Key Agreement from Signatures: Improved Protocols and Anonymous Extension

Sherman S.M. Chow[1] and Kim-Kwang Raymond Choo[2]

[1] Department of Computer Science
Courant Institute of Mathematical Sciences
New York University, NY 10012, USA
http://www.cs.nyu.edu/~schow
schow@cs.nyu.edu
[2] Independent Security Researcher
Canberra, Australia
http://raymond.choo.au.googlepages.com
raymond.choo.au@gmail.com

Abstract. We exploit the relationships between signature schemes and key agreement protocols; and propose a high performance identity-based (ID-based) key agreement protocol based on *strong* pairing challenge-response signatures. The latter is the first of its kind in ID-based cryptography and is of interest in itself. Using the proof technique of signature unforgeability against adaptive chosen-message attack, our protocol *fully* supports Session-Key Reveal queries and partially supports Session-State Reveal queries (which leaks ephemeral secret and keying material for session key derivation), without gap assumption or any unrealistic restriction. We show how to incorporate *KGC forward secrecy* so the past session keys are not compromised even the adversary gets the master secret key of the Key Generation Center (and the private keys of all users). Both proposals are efficient and have the strongest security among other unbroken identity-based two-party two-message protocols. Inspired by ring signatures and motivated by the need for a better anonymous roaming mechanism, we extend our basic protocol to support key agreement among spontaneous anonymous groups (SAG). To the best of our knowledge, this is the first ID-based SAG key agreement protocol with *bilateral* privacy.

Key words: Key agreement, provable security, reveal query, identity-based cryptography, anonymity

1 Introduction

To establish secure communications over insecure network, we need to establish a common secret key shared among the communicating parties. The establishment of session keys often involves key establishment protocols, which are the cornerstone for any security functions.

In Identity-based (ID-based) cryptosystem, a trusted key generator center (KGC) generates user's private key on demand when given an identity. Many ID-based key establishment protocols have emerged over the years. Some of them may offer an advantage in ID-based paradigm: to escape from the key-escrow of KGC. This point will become clear when we discuss escrow-free protocol.

1: Security Issues in Key Agreement Protocols – Reveal ***Queries.*** A recent survey [5] observes that the purported security of many published ID-based two-party protocols (that are not later found to be flawed) are proven in a weak variant of Bellare–Rogaway (BR) model [2] in which adversary cannot make Session-Key Reveal queries (hereafter referred to as the BR (NR) model). The Session-Key Reveal queries allow the adversary to learn previously accepted session keys. Supporting this kind of queries ensures that session key generated in each protocol round is

independent of another, and hence the compromise of any other session key(s) should not affect the security of a non-related session.

In many protocols, for efficiency or some other reason, a user may decide to perform some pre-computations and store the results. The types of values stored in a session state include:

1. the ephemeral Diffie-Hellman (DH) key (the discrete logarithm of the outgoing DH value);
2. the outgoing and incoming DH values; and
3. the keying material for session key, if a key derivation oracle is employed.

One drawback is that these parameters may not be protected as securely as the long-term private key. Consequently, this may be a potential vulnerability that can be exploited by the adversary. The Session-State Reveal query captures this security threat.

2: Privacy Issues in Key Agreement Protocols. Anonymity is required in many cryptographic applications to ensure that information about the user is not revealed. This concept is also useful and applicable to key agreement protocol. Suppose two entities, $\mathcal{U}$ and $\mathcal{V}$, want to exchange messages in a confidential manner. In anonymous key agreement protocols such as the protocols of Boyd and Park [6] and of Shoup [29], $\mathcal{U}$'s identity is protected from being known to any adversary (or network eavesdropper), but not $\mathcal{V}$ – the authentication part of key agreement protocol.

In this work, we consider anonymity on a higher level: $\mathcal{V}$ knows that $\mathcal{U}$ is a member of a group of entities, but is unable to confirm the actual identity of $\mathcal{U}$. This class of protocol is useful when $\mathcal{V}$ only needs to ensure the membership of the group for the user, but not necessarily the identities of users, who are concerned with their privacy.

Concepts in Signatures Help Key Agreement Protocols. We see many applications of digital signatures in higher-level cryptographic protocols. In the digital signature paradigm, unforgeability against adaptive chosen-message attack is the standard requirement – there exists a signing oracle giving arbitrary signatures of the adversary's choice. We observe that this concept is somewhat analogous to Session-Key Reveal queries in the key agreement paradigm that returns session keys of the adversary's choice when asked. Ephemeral parameters are often involved in the computation of signatures, which are also inherent in key agreement protocol. It is rather tempting to base the design of key agreement protocol on digital signature schemes. We also note that there are many well-established signature schemes with anonymity concerns. For example, in ring signature, formalized by Rivest, Shamir, and Tauman [24, 25], absolute anonymity is provided whereby a signer can sign a message on behalf of a spontaneous group of possible signers without exposing the identity of the actual signer. It is natural to ask, "*Can we borrow concepts from digital signatures to construct better key agreement protocols?*" This work provides an affirmative answer.

Roadmap. The next section discusses some related work and highlight our contributions. Section 3 introduces the computational assumptions used in this paper. Section 4 discusses our *strong* challenge-response signature scheme. Our proposal of ID-based key agreement protocol is presented in Section 5. We also discuss how to provide *KGC forward secrecy*, and provide a comprehensive comparison on the performance and security of our proposed protocols relative to other published ID-based two-message two-party protocols. We then extend our basic protocol presented in Section 5 to SAG setting in Section 6, followed by conclusion in Section 7. We present a brief overview of the Canetti–Krawczyk model and the mathematical preliminaries in Appendix A. The security notion and the corresponding proof for the challenge-response signature schemes can be found in Appendix B. Appendix C gives a previously unpublished attack on Yi's protocol [34].

2 Related Work and Our Contributions

2.1 Reveal Queries

Recent research efforts have been devoted towards designing protocols that can be proven secure in a model that allows the Session-Key Reveal queries. For example, Choo, Boyd and Hitchcock [14] improve the ID-based protocols of Chen and Kudla [8] and of McCullagh and Barreto [21] so that these protocols are able to be proven secure when the adversary is allowed to ask Session-Key Reveal queries in most cases, while the technicality of not being able to answer reveal queries in some special sessions can be resolved using gap assumption. Gap assumption, first proposed by Okamoto and Pointcheval [22], states that the underlying computational problem is intractable even with the help of a corresponding decisional oracle. Such oracle may or may not exist.

With gap assumption, Kudla and Paterson [19] propose a generic transformation that is able to turn any protocols that are proven secure in the BR (NR) model to a protocol that can be proven secure in the full BR model. Along somewhat similar line, Wang [32] proposes a protocol with security based on a decisional problem by using a *computational oracle* to support the Session-Key Reveal queries. Although these two works support Session-Key Reveal queries, special oracle is assumed by the simulation in the proof. Finally, we note that Cheng *et al.* [11] introduce the concept of coin queries that forces the adversary to reveal its ephemeral secret, and thus making Session-Key Reveal possible. Their approach is restricted in the sense that the possibility of an adversary breaking a protocol without knowing the ephemeral secret is not modelled.

The Session-State Reveal query, only accessible to an adversary in the Canetti–Krawczyk model (hereafter referred to as the CK model) [7], allows the adversary to learn ephemeral parameters associated with a particular session. An example of a protocol proven secure in this stronger model[1] is the HMQV protocol [18]. HMQV protocol is a "hashed" variant of a MQV protocol. Recently, MQV protocol's security is analyzed [20]. However, Session-State Reveal query is not considered. The HMQV protocol remains secure even when the ephemeral DH key is accessible by the attacker, under gap Diffie-Hellman assumption. Note that no security claim is made regarding the availability of the keying material for derivation of the session key.

Exponential challenge-response (XCR) signature scheme is introduced by Krawczyk [18] as a building block for key agreement protocol. Such a scheme, an inherently interactive public key signature protocol, requires the verifier to issue a challenge to the signer and the later responses by a signature on a given message. Apart from being unforgeable – which ensures that only the legitimate signer can generate a signature that will make the challenger convinced – challenge-response signature is challenge-specific. A distinctive feature of Krawczyk's XCR scheme is that any party who possesses the challenge can always generate the *same* signature string as the signer. This property of XCR, essentially, makes it the building block for key agreement protocol.

Contribution 1 – Strong Challenge-Response Signatures. As noted by Krawczyk [18], no assumptions regarding third party's transferability or verifiability of a XCR signature is made. In this work, we observe that the verifiability is easily transferable. In view of this, we propose a *strong* challenge-response signature scheme in which the verification requires the knowledge of the designated verifier's private key. To the best of our knowledge, our scheme is the *first* ID-based challenge-response signature scheme.

[1] The relative strengths between the BR and CK models are discussed in [13].

Contribution 2 – High-Performance ID-based Key Agreement Protocols. We propose an efficient ID-based key agreement protocol. In a similar fashion as the design of the HMQV protocol, we base our design on ID-based challenge-response signatures. We give formal security assurance of the protocol in the stronger CK model, which allows the adversary to ask Session-Key Reveal queries in *all* cases, and Session-State Reveal queries in most cases, *without* employing any gap assumption. We show how to provide *KGC forward secrecy* with small modifications. In contrast to HMQV protocol, our definition of session state also includes the ephemeral DH key of the outgoing DH values and the keying material for key derivation. Among the list of ID-based two-party protocols surveyed by [5], our protocols are efficient and achieve the strongest security properties.

2.2 Anonymous Key Agreement Protocols

To illustrate the benefit brought by our key agreement protocols with anonymity concern, consider the scenario where delegates would like to make/receive phone calls on their mobile phones by roaming into a foreign country's telecommunication network. Before secure roaming is established, the service provider must verify whether one is a legitimate subscriber with a certain home server.

Conventional anonymous roaming mechanisms [1, 26] are rather inefficient as users, usually, would have to wait online while foreign telecommunication network communicates with the original home server (which also generates extra network traffic) do the authentication. At the same time, it is inconvenient to renew constantly the alias in an unlinkable manner to hide the identities – again as required in existing systems.

Our proposed approach comes in handy in this case since our approach (1) would allow user to "hide" among the subscribers of the home server and (2) after the home server has issued matching sets of public/private key pair at the very beginning, it is no longer required to be online. The savvy reader may note that our approach does not appear to be *scalable* if one needs to hide among all (a potentially large set of) legitimate subscribers, and may not be *flexible* since it is natural that the set of subscribers is constantly changing. However, both issues can be readily solved by a solution without an *a priori* group formation step and this is where the idea of *spontaneous anonymous group* (SAG) comes into the picture. Any legitimate user can spontaneously conscripts an arbitrary group of users (i.e., without any cooperation of the other parties in the group) for each session. SAG formation also empowers the user's full control on the level of anonymity desired during the secure roaming establishment process, without using alias.

Contribution 3 – Key Agreement Protocol among Highly Spontaneous Anonymous Groups with Bilateral Privacy. Motivated by the many possible applications of anonymous roaming and our observation that existing research (e.g., [10]) appears to focus only on *unilateral* identity privacy; we take a step further by devising secure key exchange *among* spontaneous anonymous *groups*. This is applicable in both anonymous roaming and situation like an ad hoc group communication setting. Furthermore, as noted in SAG cryptography's literatures [15, 25], ID-based solution is better than its counterpart in the sense that it removes the costs in dealing with a large number of public keys, and provides a higher level of spontaneity than conventional public key cryptosystem since one can conscript virtually anyone even those without public key certificates. With these motivations in mind, we introduce the notion of ID-based key agreement protocols among SAGs with *bilateral* privacy, which is realized by an extension of our basic protocol.

3 Number Theoretic Assumptions

Let $\mathbb{G}$ be an additive group of prime order q and $\mathbb{G}_T$ be a multiplicative group also of order q. We assume the existence of an efficiently computable bilinear map $\hat{e} : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$ such that

1. There is an known element $P \in \mathbb{G}$ satisfying $\hat{e}(P, P) \neq 1_{\mathbb{G}_T}$.
2. For $Q, W, Z \in \mathbb{G}$, both $\hat{e}(Q, W + Z) = \hat{e}(Q, W) \cdot \hat{e}(Q, Z)$ and $\hat{e}(Q + W, Z) = \hat{e}(Q, Z) \cdot \hat{e}(W, Z)$.

Typically, $\mathbb{G}$ will be a subgroup of the group of points on an elliptic curve over a finite field, $\mathbb{G}_T$ will be a subgroup of the multiplicative group of a related finite field. The map $\hat{e}$ can be derived from the Weil or Tate pairing on the elliptic curve.

Definition 1 (Interactive Game with a BDH Challenger [10]). *Let $\mathcal{A}$ be a pair of probabilistic polynomial-time (PPT) algorithms $(\mathcal{A}_1(r_1; \ldots), \mathcal{A}_2(r_2; \ldots))$, where r_i is used by $\mathcal{A}_i$ as the random tape, that engages with a challenger in the following game. Let (P, aP, bP, cP) be the BDH instance where $P, aP, bP, cP \in \mathbb{G}$ and $a, b, c \in \mathbb{Z}_q^*$. The game is defined as follows.*

Stage 1: $(X, \sigma) \leftarrow \mathcal{A}_1(r_1; P, aP, bP, cP, \hat{e}, \mathbb{G}, \mathbb{G}_T, q)$, *where σ is the state information.*
Interactive Part: *After seeing X, challenger returns $h \leftarrow \mathbb{Z}_q^*$.*
Stage 2: $K \leftarrow \mathcal{A}_2(r_2; h, \sigma)$.

We say that the adversary, $\mathcal{A}$, wins the game if it computes $K = \hat{e}(aP, X + hbP)^c$.

If X is determined *after* seeing h, the problem is easy since one can set $X = rP - hbP$ for $r \in_R \mathbb{Z}_q^*$, and returns $K = \hat{e}(aP, cP)^r$. It explains the game's interactive nature.

Lemma 1 (Interactive BDH Game Assumption [10]) *For any adversary with PPT algorithm $(\mathcal{A}_1, \mathcal{A}_2)$ with advantage $\epsilon(k)$ to win the interactive BDH game, there exists an algorithm that solves BDH problem with probability $\epsilon(k)^2$.*

We also consider some simple variants of BDH problem, which can be found in Appendix A.2.

4 Strong Challenge-Response Signatures

We now introduce a major building block of our protocol – pairing challenge-response signatures. Similar to the naming of exponential challenge-response signature, pairing challenge-response signature got its name as the signature is in $\mathbb{G}_T$, the range of the pairing function.

Proofs for all the schemes can be found in Appendix B.

4.1 Deterministic Identity-Based Key Generation

We firstly describe the KGC's setup procedure and private key extraction algorithm due to Boneh and Franklin [4], which will be used in our signature schemes and key agreement protocols.

Setup: On input a security parameter k, we use a BDH instance generator [4] to generate $(\mathbb{G}, \mathbb{G}_T, \hat{e})$ where $\mathbb{G}$ and $\mathbb{G}_T$ are groups of prime order q and $\hat{e} : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$ is a pairing. We then choose a cryptographic hash function $\mathcal{H} : \{0,1\}^n \to \mathbb{G}$ and an arbitrary generator $P \in \mathbb{G}$. A random element s is selected from $\mathbb{Z}_q^*$ is assigned to be the KGC's master secret and the corresponding public key is $P_{pub} = sP$. Finally, the set of public parameters is published as $\mathsf{params} = \langle \mathbb{G}, \mathbb{G}_T, \hat{e}, n, P, P_{pub}, \mathcal{H} \rangle$.

Extract: On inputs an identity ID_A and a master secret s, the public key Q_A (for A) is set as $\mathcal{H}(\mathsf{ID}_A)$, and the corresponding private key is $S_A = sQ_A = s\mathcal{H}(\mathsf{ID}_A)$.

4.2 Strong Pairing Challenge-Response Signature Scheme

Setup, Extract: As described in Section 4.1, with an additional hash $\mathcal{H}_0 : \mathbb{G} \times \{0,1\}^* \to \mathbb{Z}_q^*$.

Sign: We suppose B is the verifier asking for a signature on message m from the signer A.

1. B picks b randomly from $\mathbb{Z}_q^*$ and sends $(W_B, W'_B) = (bQ_B, b^{-1}Q_A)$ to A.
2. After verifying $\hat{e}(W_B, W'_B) = \hat{e}(Q_B, Q_A)$, A picks $a \in_R \mathbb{Z}_q^*$ and sends $W_A = aQ_A$ and $e_A = \hat{e}((a + h_A)S_A, W_B)$ to B where $h_A = \mathcal{H}_0(W_A, m)$.

Verify: On input a signature $(W_A, e_A) \in \mathbb{G} \times \mathbb{G}_T$, a message $m \in \{0,1\}^*$, a challenge $b \in \mathbb{Z}_q^*$, a *private key* S_B and an identity string ID_A output accept if $\hat{e}(bS_B, W_A + h_A Q_A) = e_A$ where $h_A = \mathcal{H}_0(W_A, m)$, else output reject.

Discussion: Notice that bQ_B is used as the challenge instead of bP. If bP is used, B can thus easily share the verifiability of the signature with another party C as follows. Firstly, B prepares a challenge $W_B = bP$, then B forwards it as the challenge in the above protocol to get a signature from A. Once the signature is obtained, both B and C can then verify the signature by sharing the knowledge of b. Such an attack is possible since leaking b does no harm to B, in other words, B's private key is not compromised. We remark that Krawczyk's XCR scheme [18] also suffers from a similar problem, although this is not the original goal to ensure non-transferability. Non-transferability of our scheme is ensured by requiring the knowledge of the private key in the verification.

Moreover, the bilinearity ensures the challenge is in the correct form, assuming the discrete logarithm of Q_B with respect to Q_A is unknown. If symmetric pairing is used, it is possible that $(W_B, W'_B) = (b^{-1}Q_A, bQ_B)$ can pass the test; but B can no longer verify the signature.

4.3 Dual Challenge-Response Signature Scheme

A dual version of XCR scheme is proposed in [18], which means both parties assume the dual roles of challenger and signer; and each of them will produce a signature that no third party can forge.

With the notation introduced previously, our ID-based dual challenge-response signature on message m_A and m_B is in the form of $(W_A = aQ_A, W_B = bQ_B, e = \hat{e}(Q_A, Q_B)^{s(a+h_A)(b+h_B)})$, where $h_A = \mathcal{H}_0(W_A, m_A)$ and $h_B = \mathcal{H}_0(W_B, m_B)$. Both parties can compute e by either $\hat{e}((a + h_a)S_A, W_B + h_B Q_B)$ or $\hat{e}(W_A + h_A Q_A, (b + h_B)S_B)$.

Like XCR signature, the verifier who chose the challenge in the protocol can compute exactly the same pairing challenge-response signature using the random value chosen, which is the essential property for us to derive the key agreement protocol. The validity checking of W_A and W_B are omitted as using other value will only resulting in a non-matching key in the key agreement setting.

5 High Performance Identity-Based Key Agreement Protocol

5.1 Basic Construction

Now we propose our high performance identity-based key agreement protocol, as described in Figure 1. Basically the protocol is just the one for creating the ID-based dual challenge-response

signature as described in previous section, with the messages being signed set to the peer's identity. The notation used in the protocol is as follows: $\mathcal{H}$ and $\mathcal{H}_0$ are defined as in Section 4, $\mathcal{K}$ denotes a key derivation oracle, (Q_U, S_U) denotes the public/private key pair for some protocol participant U, sk_U and sid_U denote the session key and session identifier for some protocol participant U respectively, $||$ denotes the concatenation of messages, $\in_R$ denotes choosing an element uniformly at random from the corresponding domain.

Initiator A		Responder B
$a \in_R \mathbb{Z}_q^*; W_A := aQ_A$	$\xrightarrow{ID_A, W_A}$	$b \in_R \mathbb{Z}_q^*; W_B := bQ_B$
		$h_A := \mathcal{H}_0(W_A, \mathsf{ID}_B)$
		$sid_B := ID_A \|\| W_A \|\| ID_B \|\| W_B$
		$sk_B := \mathcal{K}(\hat{e}(W_A + h_A Q_A, (b + h_B) S_B))$
$h_B := \mathcal{H}_0(W_B, \mathsf{ID}_A)$	$\xleftarrow{ID_B, W_B}$	
$sid_A := ID_A \|\| W_A \|\| ID_B \|\| W_B$		
$sk_A := \mathcal{K}(\hat{e}((a + h_A) S_A, W_B + h_B Q_B))$		
$sk_A = \mathcal{K}(\hat{e}(Q_A, Q_B)^{s(a+h_A)(b+h_B)}) = sk_B$		

Fig. 1. Proposed high-performance identity-based key agreement protocol

Informally, we claim that our protocol has the following security attributes.

Known Key Security. It is reasonable to assume that the adversary is able to obtain session keys from any session different from the one under attack. A protocol has known-key security if it is secure under this assumption. This is generally regarded as a standard requirement for key establishment protocols. As Blake-Wilson, Johnson and Menezes [3] indicated, Session-Key Reveal queries in the BR and CK models is designed to capture this notion.

Unknown Key-Share Security. Sometimes the adversary may be unable to obtain any useful information about a session key, but can deceive the protocol principals about the identity of the peer entity. Such an attack was first described by Diffie, van Oorschot, and Wiener in 1992 [17], and can result in principals giving away information to the wrong party or accepting data as coming from the wrong party. Protocols proven secure in the BR or CK models that allow the Corrupt query are also secure against the unknown-key share attack. That is, if a key is to be shared between some parties, U_1 and U_2, the corruption of some other (non-related) player in the protocol, say U_3, should not expose the session key shared between U_1 and U_2 [13].

Forward Secrecy. When the long-term key of an entity is compromised, the adversary can masquerade as that entity in the future. The situation will be even worse if the adversary can also use the compromised long-term key to obtain session keys that were accepted before the compromise. Protocols that prevent this are said to provide forward secrecy. Since there is usually a computational cost in providing perfect forward secrecy, it is sometimes sacrificed and a weaker notion is considered. One example is *partial* forward secrecy whereby the compromise of *one* long-term private key or *both* ephemeral secrets of the communicating parties does not lead to the leakage of past session keys. However, no protection is made when *both* parties' long-term keys are compromised. This notion is considered in existing ID-based protocols like Chen and

Kudla's [8]. Indeed, any two-message protocol without previous establishment of secure shared state cannot achieve *perfect* forward secrecy, according to a negative result of Krawczyk [18].

Key Compromise Impersonation Resistance. Key compromise may lead to another problem. When the long-term key of an entity, A, is compromised; the adversary may be able to masquerade not only *as* A but also *to* A as another party, B. Resistance to such attacks is often seen as desirable.

Theorem 1 captures known key security and unknown key-share security.

Theorem 1 *The protocol described in Figure 1 is secure (in the sense of Definition 5 in Appendix A) assuming that the Bilinear Diffie-Hellman (BDH) problem is hard and $\mathcal{H}$, $\mathcal{H}_0$, and $\mathcal{K}$ are modelled as random oracles.*

Proof. With an adversary $\mathcal{A}$ that has a non-negligible advantage against our protocol described in Figure 1, we construct a simulator, $\mathcal{S}$, against the interactive game with a BDH challenger (the BDH problem instance is (P, xP, yP, zP) and the last part of the challenge is h), using $\mathcal{A}$ as a subroutine. $\mathcal{S}$ now simulates the view of $\mathcal{A}$ by answering the following queries of $\mathcal{A}$.

Setup : xP is assigned to be the public key of the KGC.

$\mathcal{H}$ *queries* : If an $\mathcal{H}$ query is previously asked, then the stored answer in the list $L_{\mathcal{H}}$ will be returned. Suppose the J^{th} distinct $\mathcal{H}$ query is ID_J, then $\mathcal{S}$ responses with yP; otherwise, $\mathcal{S}$ chooses $r_i \in_R \mathbb{Z}_q^*$, stores it in the list $L_{\mathcal{H}}$ along with ID_I, and outputs $r_i P$.

$\mathcal{H}_0$ *queries* : $\mathcal{S}$ maintains a list $L_{\mathcal{H}_0}$ to ensure that previously asked queries would receive the same answer. However, special value may be plugged into the list in the simulation of the Send queries with ID_J as the initiator and ID_K as the responder.

$\mathcal{K}$ *queries* : $\mathcal{S}$ just needs to ensure the random oracle property of $\mathcal{K}$, by maintaining a list $L_{\mathcal{K}}$ to ensure that previously asked queries will receive the same answer.

$\mathsf{Corrupt}$ *queries* : The simulation fails (event I) if the request is ID_J, otherwise the corresponding r_i is retrieved from the list $L_{\mathcal{H}}$ and $r_i(xP)$ is returned.

Send *queries (*ID_I *as initiator and* ID_J *as responder)* : Since $\mathcal{S}$ can compute the private key of ID_I so the simulation can be done as a typical protocol invocation. Except for the following special handling for the N^{th} invocation, τ is chosen randomly from $\mathbb{Z}_q^*$ and $W_{I,N} = r_I \tau(zP)$ is returned. After $W_{J,N}$ is obtained, if $(W_{J,N}, \mathsf{ID}_I)$ can be found in list $L_{\mathcal{H}_0}$, the simulation fails (event II). Otherwise, $\mathcal{S}$ dumps all maintained lists and system parameters to the tape σ, then outputs (X, σ) where $X = W_{J,N}$. The interactive BDH challenger returns $h \in_R \mathbb{Z}_q^*$. $\mathcal{S}$ reconstructs all the lists and system parameters from σ, and set $\mathcal{H}_0(W_{J,N}, \mathsf{ID}_I) = h$, also denoted as $h_{J,N}$.

Send *queries (*ID_J *as initiator and* ID_K *as responder)* : In this case, $\mathcal{S}$ knows neither the private key of the initiator ID_J, nor the ephemeral Diffie-Hellman key of the responder ID_K. However, $\mathcal{S}$ can still do a faithful simulation by manipulating the random oracle. Suppose it is the ℓ^{th} invocation of the protocol initiated by ID_J and responded with ID_K. $\mathcal{S}$ selects $\alpha_\ell, h_{J,\ell} \in_R \mathbb{Z}_q^*$, responses with $W_{J,\ell} = \alpha_\ell P - h_{J,\ell} Q_J$, and stores h_J as the response of $\mathcal{H}_0$ corresponding to the query $(W_{J,\ell}, \mathsf{ID}_K)$. α_ℓ is also stored in the auxiliary list corresponding to the $\Pi^n_{J,K}$ session.

Key Reveal *queries* : For session having ID_I as initiator and ID_J as responder, and if this is not the N^{th} invocation, $\mathcal{S}$ simply uses the private key of ID_I to answer the query asked by $\mathcal{A}$ since $\mathcal{S}$ knows the ephemeral Diffie-Hellman key chosen; otherwise, it fails (event III). For the case (ID_J, ID_K), suppose $h_{J,\ell} = \mathcal{H}_0(W_{J,\ell}, \mathsf{ID}_K)$ and $h_{K,\ell} = \mathcal{H}_0(W_{K,\ell}, \mathsf{ID}_J)$. $\mathcal{S}$ retrieves α_ℓ and returns $\mathcal{K}(\hat{e}(\alpha_\ell(xP), W_{K,\ell} + h_{K,\ell}Q_K)^x) = \mathcal{K}(\hat{e}(\alpha_\ell P - h_{J,\ell}Q_J + h_{J,\ell}Q_J, W_{K,\ell} + h_{K,\ell}Q_K)^x) = \mathcal{K}(\hat{e}(W_{J,\ell} + h_{J,\ell}Q_J, W_{K,\ell} + h_{K,\ell}Q_K)^x)$.

State Reveal *queries* : For session having ID_I as initiator and ID_J as responder, it is trivial to obtain the ephemeral Diffie-Hellman key, except for the N^{th} invocation where $\mathcal{S}$ will fail (event IV). For the case (ID_J, ID_K), it is not supported.

$\mathcal{S}$ knows all the outgoing and incoming DH values, even for the N^{th} invocation between ID_I and ID_J. $\mathcal{S}$ also knows the keying material for all sessions, except the N^{th} invocation (event IV).

Test *queries* : Suppose $h_{I,N} = \mathcal{H}_0(W_{I,N}, \mathsf{ID}_J)$ and $h_{J,N} = \mathcal{H}_0(W_{J,N}, \mathsf{ID}_I) = h$. If $\mathcal{A}$ does not choose the session $\Pi^N_{I,J}$, $\mathcal{S}$ aborts (event V). $\Pi^N_{I,J}$ should hold a session key of the following form.

$$\begin{aligned}
&\mathcal{K}(\hat{e}(W_{I,N} + h_{I,N}Q_I, W_{J,N} + h_{J,N}Q_J)^x) \\
&= \mathcal{K}(\hat{e}(r_I\alpha(zP) + h_{I,N}r_IP, X + hyP)^x) \\
&= \mathcal{K}(\hat{e}((\alpha z + h_{I,N})r_IP, X + hyP)^x) \\
&= \mathcal{K}(\hat{e}(xP, X + hyP)^{(\alpha z + h_{I,N})r_I}).
\end{aligned}$$

Note that $\mathcal{S}$ cannot compute $\mathcal{K}(\hat{e}(xP, X + h(yP))^{z(r_I\alpha)})$ by itself (without $\mathcal{A}$ helping), so $\mathcal{S}$ is unable to return a real session key, and thus the only choice is to return a random key drawn from session key distribution (range of $\mathcal{K}$).

Answering interactive BDH challenger : If $\mathcal{S}$ does not abort and $\mathcal{A}$ is able to distinguish between real session key and random session key (with probability $\epsilon(k)$), then $\mathcal{A}$ must have queried the key derivation oracle $\mathcal{K}$ for the keying material $\hat{e}(xP, X + hyP)^{(\alpha z + h_{I,N})r_I} = \hat{e}(xP, X + h(yP))^{z(r_I\alpha)}\hat{e}(xP, X + h(yP))^{h_{I,N}r_I}$ (we ignore the small probability that $\mathcal{A}$ correctly guess this value without making the corresponding $\mathcal{K}$ query – a standard argument in proof in random oracle model). Now $\mathcal{S}$ randomly chooses one of $\mathcal{A}$'s $\mathcal{K}$'s queries π. If $\mathcal{S}$ is lucky enough that π is the above keying material (event VI), $\mathcal{S}$ answers the interactive BDH challenger correctly with $(\pi/(\hat{e}(xP, X + h(yP))^{h_{I,N}r_I})^{1/(r_I\alpha)}$.

Probability analysis :

I. If event V does not occur, neither does event I.
II. Let $N_{\mathcal{H}}$ be the number of $\mathcal{H}_0$ queries and k be the security parameter, collusion would not occur with probability $(2^k - N_{\mathcal{H}})/2^k$.
III. If event V does not occur, neither does event III.
IV. If event V does not occur, neither does event IV.
V. Let $N_{\mathcal{C}}$ be the number of sessions created, $\mathcal{A}$ chooses the session $\Pi^N_{I,J}$ with probability $1/N_{\mathcal{C}}$.
VI. Let $N_{\mathcal{K}}$ be the number of key derivation oracle queries, event VI occurs with probability $1/N_{\mathcal{K}}$.

$\mathcal{S}$ wins the game if event II and V does not occur but event VI occurs. If $\mathcal{A}$ is able to have an advantage $\epsilon(k)$ against our protocol, then $\mathcal{S}$ can also win with an advantage of at least $\frac{\epsilon(k)(2^k - N_{\mathcal{H}})}{N_{\mathcal{C}}N_{\mathcal{K}}2^k}$. However, since such an adversary $\mathcal{A}$ does not exist, the proof for Theorem 1 follows easily. □

However, the CK model on its own is unable to capture attacks such as partial forward secrecy and key compromise impersonation resistance. Therefore, we will prove that our proposed protocol described in Figure 1 provides these two properties in separate proofs – Theorems 2 and 3.

Theorem 2 *The protocol described in Figure 1 provides partial forward secrecy assuming that the BDH problem is hard and $\mathcal{H}$, $\mathcal{H}_0$, and $\mathcal{K}$ are modelled as random oracles.*

Proof. The simulation of our basic proof (i.e., indistinguishability) allows the adversary to ask Corrupt query for the ID_I associated with the test session, it follows that our protocol also achieve partial forward-secrecy. □

In the CK model, a party can no longer interact with any other protocol participants once it has been corrupted. Therefore, one cannot make any security guarantees about future sessions associated with a corrupted party and it follows easily that key compromise impersonation (KCI) resistance is not considered. Below recalls the formal definition for KCI resistance given by Krawczyk.

Definition 2 (Security Against KCI [18]). *We say that an adversary, $\mathcal{A}$, with access to the private key of party A (but not the private key of party B) succeeds in a KCI-resistance attack against A, if $\mathcal{A}$ is able to distinguish from random the session key of a complete session at A for which the session peer is uncorrupted and the session and its matching session (if it exists) are clean (i.e., $\mathcal{A}$ does not have access to the session's state at the time of session establishment nor has $\mathcal{A}$ issued any* Reveal *query against the session upon its completion).*

Theorem 3 *The protocol described in Figure 1 provides key compromise impersonation resilience assuming that the BDH problem is hard and $\mathcal{H}$, $\mathcal{H}_0$, and $\mathcal{K}$ are modelled as random oracles.*

Proof. Following the approach of Chen and Kudla [8] (and of Krawczyk [18]), we make a slight modification to the security model to capture KCIR – the adversary $\mathcal{A}$ is allowed to corrupt ID_I, the initiating party. Recall that simulation by $\mathcal{S}$ will not abort even if $\mathcal{A}$ requested for the private key of ID_I. The proof for Theorem 1 will not be invalidated by this change and Theorem 3 follows. □

5.2 Forward-Secrecy and Escrow-Free Protocol

There is an additional concern in forward secrecy for ID-based protocols when compared with those in conventional public key cryptography, since the master secret of the KGC is another secret that could become compromised. When this happens, it is clear that the long-term keys of all users will be compromised, but it is possible that a protocol can still provide forward secrecy such that no old session keys can be computed even if the master secret becomes known. On the other hand, achieving this notion means the secure communication made possible by the key agreement protocol is *escrow-free*, assuming no active attacks by the KGC (e.g. by actively impersonating a user). We will say that a protocol provides *KGC forward secrecy* ($\mathsf{KGC}-\mathsf{FS}$) if it retains confidentiality of session keys even when the master secret of the KGC is compromised. It is easy to see that our protocol described in Figure 1 does not provide $\mathsf{KGC}-\mathsf{FS}$ since if any adversary with the knowledge of s can compute $\hat{e}(W_A + h_A Q_A, W_B + h_B Q_B)^s = \hat{e}(Q_A, Q_B)^{s(a+h_A)(b+h_B)}$.

$\mathsf{KGC}-\mathsf{FS}$ implies the forward secrecy in the usual sense since all users' private keys can be computed with the master secret. Our protocol, having only two messages in the message flow, inherently cannot achieve *perfect* forward secrecy, not to say perfect $\mathsf{KGC}-\mathsf{FS}$. Here we consider

Initiator A		Responder B
$a \in_R \mathbb{Z}_q^*; W_A := aQ_A; \underline{T_A := aP}$	$\xrightarrow{ID_A, W_A, \underline{T_A}}$	$b \in_R \mathbb{Z}_q^*; W_B := bQ_B; \underline{T_B := bP}$
		$h_A := \mathcal{H}_0(W_A, \underline{T_A}, \mathsf{ID}_B)$
		$sid_B := \mathsf{ID}_A \Vert W_A \Vert \underline{T_A} \Vert \mathsf{ID}_B \Vert W_B \Vert \underline{T_B}$
		$sk_B := \mathcal{K}(\hat{e}(W_A + h_A Q_A, (b + h_B) S_B), \underline{bT_A})$
$h_B := \mathcal{H}_0(W_B, \underline{T_B}, \mathsf{ID}_A)$	$\xleftarrow{ID_B, W_B, \underline{T_B}}$	
$sid_A := \mathsf{ID}_A \Vert W_A \Vert \underline{T_A} \Vert \mathsf{ID}_B \Vert W_B \Vert \underline{T_B}$		
$sk_A := \mathcal{K}(\hat{e}((a + h_A) S_A, W_B + h_B Q_B), \underline{aT_B})$		
$sk_A = \mathcal{K}(\hat{e}(Q_A, Q_B)^{s(a+h_A)(b+h_B)}, \underline{abP}) = sk_B$		

Fig. 2. Proposed escrow-free high-performance identity-based key agreement protocol

weak $\mathsf{KGC} - \mathsf{FS}$, such that the sessions created without the active involvement of the adversary cannot be recovered after the key compromise. We adopt the approach of Chen and Kudla [8] to give our protocol the same level of KGC-forward-secrecy as their protocol. The new protocol is described in Figure 2, with the underlined value indicates the changes from Figure 1.

Informally, the protocol described in Figure 2 does provide $\mathsf{KGC} - \mathsf{FS}$ at the additional expense of two offline scalar-point multiplications and one online scalar-point multiplication. Master secret s cannot help the adversary in computing $\mathcal{K}(\hat{e}((a+h_A)Q_A, (b+h_B)Q_B)^s, \underline{abP})$ as finding abP means the CDH problem is solvable (recall that both a and b are deleted from the internal states upon completion of the protocol execution). Security assurance is given by the following three theorems.

Theorem 4 *The protocol described in Figure 2 is secure (in the sense of Definition 5 in Appendix A) assuming that the Modified (Computational) Bilinear Diffie-Hellman (MBDH) problem is hard and $\mathcal{H}$, $\mathcal{H}_0$, and $\mathcal{K}$ are modelled as random oracles.*

Proof. For brevity we only highlight the changes that should be made on the indistinguishability proof for our basic protocol. Recall that in additional to $W_A = aQ_A$ ($W_B = bQ_B$), an additional element $T_A = aP$ ($T_B = bP$) is included in the message flow.

Send *queries (*ID_I *as initiator and* ID_J *as responder)* : For the N^{th} invocation, $W_{I,N} = r_I \tau(zP)$ is responded, since $Q_I = r_I P$, the implicit ephemeral secret is τz. The corresponding $T_{I,N}$ can be computed by $\tau(zP)$.

Send *queries (*ID_J *as initiator and* ID_K *as responder)* : Now $W_{J,\ell} = \alpha_\ell P - h_{J,\ell} Q_J$, the implicit ephemeral secret is $\alpha_\ell y^{-1} - h_{J,\ell}$, the corresponding $T_{J,\ell}$ can be computed by $\alpha_\ell(y^{-1}P) - h_{J,\ell}P$.

These two are the only necessary changes for $\mathcal{S}$'s simulation. Of course what $\mathcal{S}$ can solve is MBDH problem but not BDH problem, since $y^{-1}P$ is needed. □

Theorem 5 *The protocol described in Figure 2 provides weak* KGC-forward-secrecy *(*$\mathsf{KGC} - \mathsf{FS}$*) assuming that the Computational Diffie-Hellman (CDH) problem is hard and $\mathcal{H}$, $\mathcal{H}_0$, and $\mathcal{K}$ are modelled as random oracles.*

Proof. The proof for weak KGC-forward-secrecy is similar to that of Chen and Kudla [8]. We construct a simulator $\mathcal{S}$ that solve the CDH problem (P, xP, yP) with the help of an adversary $\mathcal{A}$ which has a non-negligible advantage against the security of our protocol described in Figure 2.

Setup : A random s is chosen from $\mathbb{Z}_q^*$ and given to $\mathcal{A}$, KGC's public key is sP.

$\mathcal{H}$ *queries* : $\mathcal{S}$ chooses $r_i \in_R \mathbb{Z}_q^*$, stores (r_i, ID_i) in the list $L_{\mathcal{H}}$, and outputs $r_i P$.

$\mathcal{H}_0$ *queries* : $\mathcal{S}$ maintains a list $L_{\mathcal{H}_0}$ to ensure the random oracle property.

$\mathcal{K}$ *queries* : Maintains the random oracle properties, with the help of a list $L_{\mathcal{K}}$.

Corrupt *queries* : With s, $\mathcal{A}$ can answer itself.

Send *queries* : For $\Pi_{I,J}^k$, $\mathcal{S}$ answers $W_I = t_k a Q_I = t_k r_I(aP)$ and $T_I = t_k(aP)$. For $\Pi_{J,I}^\ell$, $\mathcal{S}$ answers $W_J = t_\ell b Q_J = t_\ell r_J(bP)$ and $T_J = t_\ell(bP)$.

Key Reveal *and* State Reveal *queries* : Fully supported, except $\Pi_{I,J}^k$ and $\Pi_{J,I}^\ell$.

Test *queries* : $\mathcal{S}$ aborts if $\mathcal{A}$ does not choose the oracle $\Pi_{I,J}^k$. Otherwise, $\Pi_{I,J}^k$ must have accepted after having had a matching conversation with another oracle, if it happens to be $\Pi_{J,I}^\ell$, the second component of the keying material is $t_k t_\ell(abP)$. Note that $\mathcal{S}$ cannot compute this value by itself (without $\mathcal{A}$ helping), so it cannot returns a real session key, and thus the only choice is to return a random key drawn from session key distribution (range of $\mathcal{K}$).

Solving the CDH problem : If $\mathcal{S}$ does not abort and $\mathcal{A}$ is so clever (with probability $\epsilon(k)$) that can distinguish between real session key and random session key, $\mathcal{A}$ must have queried the key derivation oracle $\mathcal{K}$ for the keying material $(*, t_k t_\ell(abP))$ (where $*$ is something does not really matter) to output its guess correctly. Now $\mathcal{S}$ randomly chooses one of $\mathcal{A}$'s $\mathcal{K}$'s queries $(*, \pi)$. If $\mathcal{S}$ is lucky enough that π is the above keying material, $\mathcal{S}$ outputs the solution as $\pi/(t_k t_\ell)$.

Probability analysis : Considering the events above, $\mathcal{S}$ wins the game with probability $\epsilon(k)/(N_{\mathcal{C}}^2 N_{\mathcal{K}})$, where $N_{\mathcal{C}}$ is the number of sessions created and $N_{\mathcal{K}}$ is the number of key derivation oracle queries. □

Theorem 6 *The protocol described in Figure 2 provides key compromise impersonation assuming that the Modified Bilinear Diffie-Hellman (MBDH) problem is hard.*

Proof. As the KCIR proof for our basic protocol, $\mathcal{S}$ can correctly answer Corrupt query of ID_I, the initiating party. Hence KCIR follows. □

5.3 Security and Complexity Evaluation

Table 1 describes a summary of two parties, two messages ID-based protocols. M denotes scalar-point multiplication, H denotes MapToPoint function [4] hashing identity to a point on an elliptic curve, and P denotes pairing in the table. Off-line computation can be pre-computed before the execution of the protocol, which includes public key derivation. Note that pairings are expensive and should be avoided whenever possible. MapToPoint is slightly more expensive but its cost is still comparable with that of scalar-point multiplication. For security, BR (NR) denotes a restricted variant of the BR model whereby Session-Key Reveal query[2] is not supported, $\mathsf{KGC-FS}$ denotes *KGC forward secrecy*, and KCIR denotes key compromise impersonation resistance.

[2] Protocols in BR (NR) do not provide assurance against known (session) key attacks.

Protocol	**Computation**			**Forward Secrecy?**	**KCIR?**	**Proof / Attack?**
	On-line	Off-line	Public Key			
Our protocol 1	$1M + 1P$	$2M$	$1H$	Yes (No $\mathsf{KGC-FS}$)	Yes	CK model
Our protocol 2	$2M + 1P$	$4M$	$1H$	Yes (Weak $\mathsf{KGC-FS}$)	Yes	CK model
Wang [32]	$1M + 1P$	$2M$	$1H$	Yes (No $\mathsf{KGC-FS}$)	Yes	BR model
The following protocols are proven secure in a restricted model.						
Chen and Kudla [8] #2	$1M + 1P$	$2M$	$1H$	No	Yes	BR (NR)
Chen and Kudla [8] #2'	$2M + 1P$	$4M$	$1H$	Yes (Weak $\mathsf{KGC-FS}$)	No	BR (NR)
McCullagh and Barreto [21] #1	$1M + 1P$	$1M$	$1M$	Yes (No $\mathsf{KGC-FS}$)	No	BR (NR)
McCullagh and Barreto [21] #2	$1M + 1P$	$1M$	$1M$	Yes [3]	No	BR (NR)[4]
The following protocols do not have any security proofs.						
Smart [30]	$1P$	$2M + 1P$	$1H$	No	Yes	No
Chen and Kudla [8] #1'	$1M + 1P$	$2M + 1P$	$1H$	Yes (Weak $\mathsf{KGC-FS}$)	Yes	No
The following protocols are broken.						
Yi [34]	$2M$	$1M + 1P$	$1H$	Yes (No $\mathsf{KGC-FS}$)	Yes	Appendix C
Choie *et al.* [12] #1	$2M + 3P$	$2M$	$1H$	Yes (No $\mathsf{KGC-FS}$)	Yes	[5]
Choie *et al.* [12] #2	$1M + 2P$	$1M$	$1H$	Yes (No $\mathsf{KGC-FS}$)	Yes	[5]
Shim [27]	$1P$	$2M$[5]	$1H$	No	No	[31]
Xie [33] #1	$1M + 1P$	$2M + 1P$	$1M$	Yes (No $\mathsf{KGC-FS}$)	Yes	[28]
Xie [33] #2	$1M + 1P$	$2M + 1P$	$1M$	Yes [3]	Yes	[28]

Table 1. Security and efficiency for two-party, two-message ID-based protocols

As shown in Table 1, among the "unbroken" ID-based protocols that provide:

KCIR and FS (not KGC-FS). Our protocol described in Figure 1 and Wang's protocol [32] are the most efficient. However, our protocol is based on a milder assumption and yet proven secure in a stronger model, which makes it more attractive than that of Wang's.

KCIR and $\mathsf{KGC-FS}$. Although our protocol described in Figure 2 is as efficient as that of Chen and Kudla [8] protocol #2', our protocol is proven secure in a stronger model (allowing the adversary to ask the Session-State Reveal query). It is easy to see that Chen and Kudla protocol #2' will not be secure if the adversary is allowed to ask Session-State Reveal query. Therefore, an adversary can always obtain either value to compute the agreed session key.

6 Key Agreement Protocols among Spontaneous Anonymous Groups

In a key agreement protocol among spontaneous anonymous groups (or SAG key agreement protocol in short), the initiator conscripts a set of users which is called the *initiating ring* and similarly the responder hides in a *responding ring*.

[3] No formal proof is given, it is unclear that whether the protocol can achieve anything stronger than weak $\mathsf{KGC-FS}$.
[4] It is secure in the BR (NR) model if the mistakes in the proof as pointed out in [9, 14] are corrected.
[5] This protocol uses a mathematically undefined operation whereby it involves an exponentiation of an element $\hat{e}(P, P) \in \mathbb{G}_T$ to a power that is a multiplication of an element in $\mathbb{Z}_q^*$ and an element in $\mathbb{G}_T$.

6.1 Our Extension

We now describe our extended protocol for spontaneous anonymous groups, which can be seen as an extension from the ID-based ring signature schemes due to Chow *et al.* [15] or from an SAG extension of our strong pairing challenge-response signature scheme (or challenge-response ring signature, which is not explicitly discussed here since it is not the main emphasis of this paper).

Let A_j be a member of the initiating ring $A = \{A_1, A_2, \ldots, A_J\}$ and B_k be a member of the responding ring $B = \{B_1, B_2, \ldots, B_K\}$. Note that in our protocol, it is *not* required that $J = K$. For the proof of security, we require each user to derive a value ψ in each session that is different from the values chosen in previous sessions with overwhelming probability. Denote these value of A_j and B_k by ψ_A and ψ_B respectively, Canetti and Krawczyk suggested such a pair of (ψ_A, ψ_B) constitutes a unique SID for each session in practice.

1. A_j chooses $U_i \in_R \mathbb{G}$ and computes $h_i = \mathcal{H}_0(U_i, B, \psi_A), \forall i \in \{1, 2, \ldots, J\} \setminus \{j\}$.
2. Similarly, B_k chooses $V_i \in_R \mathbb{G}$ and computes $c_i = \mathcal{H}_0(V_i, A, \psi_B), \forall i \in \{1, 2, \ldots, K\} \setminus \{k\}$.
3. A_j chooses $r'_j \in_R \mathbb{Z}_q^*$, computes $U_j = r'_j Q_{A_j} - \sum_{i \neq j}\{U_i + h_i Q_{A_i}\}$.
4. Similarly, B_k chooses $r'_j \in_R \mathbb{Z}_q^*$, computes $V_k = r'_k Q_{B_k} - \sum_{i \neq k}\{V_i + c_i Q_{B_i}\}$.
5. A_j and B_k exchange $\bigcup_{i \in \{1,2,\ldots,J\}} \{U_i\}$ and $\bigcup_{i \in \{1,2,\ldots,K\}} \{V_i\}$
6. A_j and B_k compute respectively their session key, sk_A and sk_B, as follows.

$$
\begin{aligned}
sk_A &= \mathcal{K}(\hat{e}((r'_j + h_j)S_{A_j}, \sum_{i=1}^{K}(V_i + c_i Q_{B_i}))) = \mathcal{K}(\hat{e}(r'_j Q_{A_j} + h_j Q_{A_j}, \sum_{i=1}^{K}(V_i + c_i Q_{B_i}))^s) \\
&= \mathcal{K}(\hat{e}(U_j + \sum_{i \neq j}\{U_i + h_i Q_{A_i}\} + h_j Q_{A_j}, \sum_{i=1}^{K}(V_i + c_i Q_{B_i}))^s) \\
&= \mathcal{K}(\hat{e}(\sum_{i=1}^{J}(U_i + h_i Q_{A_i}), \sum_{i=1}^{K}(V_i + c_i Q_{B_i}))^s) \\
&= \mathcal{K}(\hat{e}(\sum_{i=1}^{J}(U_i + h_i Q_{A_i}), V_k + \sum_{i \neq k}\{V_i + c_i Q_{B_i}\} + c_k Q_{B_k})^s) \\
&= \mathcal{K}(\hat{e}(\sum_{i=1}^{J}(U_i + h_i Q_{A_i}), (r'_k + c_k)S_{B_k})) = \mathcal{K}(\hat{e}(\sum_{i=1}^{J}(U_i + h_i Q_{A_i}), r'_k Q_{B_k} + c_k Q_{B_k})^s) = sk_B
\end{aligned}
$$

6.2 Security Attributes

For the brevity of discussion about security attributes we assume both rings are of the same size n. Apart from the conventional security properties for key agreement protocols, the security key agreement protocols among SAGs also depends on 1-out-of-n anonymity as described in Definition 3.

Definition 3 (Security Attributes of SAG Key Agreement Protocols). *A SAG key agreement protocol is secure if below conditions are satisfied.*

1: Validity. *If two uncorrupted oracles complete matching sessions, then both oracles must hold the same session key.*

2: Indistinguishability. *For all probabilistic, polynomial time adversaries, $\mathcal{A}$, the advantage of $\mathcal{A}$, $\mathsf{Adv}^{\mathcal{A}}(k)$, in game $\mathcal{G}^6$ is negligible. In particular, this implies* **1-out-of-n authenticity***: for all probabilistic, polynomial time adversaries, $\mathcal{A}$, without any one of the n private keys, has negligible advantage in learning about a fresh session key.*

3: 1-out-of-n Anonymity. *A key agreement protocol among SAGs is said to have unconditional anonymity if for any group of n users, any adversary $\mathcal{A}$ (including the responder and the KGC) is unable to identify the real initiator better than a random guess, i.e. $\mathcal{A}$ can guess the identity of the initiator correctly with probability no better than $\frac{1}{n}$, or $\frac{1}{n-1}$ if $\mathcal{A}$ is in the ring. If the protocol satisfies* bilateral *privacy, the same requirement applies on the responding party.*

Validity is easy to check. Indistinguishability is formally captured by the following theorem.

Theorem 7 *The protocol described in Section 6.1 achieves indistinguishability (in the sense of Definition 3) assuming that the Bilinear Diffie-Hellman (BDH) problem is hard and $\mathcal{H}$, $\mathcal{H}_0$, and $\mathcal{K}$ are modelled as random oracles.*

Proof. We construct a simulator $\mathcal{S}$ that wins the interactive game with a BDH challenger (the BDH problem instance is (P, xP, yP, zP) and the last part of the challenge is h) with the help of an adversary $\mathcal{A}$ that can break our protocol.

Setup : KGC's public key is xP.

$\mathcal{H}$ *queries* : $\mathcal{S}$ embeds yP in the answer of many $\mathcal{H}$ queries, which depends on the result of flipping a coin $W \in \{0, 1\}$ yielding 0 with probability ζ (to be determined) and 1 with probability $1 - \zeta$.

Suppose ID_i is being queried. $\mathcal{S}$ chooses $r_i \in_R \mathbb{Z}_q^*$. If $W = 0$, r_iP is returned; otherwise, $\mathcal{S}$ responses with $r_i(yP)$. For both cases r_i and W are stored it in the list $L_{\mathcal{H}}$ along with ID_i.

$\mathcal{H}_0$ *queries* : $\mathcal{S}$ maintains a list $L_{\mathcal{H}_0}$ to ensure the random oracle property. However, special value may be plugged into the list in the simulation of Send queries when $\mathcal{S}$ cannot compute the private key of all of the members of a ring.

$\mathcal{K}$ *queries* : Maintains the random oracle properties, with the help of a list $L_{\mathcal{K}}$.

$\mathsf{Corrupt}$ *queries* : The simulation fails if the coin value stored in $L_{\mathcal{H}}$ along with ID_i is 1; otherwise corresponding r_i is retrieved from the list $L_{\mathcal{H}}$ and $r_i(xP)$ is returned.

Send *queries* : Suppose the initiating ring A is of size n_1, the responding ring B is of size n_2, ψ_A and ψ_B are the unique value used by the initiator and the responder respectively. For the creation of session, we consider the following cases.

1. At least one member of the responding ring has the public key in the form of r_iP, which further divided into two cases.
   (a) At least one member of the initiating ring has the public key in the form of r_iP: In this case, $\mathcal{S}$ can compute at least one private key among those in the initiating ring, $\mathcal{S}$ can simply simulate the protocol as normal.
   (b) All members of the initiating ring have the public key in the form of $r_i(yP)$: $\mathcal{S}$ manipulates the random oracle $\mathcal{H}_0$ as follows.

[6] Definition can be found in Appendix A.

i. Chooses an index $s \in_R \{1, 2, \cdots, n_1\}$.
ii. Chooses $U_i \in_R \mathbb{G}$, computes $h_i = \mathcal{H}_0(U_i, B, \psi_A)$ $\forall i \in \{1, 2, \cdots, n_1\} \backslash \{s\}$.
iii. Chooses $h'_s \in_R \mathbb{Z}_q^*$ and $t \in_R \mathbb{Z}_q^*$, computes $U_s = tP - h'_s Q_{ID_s} - \sum_{i \neq s} \{U_i + h_i Q_{ID_i}\}$.
iv. Stores the relationship $h'_s = \mathcal{H}_0(U_s, B, \psi_A)$ to the list $L_{\mathcal{H}_0}$ and stores $T = t(xP)$ into an auxiliary list as an auxiliary data corresponding to this session. If collision occurs (which is not likely since with high probability ψ_A does not repeat), repeats step iii.

2. All members in the responding ring have the public key in the form of $r_i(yP)$, which further divided into three cases.
   (a) No member in the responding ring has a public key in the form of $r_i(yP)$: For first such query, $\mathcal{S}$ embeds the hard problem as follows, otherwise it proceeds as case 1a.
      i. Chooses an index $s \in_R \{1, 2, \cdots, n_1\}$, retrieves (r_s, ID_{A_s}) from $L_{\mathcal{H}}$.
      ii. $\forall i \in \{1, 2, \cdots, n_1\} \backslash \{s\}$, takes $u_i \in_R \mathbb{Z}_q^*$ and keeps a record; then computes $U_i = u_i P \in \mathbb{G}$.
      iii. Computes $h_i = \mathcal{H}_0(U_i, B, \psi_A)$ $\forall i \in \{1, 2, \cdots, n_1\} \backslash \{s\}$.
      iv. Chooses $\tau \in_R \mathbb{Z}_q^*$, and computes $U_i = r_s \tau(zP)$, τ is recorded.
      v. $\mathcal{A}$ will keep on asking the value of $c_i = \mathcal{H}_0(V_i, A, \psi_B)$, $\mathcal{S}$ answers as normal except for the last query $\mathcal{H}_0(V_{n_2}, A, \psi_B)$.
      vi. When $\mathcal{S}$ has collected all V_is for a single session (which is linked by ψ_B), we give $X = \sum_{i \in \{1,2,\cdots,n_2\}} V_i + \sum_{i \in \{1,2,\cdots,n_2-1\}} c_i Q_{B_i}$ to the interactive BDH challenger. If (V_{n_2}, A, ψ_B) can be found in list $L_{\mathcal{H}_0}$ (which is unlikely as argued before), the simulation fails.
      vii. $\mathcal{S}$ dumps all maintained lists and system parameters to the tape σ and outputs (X, σ). The interactive BDH challenger returns $h \in_R \mathbb{Z}_q^*$.
      viii. $\mathcal{S}$ reconstructs all the lists and system parameters from σ, retrieves $(r_{n_2}, ID_{B_{n_2}})$ from $L_{\mathcal{H}}$ and set $\mathcal{H}_0(V_{n_2}, A, \psi_B) = h {r_{n_2}}^{-1}$.
   (b) At least one in the responding ring holds a public key in the form of $r_i P$ (excluding the above case that all members hold keys in $r_i P$ form): $\mathcal{S}$ performs exactly the same simulation as in case 1a.
   (c) All members of the initiating ring have the public key in the form of $r_i(yP)$: $\mathcal{S}$ performs exactly the same simulation as in case 1b.

If the adversary acts as the initiating party, the simulation can also be done in a similar manner as above, with the criterion on the initiator ring and the responding ring interchanged. However, $\mathcal{S}$ will not embed the hard problem in the simulation for case 2a, instead the simulation is done in the way as in case 2b (which is the same as case 1a).

Key Reveal *queries* :

1. For case 1a and 2b, it is trivial to compute the session key.
2. For case 1b and 2c, it is easy to see that $\mathcal{S}$ can use T from the corresponding entry in the auxiliary list to compute the session key.

State Reveal *queries* :

1. For case 1a and 2b, it is trivial to reveal the session state.
2. For case 1b and 2c, $\mathcal{S}$ fails, which is our limitation in revealing session state.

By the game's rule, the adversary will not make Key Reveal or State Reveal queries for case 2a.

Test *queries* : If the session prepared in case 2a is selected, the session key is in the following form.

$$
\begin{aligned}
&\mathcal{K}(\hat{e}(\sum_{i=1}^{n_1}(U_i + h_j Q_{A_j}), \sum_{i=1}^{n_2}(V_i + c_i Q_{B_i}))^x) \\
&= \mathcal{K}(\hat{e}(\sum_{i\in\{1,2,\cdots,n_1\}\backslash\{s\}}(U_i + h_j Q_{A_j}) + h_s Q_{ID_s} + r_s\tau(zP), \\
&\quad \sum_{i\in\{1,2,\cdots,n_2-1\}}(V_i + c_i Q_{B_i}) + V_{n_2} + hyP)^x) \\
&= \mathcal{K}(\hat{e}(wP + r_s\tau(zP), X + hyP)^x) \\
&\quad (w \text{ can be computed with the help of } u_i\text{s and} r_i\text{s previously recorded}) \\
&= \mathcal{K}(\hat{e}(wP, X + hyP)^x \hat{e}(zP, X + hyP)^{xr_s\tau}) \\
&= \mathcal{K}(\hat{e}(w(xP), X + hyP)\hat{e}(xP, X + hyP)^{zr_s\tau}).
\end{aligned}
$$

Note that $\mathcal{S}$ cannot compute $\hat{e}(xP, X + hyP)^{zr_s\tau}$ by itself, so it cannot returns a real session key, thus the only choice is to return a random key drawn from session key distribution (range of $\mathcal{K}$).

Answering interactive BDH challenger : If $\mathcal{S}$ does not abort and $\mathcal{A}$ is so clever (with probability $\epsilon(k)$) that can distinguish between real session key and random session key, $\mathcal{A}$ must have queried the key derivation oracle $\mathcal{K}$ for the keying material $\hat{e}(w(xP), X+hyP)\hat{e}(xP, X+hyP)^{zr_s\tau}$ to output its guess correctly. Now $\mathcal{S}$ randomly chooses one of $\mathcal{A}$'s $\mathcal{K}$'s queries π. If $\mathcal{S}$ is lucky enough (with probability $1/N_{\mathcal{K}}$, where $N_{\mathcal{K}}$ is the number of key derivation oracle queries) that π is the above keying material, $\mathcal{S}$ answers the interactive BDH challenger correctly with $(\pi/\hat{e}(w(xP), X + hyP))^{1/(r_s\tau)}$.

Probability analysis : For Corrupt, the simulation would not fail if all $N_{\mathcal{E}}$ such queries correspond to coin value $W = 0$, the probability of such event is $\zeta^{N_{\mathcal{E}}}$.

For $\mathcal{S}$ to embed the hard problem successfully, we need case 2a to happen for at least one session. Let the initiating ring is of size n_1 and the responding ring B is of size n_2. Case 2a happens with probability $\zeta^{n_1}(1 - \zeta^{n_2})$, ignoring the fact that each identity should be independent – which only results in a small error in probability if n_1 and n_2 are large. For brevity we assume $n_1 = n_2 = n$ and let $\gamma = \zeta^n(1 - \zeta^n)$. Suppose $N_{\mathcal{C}}$ is the number of sessions created. Assuming the choice of the initiating ring and the responding ring for each of these sessions are independent, the probability that $\mathcal{S}$ can successfully embed the hard problem is $1-(1-\gamma)^{N_{\mathcal{C}}}$, which is lower bounded by $N_{\mathcal{C}} \cdot \gamma$.

For $\mathcal{S}$ to solve the hard problem successfully, $\mathcal{A}$ must select the session that $\mathcal{S}$ embedded the hard problem as the test session, which happens with probability $1/N_{\mathcal{C}}$. Combing all probability, a successful simulation occurs with probability $f(\zeta)\epsilon(k)/N_{\mathcal{K}}$, where $f(\zeta) = (\zeta^{N_{\mathcal{E}}})(N_{\mathcal{C}}\zeta^n(1-\zeta^n))/N_{\mathcal{C}} = \zeta^{N_{\mathcal{E}}+n}(1-\zeta^n)$. A simple differentiation shows that $f(\zeta)$ is maximized at $\zeta = (\frac{N_{\mathcal{E}}+n}{N_{\mathcal{E}}+2n})^{1/n}$, this value of ζ maximizes the probability of $\mathcal{S}$ to win the interactive BDH challenger. □

Theorem 8 *The protocol described in Section 6.1 provides 1-out-of-n anonymity unconditionally.*

Proof. Suppose s indexes the real protocol participant among the identities set $\{A_1, A_2, \cdots, A_n\}$. All U_is for $i \in \{1, 2, \cdots, n\}\backslash\{s\}$ are randomly chosen, so does U_s since r'_s is randomly chosen from $\mathbb{Z}_q^*$. Anyway these elements leave no track of real identity. Element related to the real identity is only involved after the key is established. However, $(r'_S + h_S)S_{A_s}$, together with those U_is, constitutes Chow *et al.*'s ring signature [15], which is proven to be unconditionally signer-anonymous, meaning the real protocol participant is hidden in our setting. □

7 Conclusion

We observed the following similarities between signature schemes and key agreement protocols:

1. the existence of ephemeral parameters in both paradigms, and
2. the resemblance of the signing oracle in signature paradigm and the Session-Key Reveal oracle in key agreement paradigm.

Based on a recent work of Krawczyk [18], we propose a strong pairing challenge-response signature scheme. With this new scheme and the above observations, we then propose a new identity-based (ID-based) key agreement protocol, with formal security assurance in the random oracle model. Our protocol is proven secure even if the adversary is allowed access to the Session-Key Reveal and Session-State Reveal queries, which is the *first* work secure against such strong adversary *without* employing any gap assumption. Using of Chen and Kudla's approach [8], we show how to provide KGC forward secrecy. Our proposed protocols are efficient and yet proven secure in a stronger model among other two-party two-message ID-based protocols with a similar set of security attributes.

Motivated by the need for a better anonymous roaming mechanism and our observation that existing research appears to focus only on *unilateral* identity privacy, we extend our basic protocol to realize the first ID-based key agreement protocol with *bilateral* privacy, for key agreement among spontaneous anonymous groups. This can be seen as an extension from Chow *et al.*'s ID-based ring signature scheme [15] or a ring signature variant of our proposed challenge-response signature scheme, which gives another linkage between signature schemes and key agreement protocols.

As a result of our work, we confirm that concepts in signature schemes can, indeed, be employed to build better key agreement protocols.

References

1. G. Ateniese, A. Herzberg, H. Krawczyk, and G. Tsudik. Untraceable Mobility or How to Travel Incognito. *Computer Networks*, 31(8):871–884, 1999.
2. M. Bellare and P. Rogaway. Entity Authentication and Key Distribution. In *CRYPTO 1993*, volume 773 of *LNCS*, pages 110–125. Springer, 1993.
3. S. Blake-Wilson, D. Johnson, and A. Menezes. Key Agreement Protocols and their Security Analysis. In *IMA Cryptography and Coding 1997*, volume 1335 of *LNCS*, pages 30–45. Springer, 1997.
4. D. Boneh and M. Franklin. Identity-Based Encryption from the Weil Pairing. *SIAM Journal on Computing*, 32(3):585–615, 2003.
5. C. Boyd and K.-K. R. Choo. Security of Two-Party Identity-Based Key Agreement. In *Mycrypt 2005*, volume 3715 of *LNCS*, pages 229–243. Springer, 2005.
6. C. Boyd and D. Park. Public Key Protocols for Wireless Communications. In *ICISC 1998*, volume 1807 of *LNCS*, pages 47 – 57. Springer, 1998. (Available from `http://www.fit.qut.edu.au/~boyd/papers/icisc98.ps.gz`).
7. R. Canetti and H. Krawczyk. Analysis of Key-Exchange Protocols and Their Use for Building Secure Channels. In *EUROCRYPT 2001*, volume 2045 of *LNCS*, pages 453–474. Springer, 2001. Extended version available from `http://eprint.iacr.org/2001/040/`.
8. L. Chen and C. Kudla. Identity Based Authenticated Key Agreement Protocols from Pairings. In *CSFW 2003*, pages 219–233. IEEE Computer Society Press, 2003. Corrected version at `http://eprint.iacr.org/2002/184/`.
9. Z. Cheng and L. Chen. On Security Proof of McCullagh-Barreto's Key Agreement Protocol and its Variants. Cryptology ePrint Archive, Report 2005/201, 2005.
10. Z. Cheng, L. Chen, R. Comley, and Q. Tang. Identity-Based Key Agreement with Unilateral Identity Privacy Using Pairings. In *ISPEC 2006*, volume 3903 of *LNCS*, pages 202–213. Springer, 2006.
11. Z. Cheng, M. Nistazakis, R. Comley, and L. Vasiu. On the Indistinguishability-Based Security Model of Key Agreement Protocols-Simple Cases. Cryptology ePrint Archive, Report 2005/129, 2005.
12. Y. J. Choie, E. Jeong, and E. Lee. Efficient Identity-based Authenticated Key Agreement Protocol from Pairings. *Applied Mathematics and Computation*, 162(1):179–188, 2005.

13. K.-K. R. Choo, C. Boyd, and Y. Hitchcock. Examining Indistinguishability-Based Proof Models for Key Establishment Protocols. In *ASIACRYPT 2005*, volume 3788 of *LNCS*, pages 585–604. Springer, 2005.
14. K.-K. R. Choo, C. Boyd, and Y. Hitchcock. On Session Key Construction in Provably Secure Protocols. In *Mycrypt 2005*, volume 3715 of *LNCS*, pages 116–131. Springer, 2005.
15. S. S. M. Chow, S. M. Yiu, and L. C. K. Hui. Efficient Identity Based Ring Signature. In *ACNS 2005*, volume 3531 of *LNCS*, pages 499–512. Springer, 2005.
16. S. S. M. Chow, S. M. Yiu, L. C. K. Hui, and K. P. Chow. Efficient Forward and Provably Secure ID-Based Signcryption Scheme. In *ICISC 2003*, volume 2971 of *LNCS*, pages 352–369. Springer, 2003.
17. W. Diffie, P. C. van Oorschot, and M. J. Wiener. Authentication and Authenticated Key Exchange. *Designs, Codes and Cryptography*, 2:107–125, 1992.
18. H. Krawczyk. HMQV: A High-Performance Secure Diffie–Hellman Protocol. In *CRYPTO 2005*, volume 3621 of *LNCS*, pages 546–566. Springer, 2005. Extended version available from `http://eprint.iacr.org/2005/176/`.
19. C. Kudla and K. G. Paterson. Modular Security Proofs for Key Agreement Protocols. In *ASIACRYPT 2005*, volume 3788 of *LNCS*, pages 549–569. Springer, 2005.
20. S. Kunz-Jacques and D. Pointcheval. About the Security of MTI/C0 and MQV. In Roberto De Prisco and Moti Yung, editors, *Security and Cryptography for Networks, 5th International Conference, SCN 2006, Maiori, Italy, September 6-8, 2006, Proceedings*, volume 4116 of *Lecture Notes in Computer Science*, pages 156–172. Springer, 2006.
21. N. McCullagh and P. S. L. M. Barreto. A New Two-Party Identity-Based Authenticated Key Agreement. In *CT-RSA 2005*, volume 3376 of *LNCS*, pages 262–274. Springer, 2005. Extended version available from `http://eprint.iacr.org/2004/122/`.
22. T. Okamoto and D. Pointcheval. The Gap-Problems: a New Class of Problems for the Security of Cryptographic Schemes. In *PKC 2001*, volume 1992 of *LNCS*, pages 104–118. Springer, 2001.
23. David Pointcheval and Jacques Stern. Security Arguments for Digital Signatures and Blind Signatures. *Journal of Cryptology*, 13(3):361–396, 2000.
24. R. L. Rivest, A. Shamir, and Y. Tauman. How to Leak a Secret. In *ASIACRYPT 2001*, volume 2248 of *LNCS*, pages 552–565. Springer, 2001.
25. R. L. Rivest, A. Shamir, and Y. Tauman. How to Leak a Secret: Theory and Applications of Ring Signatures. In *Theoretical Computer Science: Essays in Memory of Shimon Even*, volume 3895 of *LNCS*, pages 164–186. Springer, 2006.
26. D. Samfat, Re. Molva, and N. Asokan. Untraceability in Mobile Networks. In *ACM MobiCom 1995*, pages 26–36. ACM Press, 1995.
27. K.-A. Shim. Efficient ID-based Authenticated Key Agreement Protocol based on Weil Pairing. *IEE Electronics Letters*, 39(8):653–654, 2002.
28. K.-A. Shim. Cryptanalysis of Two ID-based Authenticated Key Agreement Protocols from Pairings. Cryptology ePrint Archive, Report 2005/357, 2005.
29. V. Shoup. On Formal Models for Secure Key Exchange (Version 4). Technical Report RZ 3120 (#93166), IBM Research, Zurich, 1999.
30. N. Smart. An Identity based Authenticated Key Agreement Protocol based on the Weil Pairing. *IEE Electronics Letters*, 38(13):630–632, 2002.
31. H.-M. Sun and B.-T. Hsieh. Security Analysis of Shim's Authenticated Key Agreement Protocols from Pairings. Cryptology ePrint Archive, Report 2003/113, 2003.
32. Y. Wang. Efficient Identity-Based and Authenticated Key Agreement Protocol. Cryptology ePrint Archive, Report 2005/108, 2005.
33. G. Xie. An ID-Based Key Agreement Scheme from Pairing. Cryptology ePrint Archive, Report 2005/093, 2005.
34. X. Yi. An Identity-Based Signature Scheme from the Weil Pairing. *IEEE Communications Letters*, 7(2):76–78, 2003.

A Background Materials

A.1 Canetti–Krawczyk Model

We now present a brief overview of the CK model [7]. In the CK model, there are two adversarial models, namely the unauthenticated-links / real world model (UM) and the authenticated-links /

ideal world model (AM). AM is the (ideal) world where messages are authenticated magically, and UM is the (real) world in which we want our protocols to be proven secure.

Let $\mathcal{A}_{\mathrm{UM}}$ denote the (active) adversary in UM, and $\mathcal{A}_{\mathrm{AM}}$ denote the (passive) adversary in AM.

AM. $\mathcal{A}_{\mathrm{AM}}$ is allowed to invoke protocol runs, impersonate corrupted protocol participants, and reveal past session keys. However, $\mathcal{A}_{\mathrm{AM}}$ is not allowed to fabricate any messages or send a message more than once.

UM. $\mathcal{A}_{\mathrm{UM}}$ is allowed to do all the things that $\mathcal{A}_{\mathrm{AM}}$ can. In addition, $\mathcal{A}_{\mathrm{UM}}$ can fabricate any messages or send a message more than once.

The difference between $\mathcal{A}_{\mathrm{AM}}$ and $\mathcal{A}_{\mathrm{UM}}$ lies in their power. $\mathcal{A}_{\mathrm{AM}}$ is restricted to only delay, delete, and relay messages but not to fabricate any messages or send a message more than once.

Proving protocols secure in the CK model is usually by translating a provably secure protocol in the AM to a provably secure protocol in the UM with the use of an authenticator. However, we will not use this approach as the use of the authenticator can result in a more expensive protocol. Instead the protocol is proven secure in the UM following Krawczyk's approach [18] (in a similar fashion as proof in the BR model). For simplicity, we will refer $\mathcal{A}_{\mathrm{UM}}$ to as $\mathcal{A}$.

A.1.1 Adversarial Power The adversary, $\mathcal{A}$, controls the communications between the protocol participants by interacting with the set of oracles, $\Pi^i_{U_u,U_v}$, where $\Pi^i_{U_u,U_v}$ is defined to be the i^{th} instantiation of a protocol participant, U_u, in a specific protocol run and U_v is the principal with whom U_u wishes to establish a secret key. $\mathcal{A}$ controls the communication channels via the queries to the targeted oracles. A description of the oracle types is presented as follows.

Send(U_u, U_v, i, m) **query.** This query to an oracle, $\Pi^i_{U_u,U_v}$, computes a response according to the protocol specification and decision on whether to accept or reject yet, and returns them to the adversary $\mathcal{A}$. If $\Pi^i_{U_u,U_v}$ has either accepted with some session key or terminated, this will be made known to $\mathcal{A}$. Note that if $m = *$, then this will result in the instantiation of the oracle $\Pi^i_{U_u,U_v}$ if such an oracle has not been created previously.

Session-Key Reveal(U_u, U_v, i) **query.** Any oracle, $\Pi^i_{U_u,U_v}$, upon receiving such a query and if $\Pi^i_{U_u,U_v}$ has accepted and holds some session key, will send this session key back to $\mathcal{A}$. As Blake-Wilson, Johnson and Menezes [3] have indicated, the Reveal query is designed to capture this notion. Note that this query is known as a Reveal(U_u, U_v, i) query in the BR model [2].

Session-State Reveal(U_u, U_v, i) **query.** The oracle, $\Pi^i_{U_u,U_v}$, upon receiving such a query and if $\Pi^i_{U_u,U_v}$ has neither accepted nor held some session key, will return all its internal state to $\mathcal{A}$. This includes any ephemeral parameters but not long-term secret parameters.

Corrupt(U_u) **query.** This query captures unknown key share attacks and insider attacks. This query allows $\mathcal{A}$ to corrupt the principal U_u at will, and thereby learn the complete internal state of the corrupted principal. Notice that a Corrupt query does not result in the release of the session keys since $\mathcal{A}$ already has the ability to obtain session keys through Reveal queries.

Protocols proven secure in the model that allows the Corrupt query are also proven secure against the unknown-key share attack. That is, if a key is to be shared between some parties, U_1 and U_2, the corruption of some other (non-related) player in the protocol, say U_3, should not expose the session key shared between U_1 and U_2 [13].

$\mathsf{Test}(U_u, U_v, i)$ **query.** This query is the only oracle query that does not correspond to any of $\mathcal{A}$'s abilities or any real-world event. If $\Pi^i_{U_u,U_v}$ has accepted with some session key and is being asked a $\mathsf{Test}(U_u, U_v, i)$ query, then depending on a randomly chosen bit, b, $\mathcal{A}$ is given either the actual session key or a session key drawn randomly from the session key distribution. Informally, $\mathcal{A}$ succeeds if $\mathcal{A}$ can guess the bit b.

A.1.2 Partnership Partnership in the CK model is defined using the notions of matching sessions and SIDs, as described in Definition 4. There is no formal definition of how SIDs should be defined. In practice, SIDs may be determined during protocol execution.

Definition 4 (Matching Sessions [7]). *Two sessions are said to be matching if they have the same session identifiers and corresponding partner identifiers.*

A.1.3 Security Model Security in the model is defined using the game $\mathcal{G}$, played between a malicious adversary $\mathcal{A}$ and a collection of $\Pi^i_{U_u,U_v}$ oracles for players U_u, U_v and instances i. $\mathcal{A}$ runs the game $\mathcal{G}$, with the following settings.

Stage 1: $\mathcal{A}$ is able to send any oracle queries at will.
Stage 2: At some point during $\mathcal{G}$, $\mathcal{A}$ will choose a fresh session on which to be tested and send a Test query to the fresh oracle associated with the test session. Note that the test session chosen must be fresh. Depending on a randomly chosen bit b, $\mathcal{A}$ is given either the actual session key or a session key drawn randomly from the session key distribution.
Stage 3: $\mathcal{A}$ continues making any oracle queries at will but cannot make $\mathsf{Corrupt}$ and/or Reveal that trivially expose the test session key.
Stage 4: Eventually, $\mathcal{A}$ terminates the game simulation and outputs a bit b', which is its guess of the value of b.

The success of $\mathcal{A}$ in $\mathcal{G}$ is quantified in terms of $\mathcal{A}$'s advantage in distinguishing whether $\mathcal{A}$ receives the real key or a random value. $\mathcal{A}$ wins if, after asking a $\mathsf{Test}(U_u, U_v, i)$ query, where $\Pi^i_{U_u,U_v}$ is fresh and has accepted, $\mathcal{A}$'s guess bit b' equals the bit b selected during the $\mathsf{Test}(U_u, U_v, i)$ query.

Let the advantage function of $\mathcal{A}$ be denoted by $\mathsf{Adv}^{\mathcal{A}}(k)$, where $\mathsf{Adv}^{\mathcal{A}}(k) = |2 \times Pr[b = b']| - 1$. We now define security in the model as described in Definition 5.

Definition 5 (Security). *A protocol is secure in the CK model if,*

Validity. *If two uncorrupted oracles complete matching sessions, then both oracles must hold the same session key.*
Indistinguishability. *For all probabilistic, polynomial time adversaries, $\mathcal{A}$, the advantage of $\mathcal{A}$, $\mathsf{Adv}^{\mathcal{A}}(k)$, in game $\mathcal{G}$ is negligible.*

A.2 More Number Theoretic Assumptions

Definition 6 (Modified (Computational) Bilinear Diffie-Hellman (MBDH) Problem [16]). *Given $(P, aP, bP, cP, c^{-1}P)$, output $\hat{e}(P, P)^{abc} \in G_2$.*

MBDH problems (both computational and decisional) are proposed in [16] to realize an ID-based signcryption scheme with forward-secrecy. In this paper, we use it in the form of interactive MBDH assumption, which is defined in a similar way as Definition 1 to support KGC forward-secrecy. Similar to Lemma 1, we have the following result.

Lemma 2 (Interactive MBDH Game Assumption) *For any adversary with PPT algorithm* $(\mathcal{A}_1, \mathcal{A}_2)$ *with advantage* $\epsilon(k)$ *to win the interactive MBDH game, there exists an algorithm that solves MBDH problem with probability* $\epsilon(k)^2$.

The key idea to prove the above result (due to Cheng *et al.* [10] in proving Lemma 1) is that the same X will be returned by the challenger whenever the same problem instance $(P, aP, bP, cP, c^{-1}P)$ is received. We can program $\mathcal{A}_2$ with input (r_2, h, σ) for the first time and $(r_2, h-1, \sigma)$ for the second time to extract the solution of MBDH problem.

B Security Proofs for Proposed Signature Schemes

Borrowing the idea of the forgery game in [18], we define below the existential unforgeability against adaptive-chosen-message-and-identity-attack of our pairing challenge-response signature scheme.

To simplify the discussion, we first describe a simplified variant of pairing challenge response signature without validity checking of the challenge and the user of private key for verification; and we prove its security by building a BDH solver $\mathcal{S}$ from a forger $\mathcal{F}$.

The unforgeability of the strong scheme in Section 4.2 can be proven in a similar manner by relying on another intractability assumption. Finally, we discuss how to extend the security notion of the basic scheme to that of the dual signature and show its security. Savvy readers will find the below proof and the proof for our key agreement protocols share many similarities.

B.1 Unforgeability of Identity-based Challenge-Response Signatures

Definition 7 (EUF-PCR-CMIA2). *For an ID-based challenge-response signature scheme, the existential unforgeability against adaptive-chosen-message-and-identity-attack is defined in the following game between a challenger* $\mathcal{S}$ *and an adversary* $\mathcal{F}$*:*

1. *System parameters generated according to the* Setup *procedure (excluding the* master secret*) and the challenge* W^* *is given to* $\mathcal{F}$.
2. $\mathcal{F}$ *can adaptively choose a polynomial number of oracle queries and ask* $\mathcal{S}$ *for answers.*
   (a) *Extraction oracle* $\mathcal{XO}$*: Given an identity* ID*, the corresponding private key* S_{ID} *is returned.*
   (b) *Signing oracle* $\mathcal{SO}$*: Given a challenge-identity-message tuple* (W, ID, m)*, the corresponding signature* σ *is returned.*
3. $\mathcal{F}$ *either halts with* fail*, or outputs a signature-identity-message tuple* $(\sigma^*, \mathsf{ID}^*, m^*)$ *(for verifier* ID'*).*
4. $\mathcal{F}$ *wins if all of the following conditions hold.*
   (a) *The signature* σ *is a valid challenge-response signature of* ID^* *on message* m^* *with respect to challenge* W^* *(for verifier* ID'*).*
   (b) *The identity* ID^* *(*not ID'*) did not appear in any extraction oracle* $\mathcal{XO}$ *query.*
   (c) *The pair* $(W^*, \mathsf{ID}^*, m^*)$ *did not appear in any signing oracle* $\mathcal{SO}$ *query (for verifier* ID'*).*

B.2 Basic Scheme

Setup, Extract: Same as the strong scheme described in Section 4.2.

Sign: We suppose B is the verifier requesting for a signature on message m from the signer A.

1. B picks $b \in_R \mathbb{Z}_q^*$ and sends $W_B = bP$ to A.
2. A picks $a \in_R \mathbb{Z}_q^*$ and sends $W_A = aQ_A$ and $e_A = \hat{e}((a + h_A)S_A, W_B)$ to B where $h_A = \mathcal{H}_0(W_A, m)$.

Verify: As the strong scheme, bP_{pub} is used instead of bS_B.

B.3 Proof for Unforgeability

Given a forger $\mathcal{F}$ of our basic scheme, below show how to build a BDH solver $\mathcal{S}$.

Setup : Suppose the problem instance is (P, xP, yP, zP), the public key of the KGC is set as xP and the challenge W^* is set as zP.

$\mathcal{H}$ *queries* : If an $\mathcal{H}$ query is previously asked, then the stored answer in the list $L_{\mathcal{H}}$ will be returned. Suppose the J^{th} distinct $\mathcal{H}$ query is ID_J, then $\mathcal{S}$ responses with yP; otherwise, $\mathcal{S}$ chooses $r_i \in_R \mathbb{Z}_q^*$, stores it in the list $L_{\mathcal{H}}$ along with ID_I, and outputs r_iP.

$\mathcal{H}_0$ *queries* : $\mathcal{S}$ maintains a list $L_{\mathcal{H}_0}$ to ensure that previously asked queries will receive the same answer. However, special value may be plugged into the list in the simulation of the $\mathcal{SO}$ queries with ID_I as the signer.

$\mathcal{XO}$ *queries* : The simulation fails (event 0) if the request is ID_J, otherwise the corresponding r_i is retrieved from the list $L_{\mathcal{H}}$ and $r_i(xP)$ is returned.

$\mathcal{SO}$ *queries* : Suppose W_V is received as the challenge and the designated verifier is ID_V. For any signing query of ID_I, $\mathcal{S}$ knows the corresponding private key, so the simulation can be done as a typical protocol invocation. Except for the following special handling for ID_J, α and h are chosen randomly from $\mathbb{Z}_q^*$ and setting $h = \mathcal{H}_0(\alpha P - hQ_J, m)$. If $\mathcal{H}_0(\alpha P - hQ_J, m)$ is previously queried, another α is chosen. The signature (W_J, e_J) can be computed by $W_J = \alpha P - hQ_J$ and $e_J = \hat{e}(\alpha xP, W_V)$. It is easy to see the signature is valid since $\hat{e}(x(W_J + hQ_J), W_V) = \hat{e}(\alpha xP, W_V)$.

Forgery : Suppose $\mathcal{F}$ does not halt, now $\mathcal{S}$ returns $\sigma^* = (W_J^*, e_J^*)$ as a valid signature. If it is not made on behalf of ID_J, the simulation fails. Event 0 would not occur if ID_J were chosen by $\mathcal{F}$ as the target of attack, such choice is made with probability $1/N_{\mathcal{Q}}$ where $N_{\mathcal{Q}}$ is the number of $\mathcal{H}$ queries.

Suppose the simulation does not abort we have $e_J^* = \hat{e}(W_J^* + h_J^* yP, zP)^x$ where $h_J^* = \mathcal{H}_0(W_J^*, m^*)$. We ignore the small probability that $\mathcal{F}$ can correctly guess the value of $\mathcal{H}_0(W_J^*, m^*)$ without making the corresponding $\mathcal{H}_0$ query. Now $\mathcal{S}$ runs $\mathcal{F}$ for a second time with the same settings except setting the response of $\mathcal{H}_0$ query of (W_J, m^*) as h'_J. By the standard forking lemma argument [23], in this second time $\mathcal{F}$ gives a valid forgery with $e'_J = \hat{e}(W_J^* + h'_J yP, zP)^x$. The solution of the BDH problem is given by $(e'_J/e_J^*)^{(h'_J - h_J^*)^{-1}} = \hat{e}(yP, zP)^x$.

B.4 Strong Scheme

For our ID-based strong challenge-response signature, the simulator $\mathcal{S}$ needs to provide W'^* in addition to W^* which satisfy the relation $\hat{e}(W^*, W'^*) = \hat{e}(Q_B, Q_A)$ for certain users A and B. This requires the forgery game of our basic scheme to be modified such that the adversary can choose two identities A and B after simulator $\mathcal{S}$ outputted the system parameter. The security of our strong scheme depends on a variant of BDH problem of (P, xP, yP, zP), in which $yz^{-1}P$ is also included in the problem instance. It is easy to see that the proof for our basic scheme works equally well since a valid W'^* can be returned by returning $yz^{-1}P$, if $\mathcal{S}$ sets $\mathcal{H}(\mathsf{ID}_A) = yP$.

B.5 Dual Scheme

Signatures produced by the dual scheme is in the following form

$$(W_A = aQ_A, W_B = bQ_B, e = \hat{e}(Q_A, Q_B)^{s(a+h_A)(b+h_B)})$$

which can be seen as two signatures (W_A, e) and (W_B, e) of the underlying (single) scheme. If $h_A = \mathcal{H}_0(W_A, m_A)$ and $h_B = \mathcal{H}_0(W_B, m_B)$, the former signature is one produced by A on message m_A under the challenge $W_B + h_B Q_B$ while the later is one produced by B on message m_B under the challenge $W_A + h_A Q_A$. Our security notion for ID-based scheme can be naturally extended to our ID-based dual challenge-response signature's case. More precisely, in the forgery game, the signing oracle query should include one more parameter that is the message to be signed by the peer, and a successful forgery comes with an extra message that is chosen by the forger $\mathcal{F}$.

The simulation for our basic scheme still works in this modified forgery game, but the solution of the BDH problem should be computed in another way since the forgery comes with an extra component. With the notation in the proof for basic scheme and suppose the dual signature given by the forger $\mathcal{F}$ is the one by J and V on message m_J and m_V respectively; $e_J^* = \hat{e}(W_J^* + h_J^* yP, zP + h_V^* Q_V)^x$ where $h_J^* = \mathcal{H}_0(W_J^*, m_J^*)$ and $h_V^* = \mathcal{H}_0(W_V^*, m_V^*)$. The second run gives $e'_J = \hat{e}(W_J^* + h'_J yP, zP + h'_V Q_V)^x$ where $h'_J = \mathcal{H}_0(W_J^*, m_J^*)$ and $h'_V = \mathcal{H}_0(W_V^*, m'_V)$. Note that forger $\mathcal{F}$ may give another message m'_V in the second run since it is something which is under $\mathcal{F}$'s control. e'_J/e_J^* gives $\hat{e}((h'_J - h_J^*)yP, zP)^x \cdot \hat{e}((h'_J - h_J^*)yP, h'_V Q_V)^x / \hat{e}((h_J^* - h_J^*)yP, h_V^* Q_V)^x$. Notice that Q_V is in the form of $r_V P$, so xQ_V can be computed as $r_V(xP)$ by simulator $\mathcal{S}$. Thus the last two term of the above expression can be cancelled out which leaves the term $\hat{e}((h'_J - h_J^*)yP, zP)^x$. BDH problem can be solved in a similar way as the proof for our basic scheme.

C Previously Unpublished Attack

Figure 3 describes Yi's protocol [34], $x(W)$ means the x-coordinate of point W.
Adversary $\mathcal{A}$ learns the session key of a fresh session of Yi's protocol as follows.

1. $\mathcal{A}$ asks a $\mathsf{Corrupt}$ query to C prior to the execution of the protocol – static corruption. $\mathcal{A}$ now runs as C.
2. A sends $W_A = aQ_A$ to B, which is intercepted by C.
3. C picks $r \in_R \mathbb{Z}_q^*$, computes $W_C = W_A + x(W_A)Q_A + rP$, and sends W_C to B impersonating A.
4. B responds with W_B as per protocol specification.
5. B computes the session key, $sk_B = \hat{e}(W_C + x(W_C)Q_C, (b + x(W_B))S_B)$.

Initiator A		Responder B
$(Q_A := \mathcal{H}(\mathsf{ID}_A), S_A := sQ_A)$		$(Q_B := \mathcal{H}(\mathsf{ID}_B), S_B := sQ_B)$
$a \in_R \mathbb{Z}_q^*; W_A := aQ_A$		
	$\xrightarrow{ID_A, W_A}$	$b \in_R \mathbb{Z}_q^*; W_B := bQ_B$
$h_A := x(W_A)$	$\xleftarrow{ID_B, W_B}$	$h_A := x(W_B)$
$sk_A = \hat{e}((a + x(W_A))Q_A, (b + x(W_B))Q_B)^s = sk_B$		

Fig. 3. Yi's identity-based key agreement protocol

6. Note that B and A are non-partners as the message received by B, W_C, is not being sent by A. Hence, the adversary is able to reveal B's session key with a Reveal query.
7. C forwards W_B to A impersonating B.
8. A computes the session key as $sk_A = \hat{e}((a + x(W_A))S_A, W_B + x(W_B)Q_B)$.
9. The adversary now computes the following.

$$
\begin{aligned}
&\frac{sk_B}{\hat{e}(rP_{pub} + x(W_C)S_C, W_B + x(W_B)Q_B)} \\
&= \frac{\hat{e}(W_C + x(W_C)Q_C, (b + x(W_B))S_B)}{\hat{e}(rP_{pub} + x(W_C)S_C, W_B + x(W_B)Q_B)} \\
&= \frac{\hat{e}(W_A + x(W_A)Q_A + rP + x(W_C)Q_C, (b + x(W_B))S_B)}{\hat{e}(rP + x(W_C)Q_C, W_B + x(W_B)S_B)} \\
&= \hat{e}(W_A + x(W_A)Q_A + rP + x(W_C)Q_C - rP - x(W_C)Q_C, (b + x(W_B))S_B) \\
&= \hat{e}(W_A + x(W_A)Q_A, (b + x(W_B))S_B) \\
&= \hat{e}((a + x(W_A))Q_A, (b + x(W_B))S_B) \\
&= \hat{e}((a + x(W_A))S_A, (b + x(W_B))Q_B) \\
&= \hat{e}((a + x(W_A))S_A, W_B + x(W_B)Q_B) \\
&= sk_A
\end{aligned}
$$