp. 324-337, Springer-Verlag, New York, 2002.
16. C. Kudla and K. G. Paterson. Modular security proofs for key agreement protocols. In *Proc. of ASIACRYPT'05*, LNCS vol. 3788, pp. 549-565, Springer-Verlag, New York, 2005.
17. C. Kudla. Special signature schemes and key agreement protocols. PhD Thesis, Royal Holloway University of London, 2006.
18. H. Krawczyk. HMQV: A high performance secure Diffie-Hellman protocol. In *Proc. of Crypto'05*, LNCS 3621, pp. 546-566, Springer-Verlag, New York, 2005.

19. T. Okamoto and D. Pointcheval. The Gap-problems: a new class of problems for the security of cryptographic schemes. In *Proc. of PKC 2001*, LNCS vol. 1992, pp. 104-118, Springer-Verlag, New York, 2001.
20. N.P. Smart. An identity based authenticated key agreement protocol based on the Weil pairing. *Electronics Letters* 38(13), pp. 630-632, 2002.
21. A. Shamir. Identity-based cryptosystems and signature schemes. In *Proc. of CRYPTO 1984*, LNCS vol.196, pp. 47-53, Springer-Verlag, New York, 1984.
22. K. Shim. Efficient ID-based authenticated key agreement protocol based on theWeil pairing.*Electronics Letters* 39(8), pp. 653-654, 2003.
23. H. Sun, B. Hsieh. Security analysis of Shim's authenticated key agreement protocols from pairings. Cryptology ePrint Archive, Report 2003/113, 2003. Available at http://eprint.iacr.org/2003/113.
24. R. Sakai, K. Ohgishi and M. Kasahara. Cryptosystems based on pairing. In *Proc. of the 2000 Symposium on Cryptography and Information Security*, Okinawa, Japan, 2000.
25. Y. Wang. Efficient identity-based and authenticated key agreement protocol. Cryptology ePrint Archive, Report 2005/108, 2005. Available at http://eprint.iacr.org/2005/108.