Security Proof for the Improved Ryu–Yoon–Yoo Identity-Based Key Agreement Protocol ⋆

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

[1] Department of Computer Science and Engineering, Shanghai Jiao Tong University, China
{shengbao-wang,cao-zf}@cs.sjtu.edu.cn

[2] Independent Security Researcher, Canberra, Australian Capital Territory, Australia
raymond.choo.au@gmail.com

[3] Information Security Research Center, National Institute of Information and Communications Technology, Japan
wlh@nict.go.jp

November 15, 2007

Abstract. Key agreement protocols are essential for secure communications in open and distributed environments. The protocol design is, however, extremely error-prone as evidenced by the iterative process of fixing discovered attacks on published protocols. We revisit an efficient identity-based (ID-based) key agreement protocol due to Ryu, Yoon and Yoo. The protocol is highly efficient and suitable for real-world applications despite offering no resilience against key-compromise impersonation (K-CI). We then show that the protocol is, in fact, insecure against reflection attacks. A slight modification to the protocol is proposed, which results in significant benefits for the security of the protocol without compromising on its efficiency. Finally, we prove the improved protocol secure in a widely accepted model.

Key Words. Key agreement protocols; Reflection attack; Provable security; Modular proof; PKG forward secrecy

1 Introduction

Key agreement protocols are designed to provide secure communications between two or more parties in a hostile environment. For example, a two-party key agreement protocol allows two communicating parties to establish a common secret key – *session key* – via a public communication channel. The session key can subsequently be used to establish a secure communication channel between both parties. If a party in the protocol is assured that no other party other than the designated party (or parties) can gain access to the particular session key, then the protocol is said to provide *implicit key authentication* (IKA). An authenticated key agreement (AK) protocol provides mutual IKA between (or among) parties.

Diffie and Hellman [18] proposed the first efficient (un-authenticated) two-party key agreement protocol in 1976. Since then, many key agreement protocols have been proposed with

⋆ Note that the RYY (RYY^+) protocol analyzed in this paper can be seen as the ID-based version of the famous UMP [1] protocol. For more information on the relations between authenticated Diffie-Hellman and ID-based authenticated key agreement protocols, see [35].

⋆⋆ The views and opinions expressed in this paper are those of the author and do not reflect those of any organization with which the author may be affiliated. This research was undertaken in the author's personal capacity.

various security properties. We refer interested reader to [9] for a summary of security properties for key agreement protocols.

Identity-based (ID-based) cryptography was first introduced by Shamir in 1984 [30]. The basic idea behind an ID-based cryptosystem is that end users can choose arbitrary strings such as email addresses as their public keys and users' private keys are generated by a trusted Private Key Generator (PKG), which eliminates much of the overhead associated with key management. In 2001, Boneh and Franklin [5] published the first feasible solution for ID-based encryption (IBE) using Weil pairing on elliptic curves. Since then many ID-based key agreement protocols using pairings have been proposed. Examples include:

- In 2002, Smart [31] proposed an ID-based key agreement protocol based on the IBE scheme of Boneh and Franklin [5]. Shim [32] and Chen and Kudla [17] independently pointed out that Smart's protocol does not provide perfect forward secrecy — an important security requirement for key agreement protocols.
- In 2004, Shim [32] proposed an efficient ID-based key agreement protocol claiming to provide perfect forward secrecy, known-key secrecy, key-compromise impersonation (K-CI) resilience, and unknown key-share (UK-S) resilience (refer to Section 2.2 for the definitions of all these terms). Sun and Hsieh [33], however, pointed out that Shim's protocol is vulnerable to a man-in-the-middle attack which totally breaks the protocol.
- In 2004, Ryu *et al.* [29] proposed an efficient ID-based key agreement protocol using pairings in which computation and communication overheads for establishing a session key are significantly reduced. This protocol, best known for its efficiency, reduces the number of online pairing computation to zero. A year later, Boyd and Choo [4] and Wang *et al.* [37] independently showed that Ryu *et al.*'s protocol (hereafter referred to as the RYY protocol) does not provide K-CI resilience as claimed. Although no one has yet to show that the protocol is vulnerable to other attacks, the other (basic) security attribute (*i.e.* known-key secrecy, UK-S resilience and no key control) claims until now have never been formally proven.

In this paper, we show that the RYY protocol is, in fact, vulnerable to *reflection attacks* — in violation of the claim of the protocol designers. We then propose a slight modification to the RYY protocol (the RYY$^+$ protocol) and prove it secure in a widely accepted model using the modular proof technique of Kudla and Paterson [23].

The remainder of this paper is structured as follows. In Section 2, we briefly describe bilinear pairings, the complexity assumptions, the security model, and the Kudla–Paterson modular proof approach [23] required in this paper. Section 3 revisits the protocol due to Ryu *et al.* [29]. We present our improved protocol in the next section. In Section 5, a detailed security proof of our proposed RYY$^+$ protocol in the random oracle model [10] is provided. Performance comparison between several related published ID-based protocols is presented in Section 6. We then draw our conclusions in Section 7.

2 Preliminaries

2.1 Bilinear Pairings and Complexity Assumptions

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 $\hat{e} : \mathbb{G}_1 \times \mathbb{G}_1 \to \mathbb{G}_2$ 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 [7, 5, 19] 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 [5], 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 $\hat{e} : \mathbb{G}_1 \times \mathbb{G}_1 \to \mathbb{G}_2$.*

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 [27]. The security of the ID-based key agreement protocol in this paper is based on the difficulty of the GBDH problem:

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 $\hat{e}(P, P)^{abc}$.*

A real-valued function $f(l)$ is *negligible* if for any integer $n > 0$, $|f(l)| < l^{-n}$ for sufficiently large l. 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 Security Attributes

Security attributes for (ID-based) key agreement protocols have been identified in several previous work [8, 25, 9, 17]. We briefly explain the security attributes as follows (refer to [8, 25] for more detailed discussions):

- **Known-key secrecy (K-KS).** Suppose an established session key between two entities is disclosed, the adversary is unable to learn other established session keys.
- **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 the two protocol principals (A and B) can predetermine any portion of the shared session key being established between them.
– **Key-compromise impersonation (K-CI) resilience.** Assume that entities A and B are two principals. Suppose $A's$ secret key is disclosed. Obviously, an adversary who knows this secret key can impersonate A to other entities (*e.g.* B). However, it is desired that this disclosure does not allow the adversary to impersonate other entities (*e.g.* B) to A.
– **Perfect forward secrecy (PFS).** If both long-term secret keys of two entities (*i.e.* the protocol principals) are disclosed, the adversary is unable to derive old session keys established by that two entities.
– **PKG forward secrecy (PKG-FS).** If in an ID-based key agreement protocol, the master key known only to the PKG is disclosed, the adversary is unable to derive old session keys established by that two entities. Note this attribute implies that the PKG is not able to passively escrow any session key of its users.

Remark 1. Obviously, the preceding three attributes (i.e. K-KS, UK-S and no key control) are more fundamental, since they do not assume the compromise of any long-term private keys (of the protocol participants or the PKG).

For a protocol to be used in practice, it must at least satisfy the above three *basic* security attributes. On the other hand, although K-CI resilience is a very important security properties it is *not* required for all application scenarios, *e.g.*, when the long-term private keys are protected with extreme care, or when those keys are updated in a relatively short period of time. Similarly, forward secrecy is not required when short-term secrecy suffices [16]. Indeed, it is common to find in practice protocols that do not provide K-CI resilience and/or forward secrecy and still are *not* considered insecure. A well-known example of practical protocol without K-CI resilience is the Unified Model Protocol (UMP) due to Ankney, Johnson and Matyas [1], which is in the draft standards ANSI X9.42 [2], ANSI X9.63 [3], and IEEE P1363 [28].

Other than the above-mentioned security attributes, it is also desirable for key agreement protocols to have low computation overhead, low communication overhead (*i.e.* total number of bits transmitted), and minimal number of passes (*i.e.* the number of messages exchanged in a run of the protocol).

2.3 Security Model for ID-Based AK Protocols

In this subsection, we present our refined formal security model for ID-based authenticated key agreement protocols. Kudla [21] proposed the so called ID-BJM model, which is an extension of the model of Blake-Wilson *et al.* [6] (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.* [14], 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. 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).

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), it is not opened, both I and J have not been corrupted, and there is no opened oracle $\Pi^{n'}_{J,I}$ which is a partner oracle of $\Pi^n_{I,J}$.*

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 at random on session key space;*
2. *For any adversary E, $Adv^E(l)$ is negligible.*

In the following, we briefly discuss the attributes that the above security model captures. Recall the security attributes we described in Section 2.2, and here we examine them one by one.

- *Known-key secrecy (K-KS).* The property of known-key secrecy is implied by the above definitions of AK security. Since E is allowed to make Reveal queries to any oracles except for the target Test oracle $\Pi^n_{I,J}$ and its partner oracle $\Pi^{n'}_{J,I}$ to obtain any session keys. Even with the knowledge of many other session keys, E's ability to distinguish between the session key held by $\Pi^n_{I,J}$ and a random number is still negligible. That is to say, the knowledge of any other session keys does not help E to deduce any information about the tested session key.
- *Unknown key-share (UK-S) resilience.* The definition also imply the unknown key-share resilience property. If ID_I establishes a session key with ID_J though he believes that he is talking to ID_K, then there is an oracle $\Pi^n_{I,K}$ that holds this session key sk_{IK}. At the same time, there is an oracle $\Pi^{n'}_{J,I}$ that holds this session key sk_{IK}, for some n' (normally $n' = n$). During an unknown key share attack, the user ID_K may not know this session key. Since $\Pi^n_{I,K}$ and $\Pi^{n'}_{J,I}$ are not partner oracles, the adversary can make a Reveal query to $\Pi^{n'}_{J,I}$ to learn this session key before asking a Test query to $\Pi^n_{I,K}$. Thus the adversary will succeed for this Test query challenge (*i.e.*, the protocol is not secure) if the unknown key share attack is possible. By contradiction, a secure protocol in the model is resistant to the unknown key share attack.
- *No key control.* The above definition of AK security does not imply resilience to key control attacks that are launched by one of the protocol participants. However, key control attacks launched by an outside adversary are captured by the model. In the model, all participants are assumed to be honest participants. If the protocol can be proven secure in the model, then we can be assure that the session key established is distributed uniformly at random in the session key space. Otherwise, the adversary E must have a non-negligible ability to distinguish between the session key held by $\Pi^n_{I,J}$ and a random number.
- *Key-compromise impersonation (K-CI) resilience.* Since the model does not allow the adversary to make Test queries of corrupted oracles, the model does not capture the property of K-CI resilience.
- *Forward secrecy (PFS and PKG-FS).* The definition does not imply the property of PFS or PKG-FS. This is because the model does not allow the adversary to make Test queries of corrupted oracles and therefore does not model this type of attack.

We will also show that our improved protocol achieves PKG forward secrecy (PKG-FS). To model PKG-FS (which implies PFS), the definition of fresh oracle (refer to Definition 6) should be modified so that the PKG (and thus the two participants associated with the Test oracle) can also be corrupted. Note that as pointed out by Krawczyk [20], no two-pass key agreement protocol can achieve strong perfect forward secrecy. Hence here we refer to the weak notion of PKG forward secrecy that involves a benign adversary eavesdropping on a session of the protocol and then attempting to expose the key. We define (weak) PKG forward secrecy as follows.

Definition 8 (PKG Forward Secrecy (PKG-PFS)). *An ID-based key agreement protocol is said to have PKG-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 an unopened partner oracle $\Pi^{n'}_{J,I}$ as the test oracle, and both oracles accepted. In addiction, the PKG can be corrupted.*

2.4 Modular Proof Technique for ID-Based AK Protocols

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 [13]). Although there are some provably secure protocols in the literature, their proofs of security are often complicated and error-prone [13]. In this regard, Kudla and Paterson

[21, 23] developed an elegant modular technique for constructing security proofs for a large class of key agreement protocols. The overall proof technique using the modular approach is far easier than the conventional approach. In a nutshell, 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 [27].

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 $\Pi^{i'}_{J,I}$ 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 [21], 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 improved protocol.

Reduced Games. A highly reduced game (called the *cNR-ID-mBJM* game) is used in the modular security proof. The reduced game is identical to 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 *compute* 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 [21]). *A protocol 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;*
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 defined 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 [21]). *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 Revisiting the Ryu, Yoon and Yoo Key Agreement Protocol

In this section, we revisit the ID-based key agreement protocols due to Ryu *et al.* (the RYY protocol) [29]. The protocol involves three entities: two users who wish to establish a shared secret session key (*e.g.* Alice and Bob), and a trusted PKG from whom they each acquire their private keys. There are two stages in the protocol, namely: the *System Setup* stage and the *Authenticated Key Agreement* stage.

System Setup. Suppose we have an admissible pairing $\hat{e} : \mathbb{G}_1 \times \mathbb{G}_1 \to \mathbb{G}_2$ as described in the preceding section, 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 $H_1 : \{0,1\}^* \to \mathbb{G}_1$;
3. publishes the system parameters $params = \langle \mathbb{G}_1, \mathbb{G}_2, \hat{e}, P, H_1 \rangle$;
4. computes the private key $S_{ID} = sQ_{ID}$ 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 S_{ID} to the user with the identity information ID via a secure channel.

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

Authenticated Key Agreement. We denote users Alice's and Bob's public/private key pairs as (Q_A, S_A) and (Q_B, S_B), respectively. To establish a shared session key, Alice and Bob generate an ephemeral private key (say a and $b \in \mathbb{Z}_q^*$) independently, and compute the corresponding ephemeral public keys $T_A = aP$ and $T_B = bP$. They then exchange T_A and T_B as described in Fig. 1.

After the message exchange, the two users do the following:

1. Alice computes the shared session key sk_{AB} as follows (after receiving T_B):

$$sk_{AB} = H(A||B||aT_B||K_{AB}),$$

   in which $K_{AB} = \hat{e}(S_A, Q_B)$ and H is a predetermined key derivation function of the two users.

Alice		**Bob**
$a \in_R \mathbb{Z}_q^*$		$b \in_R \mathbb{Z}_q^*$
$T_A = aP$		$T_B = bP$
	$\xrightarrow{T_A}$	
	$\xleftarrow{T_B}$	
$K_{AB} = \hat{e}(S_A, Q_B)$		$K_{BA} = \hat{e}(S_B, Q_A)$
$sk_{AB} = H(A\|\|B\|\|aT_B\|\|K_{AB})$		$sk_{BA} = H(A\|\|B\|\|bT_A\|\|K_{BA})$

Fig. 1. The RYY Protocol

2. Bob computes the shared session key sk_{BA} as follows (after receiving T_A):

$$sk_{BA} = H(A||B||bT_A||K_{BA}),$$

in which $K_{BA} = \hat{e}(S_B, Q_A)$.

Protocol Correctness. It is easy to see $aT_B = bT_A = abP$. And, by the bilinearity of the pairing, we can easily get the following equation:

$$\begin{aligned} K_{AB} &= \hat{e}(S_A, Q_B) \\ &= \hat{e}(sQ_A, Q_B) \\ &= \hat{e}(Q_A, Q_B)^s \\ &= \hat{e}(Q_A, S_B) \\ &= K_{BA}. \end{aligned}$$

Thus, the two session keys computed by Alice and Bob are equal to each other. Put in other words, the two users successfully established a shared session key after running an instance of the protocol.

4 Our Attack and an Improved Protocol

4.1 Attack on the RYY Protocol

Ryu *et al.* showed that their protocol is more efficient than all previously known protocols and at the same time enjoys all the security properties [29]. Contrary to their claim, it has been shown that their protocol is not secure against the K-CI attack [4, 37]. In this section, we will show that their protocol is vulnerable to (yet) another attack – the *reflection attack*.

We remark that in order for the reflection attack to work, the protocol must allow concurrent sessions to be executed. The latter is a reasonable assumption as shown by several researchers (see [11, 20]). For example, as noted by Boyd and Mathuria [9], if one party is an Internet host, it may accept concurrent sessions from multiple principals while using the same identity and set of cryptographic keys. In other words, a party (with an unique identity) may initiate concurrent sessions with an intended communicating partner (with another unique identity).

A typical reflection attack scenario is where two protocol principals, A and B, engage in a statically shared secret key protocol and the adversary simply replays a challenge that is intended for herself. Ryu *et al.*'s protocol uses the ID-based non-interactive statically shared secret key $K_{AB} = K_{BA}=\hat{e}(Q_A, Q_B)^s$ of the two parities, which was first suggested by Sakai *et al.* [34] in 2000. Although this technique of authentication is computationally efficiency, protocols adopting such an authentication mechanism may not resist the reflection attack. Consider the following scenario. Suppose that Alice initiates two parallel sessions of the protocol with Bob; let Alice's ephemeral public keys be $T_A = aP$ and $T'_A = a'P$ in the first (S1) and second (S2) sessions, respectively. The attack is described in Fig. 2.

Alice		**Eve** (Bob)
$a \in_R \mathbb{Z}_q^*$ $T_A = aP$		Eve masquerades as Bob.
	S1. $\xrightarrow{T_A}$	
$a' \in_R \mathbb{Z}_q^*$		
$T'_A = a'P$	S2. $\xrightarrow{T'_A}$	
		Eve replays T'_A in session 1.
	$\xleftarrow{T'_A}$ S1.	
		Eve replays T_A in session 2.
	$\xleftarrow{T_A}$ S2.	
$sk_{AB} = H(A\|\|B\|\|aa'P\|\|K_{AB})$ $sk'_{AB} = H(A\|\|B\|\|aa'P\|\|K_{AB})$		Eve terminates.

Fig. 2. A Reflection Attack on the RYY Protocol

After intercepting T_A and T'_A, the adversary (Eve) replays T'_A and T_A (notice that the order is in reverse) to Alice in the first and second sessions respectively, purportedly as Bob's ephemeral public keys. For both the two sessions, Alice is the protocol *initiator*, so she computes both session keys as $sk = H(A||B||aa'P||\hat{e}(S_A,\ Q_B))$. In fact, all the cryptographic operations have been performed by Alice herself, while she believes that the two protocol sessions have been completed with Bob. However, Bob may be completely unaware of the attack. The refection attack makes two different sessions of the protocol result in a same session key, thus breaks the basic security attribute of known-key secrecy. That is to say, the compromise of a session will lead to the compromise of another.

4.2 An Improved Protocol

Recent work of Choo, Boyd and Hitchcock [13, 12] suggests that the inclusion of the transcript T, *i.e.*, the concatenation of all messages sent and received, in the key derivation function H effectively binds the session key to all messages sent and received by both protocol principals.

We note that the inclusion of the protocol transcript in the key derivation function alone for the RYY protocol may *not* be effective in countering the reflection attack described in Fig. 2. For example, in the case of Alice acting in different roles (initiator or responder) during the parallel sessions, the two protocol transcripts in our attack are still equal, hence resulting in the same session keys. In other words, our reflection attack still works. To counter the reflection attack, we would have to include the identities of the participants in the order of their roles (*e.g.* identity of initiator concatenates identity of responder). We remark that the

Alice		**Bob**
$K_{AB} = \hat{e}(S_A, Q_B)$		$K_{BA} = \hat{e}(S_B, Q_A)$
$a \in_R \mathbb{Z}_q^*$		$b \in_R \mathbb{Z}_q^*$
$T_A = aP$		$T_B = bP$
	$\xrightarrow{T_A}$	
	$\xleftarrow{T_B}$	
$sk_{AB} = H(A\|\|B\|\|aT_B\|\|K_{AB}\|\|T_A\|\|T_B)$		
		$sk_{BA} = H(A\|\|B\|\|bT_A\|\|K_{BA}\|\|T_A\|\|T_B)$

Fig. 3. The RYY$^+$ Protocol

inclusion of the identities of the participants and their roles in the key derivation function is generally regarded as a common strategy to provide resilience against unknown key-share attacks (see [13]).

The RYY$^+$ Protocol. To counter the reflection attack described in Fig. 2, we modified the protocol described in Fig. 1. The improved protocol is described in Fig. 3. According to the above analysis, the modified key generation (for Alice) of Ryu *et al.*'s protocol is as follows, with the underlined value indicates the necessary input to withstand the reflection attack. We now have

$$sk = H(\underline{A||B}||aT_B||K_{AB}||\underline{T}),$$

in which the protocol transcript is computed as $T = T_A||T_B$.

In the next section, we strictly prove that the above improved RYY$^+$ protocol is secure in the ID-mBJM model, assuming the hardness of the GBDH problem. We also prove that the protocol achieves PKG forward secrecy, assuming the hardness of the CDH problem for $\mathbb{G}_1$[4].

5 Security Proof*s*

We prove the security (*i.e.* ID-mBJM security plus PKG-PFS) of our improved RYY$^+$ protocol in two stages. We first prove that it is ID-mBJM secure using the Kudla–Paterson modular technique [23, 21] (Theorem 2), then we prove that it also provides the additional attribute of PKG-FS (Theorem 3).

5.1 Proof of Basic Security Attributes

With the description of the ID-mBJM model in Section 2.3, we now state:

Theorem 2 (ID-mBJM Security of RYY$^+$). *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 the RYY$^+$ protocol is a secure key agreement protocol.*

[4] The CDH problem for $\mathbb{G}_1$ is, given $x_1P, x_2P \in \mathbb{G}_1$ for unknown $x_1, x_2 \in \mathbb{Z}_q^*$, to compute x_1x_2P, where P is a generator of $\mathbb{G}_1$.

We now prove Theorem 2 in three steps. Firstly, we show that protocol has strong partnering. Secondly, we prove that the related protocol π of RYY$^+$ is secure in the cNR-ID-mBJM model (see Definition 10). Lastly, we show that the session string decisional problem (see Definition 11) of RYY$^+$ is reducible to the DBDH problem.

Lemma 1 (Strong Partnering of RYY$^+$). *The RYY$^+$ protocol 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 identities 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. 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 two random integers u, 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 u-th distinct H_1 query, then the oracle outputs $Q_i = yP$; If ID_i is the v-th distinct H_1 query, then the oracle outputs $Q_i = zP$. S 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 I (resp. J) is the u-th (resp. v-th) distinct participant created in the game. Therefore, we have $Q_I = yP$ and $Q_J = zP$.

$\mathsf{Corrupt}(ID_i)$**:** S simulates the $\mathsf{Corrupt}$ query on input ID_i as follows.

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

Send(A, B, t, M)**:** To answers the queries, S randomly samples $\xi_t \in \mathbb{Z}_q^*$ and responds with $\xi_t P$.

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 2**).

Output: At the end of the game, the algorithm E outputs a session key of the form $(U, V, K_1, K_2, T_1, T_2)$ where $U, V \in \{0, 1\}^*$, $K_1, T_1, T_2 \in \mathbb{G}_1$ and $K_2 \in \mathbb{G}_2$.

Solving the BDH Problem: S outputs K_2 as its guess for the value $\hat{e}(P, P)^{xyz}$.

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 is an instance of participant I and it supposes to establish a session key with participant J, then by the rules of the game **Event 1** and **2** would not happen. We have

$$\Pr[S \text{ does not abort}] \geq \frac{1}{q_S q_1^2}.$$

Note that the (unknown) master-key is x and participant I (resp. J) has the public key $Q_I = yP$ (resp. $Q_J = zP$). For any oracle $\Pi^n_{I,J}$, part of the agreed secret (namely K_2) is $\hat{e}(yP, zP)^x$. So if the adversary computes the correct session key with non-negligible probability ϵ, then S answers the BDH problem correctly with probability with $\epsilon/(q_S q_1^2)$ (which is non-negligible in the security parameter l), contradicting to the hardness of the BDH problem. □

Lemma 3 (Session String Decisional Problem of RYY$^+$). *The session string decisional problem of the RYY$^+$ protocol is reducible to the* DBDH *problem.*

Proof. Recall the session string of the RYY$^+$ protocol is of the form $(A, B, K_1, K_2, T_A, T_B)$ with $K_2 = \hat{e}(Q_A, Q_B)^s$, s being the master-key and P, sP being the public parameters, then we see $\langle P, sP, Q_A, Q_B, K_2\rangle$ is a BDH tuple. Besides, since $T_A = aP$, $T_B = bP$ and $K_1 = abP$, $\langle P, T_A, T_B, K_1\rangle$ forms a DDH tuple. Note that because of the existence of pairings, the DDH problem in $\mathbb{G}_1$ is not hard. This implies that the session string decisional problem of the RYY$^+$ protocol 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. □

5.2 Proof of PKG Forward Secrecy

As mentioned above, the improved RYY$^+$ protocol also enjoys an extra security attribute for ID-based key agreement protocols, *i.e.* PKG forward secrecy (or equivalently, no session key escrow). Key escrow is an inherent problem of ID-based cryptography, namely the PKG knows all the long-term private keys. One advantage of a PKG forward secure ID-based key agreement protocol is that the users can generate a fresh session key that the PKG *cannot* compute unless it launches active (man-in-the-middle) attacks. In this regard, PKG forward secrecy is a big advantage for ID-based key protocols in practice [22].

Theorem 3 (PKG-FS of RYY$^+$). *The RYY$^+$ protocol has the property of PKG forward secrecy (PKG-PFS), assuming the* CDH *problem for group $\mathbb{G}_1$ is hard and provided that H and H_1 are random oracles.*

Proof. According to our definition of PKG 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 CDH problem in $\mathbb{G}_1$ with non-negligible probability. Given group $\mathbb{G}_1$, a generator P of $\mathbb{G}_1$, and two elements xP, $yP \in \mathbb{G}_1$ with $x, y \in \mathbb{Z}_q^*$ where q is the prime order of $\mathbb{G}_1$, its task is to compute and output the value xyP.

The algorithm S selects two random integers n_1, n_2 from $\{1, ..., q_S\}$ (assuming $n_1 < n_2$) and works by interacting with E as follows:

Setup: S first picks random a master-key $s \in \mathbb{Z}_q^*$ and sets the PKG's master key to be sP. E is given the master-key s and she will no longer need to make Corrupt queries since E can now compute all private keys for herself. 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.

$H(ID_i, ID_j, K_1, K_2, T_i, T_j)$: S simulates the random oracle H by keeping an H-List with tuples of the form $\langle ID_i, ID_j, K_1, K_2, T_i, T_j, k_i\rangle$. 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.

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

– When $t = n_1$, if oracle $M \neq \lambda$, then abort (**Event 1**), otherwise return xP.
– When $t = n_2$, if $M \neq xP$, then abort (**Event 2**), otherwise return yP.
– When $t \neq n_1, n_2$, randomly sample $\xi_t \in \mathbb{Z}_q^*$, return $\xi_t P$.

Reveal$(\Pi^t_{A,B})$: Upon receiving a Reveal query, S outputs the appropriate session key, except if E asks the oracle $\Pi^{n_1}_{A,B}$ or $\Pi^{n_2}_{A,B}$, 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 $\Pi^{n_1}_{A,B}$ or $\Pi^{n_2}_{A,B}$ 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 CDH Problem: S picks a tuple of the form $\langle I, J, K_1, K_2, T_I, T_J\rangle$ (for some h) from the H-List and returns K_1 as the response to the CDH 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 n_1-th and n_2-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 $\Pi^{n_1}_{I,J}$ must have obtained the value $T_J = yP$ from its partner oracle $\Pi^{n_2}_{J,I}$. The oracle should hold a session key of the form $H(I, J, K_1, K_2, T_I, T_J)$, in which $K_1 = xyP$.

Let $\mathtt{Q}$ be the event that the session string of the test oracle has been queried to H. Because H is a random oracle, we have $\Pr[E \text{ wins}|\bar{\mathtt{Q}}] = 1/2$. Then

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

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

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

Combining all the above results, we have that S solves the CDH 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 CDH problem. □

6 Comparison of Protocols

Here we summarize the security properties and performances of the proposed RYY$^+$ protocol and several other previously published provably-secure two-pass ID-based key agreement protocols in Table 1. Note that:

- Since all listed protocols offer the basic security properties (*i.e.*, known-key secrecy, unknown key-share resilience, and no key control), we will restrict our comparison to only the forward secrecy and key-compromise impersonation resilience.
- 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.

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

↓Protocols / Items→	$\mathbb{P}$	$\mathbb{M}$	$\mathbb{E}$	$\mathbb{A}$	Bandwidth	K-CIR	PKG-FS
Chen–Kudla [17]	1	0	0	1	1 point	✓	×
Smart's [31]	1	0	0	0	1 point	✓	×
Wang's [36]	1	1	0	2	1 point	✓	×
Chow–Choo [15]	1	1	0	2	1 point	✓	×
RYY$^+$	0	1	0	0	1 point	×	✓

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 $\mathbb{G}_1$.

From the table, we observe that only the RYY$^+$ protocol eliminates on-line pairing evaluation. Since pairing evaluation is far more computationally expensive that other operations, the RYY$^+$ protocol is the most efficient one among all the listed protocols, especially when we take into consideration that certain computations can be performed off-line.

Remark 2. Similar to the UMP protocol [1], our proposed RYY$^+$ protocol does not provide K-CI resilience. However, since it is provably-secure in a model which captures all the *basic* security attributes, it is still of great practicality, especially for its excellent performances.

Moreover, despite its lack of K-CI resilience, only the RYY$^+$ protocol achieves PKG forward secrecy (PKG-FS). As we noted in Section 5.2, PKG-FS is a big advantage for ID-based key agreement protocols in practice. Although all the other protocols listed in Table 1 can also be extended to achieve PKG forward secrecy using the simple idea due to Chen and Kudla [17] by embedding a raw Diffie–Hellman protocol [18]. However, this will introduce extra computational operations (or even worse, this will introduce both extra protocol tokens and extra computational operations [17, 36, 15]), hence degrades the performances of these protocols (interested readers are referred to [17] for the details).

7 Conclusion

We further studied the security of an efficient identity-based key agreement protocol from pairings due to Ryu, Yoon and Yoo. We revealed previously unpublished attack on the protocol and proposed an improved protocol that is proven secure with respect to all the basic security attributes (*i.e.* known-key secrecy, unknown key-share resilience and no key control) *and* PKG forward secrecy under the standard Gap Bilinear Diffie-Hellman assumption in the random oracle model. The improved protocol preserves excellent efficiency (in particular, it eliminates the on-line pairing evaluation) and it is suitable for low-power computing devices.

Quite recently, we noticed that Menezes and Ustaoglu [26] studied the *stronger* security of the UMP protocol. As observed in [35], there are close relations between an authenticated Diffie-Hellman protocol and its ID-based counterpart. Naturally, a future work is to further strengthen the RYY$^+$ to achieve stronger security, for example as in [26], to investigate its security in the extended Canetti–Krawczyk (eCK) model [24].

References

1. R. Ankney, D. Johnson, M. Matyas, The Unified Model, contribution to X9F1, October 1995. [1, 4, 16]
2. ANSI X9.42, Agreement of Symmetric Algorithm Keys Using Diffie–Hellman,working draft, May 1998. [4]
3. ANSI X9.63, Elliptic Curve Key Agreement and Key Transport Protocols, working draft, July 1998. [4]
4. C. Boyd, K.-K.R. Choo, Security of two-party identity-based key agreement. in: Proc. Mycrypt'05, LNCS vol. 3715, 2005, pp. 229-243. [2, 9]
5. D. Boneh, M. Franklin, Identity-based encryption from the Weil pairing. in: Proc. CRYPTO'01, LNCS vol. 2139, 2001, pp. 213-229. [2, 3]
6. S. Blake-Wilson, C. Johnson, A. Menezes, Key agreement protocols and their security analysis, in: Proc. the Sixth IMA International Conference on Cryptography and Coding, LNCS vol. 1355, 1997, pp. 30-45. [4]
7. P.S.L.M. Barreto, H.Y. Kim and B. Lynn. Efficient algorithms for pairing-based cryptosystems. In *Proc. CRYPTO 2002*, LNCS vol. 2442, 2002, pp. 354-368. [3]

8. S. Blake-Wilson, A. Menezes, Authenticated Diffie-Hellman key agreement protocols, in: Proc. SAC'98, LNCS vol. 1556, 1999, pp. 339-361. [3]
9. C. Boyd, A. Mathuria, Protocols for Authentication and Key Establishment. Springer-Verlag, June 2003. [2, 3, 9]
10. M. Bellare, P. Rogaway, Entity authentication and key distribution, in Proc. CRYPTO'93, LNCS vol. 773, 1993, pp. 110-125. [2]
11. K.-K.R. Choo, Key Establishment: Proofs and Refutations. Ph.D. Thesis, Queensland University of Technology, 2006. [9]
12. K.-K.R. Choo, On the security analysis of Lee, Hwang & Lee (2004) and Song & Kim (2000) key exchange agreement protocols, *Informatica*, 17(4), 2006, 467-480. [10]
13. K.-K.R., Choo, C. Boyd, Y. Hitchcock, On session key construction in provably secure protocols, in: Proc. MYCRYPT'05, LNCS vol. 3715, 2005, pp. 116–131. [6, 10, 11]
14. K.-K.R. Choo, C. Boyd, Y. Hitchcock, G. Maitland, On session identifiers in provably secure protocols: The Bellare-Rogaway three-party key distribution protocol revisited, in: Proc. SCN'04, LNCS vol. 3352, 2005, pp. 351-366. [4]
15. S. S.-M. Chow, K.-K.R. Choo, Strongly-secure identity-based key agreement and anonymous extension, in: Proc. ISC'07, LNCS vol. 4779, 2005, pp. 203-220. [15, 16]
16. R. Canetti, Hugo Krawczyk, Analysis of key-exchange protocols and their use for building secure channels, in: Proc. EUROCRYPT'01, LNCS vol. 2045, 2001, pp. 453-474. [4]
17. L. Chen, C. Kudla, Identity based key agreement protocols from pairings, in: Proc. the 16^{th} IEEE Computer Security Foundations Workshop, 2002, pp. 219-213. [2, 3, 15, 16]
18. W. Diffie, M.E. Hellman, New directions in cryptography, IEEE Trans. Inf. Theory, 22(6), (1976) 644 - 654. [1, 16]
19. S.D. Galbraith, K. Harrison, D. Soldera, Implementing the Tate pairing, in: Proc. ANTS-V, LNCS vol. 2369, 2002, pp.324-337. [3]
20. H. Krawczyk, HMQV: A high performance secure Diffie-Hellman protocol, in: Proc. Crypto'05, LNCS vol. 3621, 2005, pp. 546-566. [6, 9]
21. C. Kudla, Special signature schemes and key agreement protocols. Ph.D. Thesis, Royal Holloway University of London, 2006. [4, 7, 8, 11]
22. C. Kudla, Private communications, May 2006. [13]
23. C. Kudla, K. G. Paterson, Modular security proofs for key agreement protocols, in: Proc. ASIACRYPT'05, LNCS vol. 3788, 2005, pp. 549-565. [2, 7, 11]
24. B. LaMacchia, K. Lauter and A. Mityagin, Stronger Security of Authenticated Key Exchange, Cryptology ePrint Archive: Report 2006/073. [16]
25. A. Menezes, P. van Oorschot, S. Vanstone, Handbook of Applied Cryptography, CRC Press, 1997, pp. 237-238. [3]
26. A. Menezes, B. Ustaoglu, Security arguments for the UM key agreement protocol in the NIST SP 800-56A standard, to appear in Proc. ACM ASIACCS'08. www.cacr.math.uwaterloo.ca/ ajmeneze/publications/um.pdf. [16]
27. T. Okamoto, D. Pointcheval, The Gap-problems: a new class of problems for the security of cryptographic schemes, in: Proc. PKC'01, LNCS vol. 1992, 2001, pp. 104-118. [3, 7]
28. IEEE P1363, Standard Specifications for Public-Key Cryptography, working draft, July 1998. [4]
29. E. Ryu, E. Yoon, K. Yoo, An efficient ID-based authenticated key agreement protocol from pairings, in: Proc. NETWORKING'04, LNCS vol. 3042, 2004, pp. 1458-1463. [2, 8, 9]
30. A. Shamir, Identity-based cryptosystems and signature schemes, in: Proc. CRYPTO'84, LNCS vol. 196, 1984, pp. 47-53. [2]
31. N. Smart, An ID-based authenticated key agreement protocol based on the Weil pairing, Electron. Lett., 38(13), (2002) 630-632. [2, 15]
32. K. Shim, Efficient ID-based authenticated key agreement protocol based on Weil pairing, Electron. Lett., 39(8), (2003) 653-654. [2]
33. H. Sun, B. Hsieh, Security analysis of Shim's authenticated key agreement protocols from pairings, Cryptology ePrint Archive 2003/113. http://eprint.iacr.org/2003/113/. [2]
34. R. Sakai, K. Ohgishi, M. Kasahara, Cryptosystems based on pairing, in Proc. SCIS'00, 2005, pp. 1-8. [10]

35. S. Wang, On the relations between authenticated Diffie-Hellman and ID-based authenticated key agreement — A parallel design and analysis methodology, Manuscript, 2007. http://tdt.sjtu.edu.cn/ swang/paper/ID-MQV.pdf [1, 16]
36. Y. Wang, Efficient identity-based and authenticated key agreement protocol, Cryptology ePrint Archive 2005/108. http://eprint.iacr.org/2005/108/. [15, 16]
37. S. Wang, Z. Cao, H. Bao, Security of an efficient ID-based authenticated key agreement protocol from pairings, in: Proc. ISPA'05 Workshops, LNCS vol. 3759, pp. 342-349. [2, 9]