Efficient One-round Key Exchange in the Standard Model

Colin Boyd[1], Yvonne Cliff[1], Juan Gonzalez Nieto[1], and Kenneth G. Paterson[2]

[1] Information Security Institute,
Queensland University of Technology,
GPO Box 2434 Brisbane Qld 4001, Australia
y.cliff@isi.qut.edu.au, {c.boyd,j.gonzaleznieto}@qut.edu.au
[2] Information Security Group,
Royal Holloway University of London,
Egham, Surrey TW20 0EX, U.K.
Kenny.Paterson@rhul.ac.uk
May 7, 2008

Abstract. We consider one-round key exchange protocols secure in the standard model. The security analysis uses the powerful security model of Canetti and Krawczyk and a natural extension of it to the ID-based setting. It is shown how KEMs can be used in a generic way to obtain two different protocol designs with progressively stronger security guarantees. A detailed analysis of the performance of the protocols is included; surprisingly, when instantiated with specific KEM constructions, the resulting protocols are competitive with the best previous schemes that have proofs only in the random oracle model.
Keywords: key exchange, standard model.

1 Introduction

There has been a recent rapid growth of interest in efficient cryptographic primitives of all kinds that carry proofs in the standard model. Avoiding the random oracle model (ROM) or generic group model is to be preferred, given the known problems with instantiating these models in practice [9, 13, 3]. However, the usual price to be paid for working in the standard model is a loss of efficiency.

This paper initiates the systematic study of key exchange protocols whose security can be analyzed in the standard model. Our focus here is on two-party, one-round protocols — protocols in which only two message flows are required to securely establish a key between two parties. We provide two related, yet distinct, approaches to building such protocols using KEMs [1], both in the ID-based setting and the traditional PKI-based setting. Our security proofs use the Canetti-Krawczyk model (appropriately adapted for the identity-based case), which is sufficiently powerful to allow the capture of a variety of security properties including basic session key security, key compromise impersonation resistance, and various types of forward security.

In the identity-based setting, there is no shortage of protocols with security analysis in the ROM, with Chen, Cheng and Smart [11] providing a useful survey and comparison of these. Our protocols appear to be the first explicit constructions that are proven secure in the standard model in this setting. A recent preprint [32] also considers ID-based key exchange in the standard model, but the security analysis therein is incomplete – we comment in more detail on this below.

We consider the instantiation of our ID-based protocol designs with a variety of suitable concrete KEM components. These are derived from ID-based KEMs of Kiltz [20], Kiltz-Galindo [22] and Gentry [18]. By modifying these to operate in the setting of asymmetric pairings and ordinary elliptic curves, we are able to produce concrete ID-based protocols with security proven in the standard model that are only 2.5 times slower than the most

efficient protocols with security established in the ROM, the comparison being made on elliptic curves with a 128-bit security level.

In the PKI setting we also obtain efficient, one round, concrete protocol designs in the standard model, which compare favorably with the protocols of Jeong, Katz and Lee [19], Krawczyk [25] and Okamoto [30] which are to our knowledge the only one-round protocols secure in the standard model. The protocols are reasonably efficient even when compared to the best ROM protocols. For example, they can be instantiated with standard model KEMs to yield protocols with a computational increase of a factor around 3 when compared with HMQV [26].

Our first protocol design is the most efficient of the two, and provides key-compromise impersonation (KCI) resistance but not forward secrecy (FS). The basic idea of our first protocol design is very simple: the two parties simply send each other a random secret value using the IB-KEM and then use a randomness extractor to derive a session key from the combined secrets. Our second protocol design is based on the first, but adds an independent Diffie-Hellman exchange to achieve forward secrecy. It also achieves KCI resistance.

In the rest of this introduction we give an overview of related work. We then establish some essential definitions, and go on to outline the Canetti-Krawczyk model in which our proofs are presented. Section 4 of the paper develops our two protocol designs and proves their security in the Canetti-Krawczyk model. Section 5 compares the efficiency and security of the new protocols with the (ROM) ID-based protocol of Boyd, Mao and Paterson [8], one of the most efficient protocols in the identity-based literature. Section 6 compares the efficiency and security of the PKI-based version of the new protocols with the protocols of Jeong, Katz and Lee [19], Krawczyk [25] and Okamoto [30].

1.1 Related Work

Following the development of practical schemes for identity-based encryption [31, 7] many other identity-based primitives have been designed; due to their practical importance, these have included many key exchange protocols. Chen et al. [11] have provided a useful survey and comparison of work to date on identity-based key exchange.

Initially all security proofs for identity-based primitives relied on the random oracle model. More recently there has been a focus on providing new identity-based encryption (IBE) and identity-based key encapsulation (IB-KEM) schemes with security proofs in the standard model. Recent and quite efficient proposals include those of Waters [33], Kiltz [20], Gentry [18] and Kiltz–Galindo [22, 23].

Up until now, all proofs for identity-based key exchange protocols have continued to rely on the ROM, with the exception [32] noted. However, although Wang et al. [32] propose three protocols, a proof for only one is provided; the other two proofs supposedly use similar techniques. The protocol with a claimed proof applies a key derivation function H_2 to the shared secret, exchanged messages and identities. No properties of the key derivation function are stated or used in the proof; indeed the proof ignores the presence of H_2 altogether. However, without the key derivation function, the protocol is completely insecure, because it is based on the CPA (rather than CCA) version of Gentry's IB-KEM [18] and so has malleable messages. This malleability is easily exploited to find attacks which break the security of the protocol. The problems in the paper of Wang et al. [32] illustrate that it is not hard to devise ID-based protocols that look secure in the standard model but making the proofs work is not always so simple.

We note too that it is relatively straightforward to obtain standard-model secure key exchange protocols (in both settings) using the authenticator approach of Canetti-

Krawczyk [10] and Bellare-Canetti-Krawczyk [4], by working with standard-model-secure cryptographic primitives. The resulting protocols can be quite computationally efficient, but generally require more than one round of communication. A detailed study of such protocols is deferred to our future work.

In the normal public key model, Jeong et al. [19], proposed a protocol, called TS3, which is one-round and proven secure in the standard model[3]. TS3 is a Diffie-Hellman (DH) key exchange authenticated using a MAC keyed under the (static) DH of the long term keys of the two users. TS3 provides (weak) forward secrecy, but fails to achieve KCI resistance – a consequence of the static key used for authentication being the same for both parties. Interestingly, the ID-version of TS3, which is essentially the protocol of Boyd et al., appears to be limited to be only secure in the ROM. An ID-based version of TS3 secure in the standard model would imply a non-interactive ID-based key establishment protocol also secure in the standard model, which to date is not known. Even if we had such a primitive, the protocol would still not be KCI resistant.

More recently and closely related to our work, Okamoto [30] has proposed a one-round PKI-based protocol also secure in the standard model, which provides both (weak) forward secrecy and KCI resistance. The main advantage of our protocols over Okamoto's is that ours are generic. They can be instantiated using any combination of KEMs as long as they are CCA secure. Okamoto's protocol is highly specialised and the proof does not seem to generalise easily. Additionally Okamoto's key derivation function needs a non-standard notion of pseudo-random function security. Okamoto's proof is in the extended Canetti-Krawczyk (eCK) security model proposed by [27], while our proofs are on the Canetti-Krawczyk (CK) model, with the modifications by Krawcyzk [26] to capture KCI security and weak FS. The difference between the two models is rather subtle and will be discussed in Section 3.1. However, we remark now that contrary to Okamoto's statement in his paper, the eCK model is not stronger than the CK model, i.e. security in the eCK model does not imply security in the CK model. Furthermore it is arguable whether the eCK adversarial model is more realistic than the CK one.

2 Preliminaries

In this section we present standard definitions and results needed in the rest of the paper.

Definition 1 (Min-entropy [17, p.9]). *Let $\mathcal{X}$ be a probability distribution over A. The min-entropy of $\mathcal{X}$ is the value*

$$\text{min-ent}(\mathcal{X}) = \min_{x \in A: \Pr_{\mathcal{X}}[x] \neq 0}(-log_2(\Pr_{\mathcal{X}}[x])) \quad (1)$$

(Note that if $\mathcal{X}$ has min-entropy t then for all $x \in A$, $\Pr_{\mathcal{X}}[x] \leq 2^{-t}$.)

Definition 2 (Strong randomness extractor [15, p.42][29]). *A family of efficiently computable hash functions $\mathcal{H} = \{h_\kappa : \{0,1\}^n \rightarrow \{0,1\}^k | \kappa \in \{0,1\}^d\}$ is called a strong (m, ϵ)-randomness extractor, if for any random variable X over $\{0,1\}^n$ that has min-entropy at least m, if κ is chosen uniformly at random from $\{0,1\}^d$ and R is chosen uniformly at random from $\{0,1\}^k$, the two distributions $\langle \kappa, h_\kappa(X) \rangle$ and $\langle \kappa, R \rangle$ have statistical distance ϵ, that is*

[3] Interestingly, several errors in the proofs of Jeong, Katz and Lee have recently been corrected by the authors. Unlike in Wang et al.'s [32], there is no attack on their protocol, but still, it shows that getting the proofs right is non-trivial.

$$\frac{1}{2} \sum_{x \in \{0,1\}^k} |\Pr[h_\kappa(X) = x] - \Pr[R = x]| = \epsilon$$

To implement the randomness extraction function, one could apply the work of Chevassut *et al.* [12] to use a pseudo-random function as a randomness extractor.

Definition 3 (Pseudorandom Function Family (PRF)). *Let $\mathcal{F} = \{f_s\}_{s \in S}$ be a family of functions for security parameter $k \in \mathbb{N}$ and with seed $s \in S = S(k)$. Let $\mathcal{C}$ be an adversary that is given oracle access to either F_s for $s \in_R K$ or a truly random function with the same domain and range as the functions in $\mathcal{F}$. $\mathcal{F}$ is said to be* pseudorandom *if $\mathcal{C}$'s advantage in distinguishing whether it has access to a random member of $\mathcal{F}$ or a truly random function is negligible in k, for all polynomial-time adversaries $\mathcal{C}$. That is,*

$$\mathbf{Adv}^{\mathrm{p-rand}}_{\mathcal{F},\mathcal{C}}(k) = |\Pr[\mathcal{C}^{F_s(\cdot)}(1^k) = 1] - \Pr[\mathcal{C}^{\mathrm{Rand}(\cdot)}(1^k) = 1]|$$

is negligible in k.

Functions that are proven to be pseudorandom include CBC-MAC [5] (provided the underlying block cipher is a secure pseudorandom permutation family and the input length is constant) and HMAC [2] (provided the compression function is a PRF).

Assumption 1 (Decisional Diffie-Hellman (DDH)) *Let F be a cyclic group of order p' generated by an element f. Consider the set $F^3 = F \times F \times F$ and the following two probability distributions over it:*

$$\mathcal{R}_F = \{(f^a, f^b, f^c) \text{ for } a, b, c \in_R \mathbb{Z}_{p'}\} \tag{2}$$

and

$$\mathcal{DH}_F = \{(f^a, f^b, f^{ab}) \text{ for } a, b \in_R \mathbb{Z}_{p'}\} \tag{3}$$

We say the Decisional Diffie-Hellman (DDH) Assumption holds over $F = \langle f \rangle$ if the two distributions $\mathcal{R}_F$ and $\mathcal{DH}_F$ are indistinguishable by all polynomial-time adversaries $\mathcal{D}$. More precisely, for $k = |p'|$

$$\mathbf{Adv}^{\mathrm{ddh}}_{F,\mathcal{D}}(k) = |\Pr[\mathcal{D}(1^k, \rho) = 1 | \rho \in_R \mathcal{DH}_F] - \Pr[\mathcal{D}(1^k, \rho) = 1 | \rho \in_R \mathcal{R}_F]|$$

is negligible in k.

Definition 4 (ID-based KEM). *An IB-KEM $\mathcal{E} =$ (KeyGen, KeyDer, Enc, Dec) consists of four polynomial-time algorithms:*

- $(pk, \alpha) \in_{\mathrm{R}}$ KeyGen(1^k)*, given the security parameter $k \in \mathbb{N}$, returns a master public key, pk, and master secret key α;*
- $d_{id} \in_{\mathrm{R}}$ KeyDer(pk, α, id) *generates a private key corresponding to the identity id.*
- $(C, K) \in_{\mathrm{R}}$ Enc(pk, id) *outputs a key $K \in_R \mathbb{K}$ (the key space) and an encapsulation (ciphertext) C of the key under the identity id;*
- $K =$ Dec(pk, d_{id}, C) *outputs key K corresponding to the encapsulation C.*

Our definition of security for an identity-based key-encapsulation mechanism (IB-KEM) scheme is based upon that of Kiltz and Galindo [22].

Definition 5 (IB-KEM-CCA Security). *The security of an IB-KEM scheme $\mathcal{E} =$ (KeyGen, KeyDer, Enc, Dec) is defined using the following experiment.*

$$
\begin{array}{l}
\textbf{\textit{Experiment}}\ \mathbf{Exp}^{\mathrm{ib-kem-cca}}_{\mathcal{E},\mathcal{A}}(k) \\
(pk, \alpha) \in_{\mathrm{R}} \mathit{KeyGen}(1^k) \\
(id^*, state) \in_{\mathrm{R}} \mathcal{A}^{\mathcal{O}_{\mathit{KeyDer}}(\cdot),\mathcal{O}_{\mathit{Dec}}(\cdot,\cdot)}(\mathit{find}, pk) \\
K_0^* \in_{\mathrm{R}} \mathbb{K} \\
(C^*, K_1^*) \in_{\mathrm{R}} \mathit{Enc}(pk, id^*) \\
\gamma \in_{\mathrm{R}} \{0,1\} \\
K^* = K_\gamma^* \\
\gamma' \in_{\mathrm{R}} \mathcal{A}^{\mathcal{O}_{\mathit{KeyDer}}(\cdot),\mathcal{O}_{\mathit{Dec}}(\cdot,\cdot)}(\mathit{guess}, K^*, C^*, state) \\
\textit{If } \gamma \neq \gamma' \textit{ then return 0 else return 1}
\end{array}
$$

where the oracles and advantage of $\mathcal{A}$ are defined as follows:

$$
\begin{array}{l}
\mathcal{O}_{\mathit{KeyDer}}(id) = \mathit{KeyDer}(pk, \alpha, id)\ \textit{(where } id \neq id^*\textit{)} \\
\mathcal{O}_{\mathit{Dec}}(id, C) = \mathit{Dec}(pk, \mathit{KeyDer}(pk, \alpha, id), C)\ \textit{(where } id \neq id^*\ \textit{or } C \neq C^*\textit{)}
\end{array}
$$

The advantage of $\mathcal{A}$ in the above experiment is:

$$
\mathbf{Adv}^{\mathrm{ib-kem-cca}}_{\mathcal{E},\mathcal{A}}(k) = \left| 2\Pr\left[\mathbf{Exp}^{\mathrm{ib-kem-cca}}_{\mathcal{E},\mathcal{A}}(k) = 1\right] - 1 \right| .
$$

$\mathcal{E}$ is secure against adaptively-chosen ciphertext attacks if $\mathbf{Adv}^{\mathrm{ib-kem-cca}}_{\mathcal{E},\mathcal{A}}(k)$ is a negligible function in k for all polynomial-time adversaries $\mathcal{A}$.

3 Canetti-Krawczyk model

In this section the CK approach is reviewed. Further details of the model can be found in the original papers [4, 10].

In the CK model a protocol π is modeled as a collection of n programs running at different parties, $P_1, \ldots, P_n$. Each program is an interactive probabilistic polynomial-time (PPT) machine. Each invocation of π within a party is defined to be a *session*, and each party may have multiple sessions running concurrently. The communications network is controlled by an adversary $\mathcal{A}$, also a PPT machine, which schedules and mediates all sessions between the parties. When first invoked within a party, a key exchange protocol π calls an initialization function that returns any information needed for the bootstrapping of the cryptographic authentication functions. After this initialization stage, the party waits for activation. $\mathcal{A}$ may activate a party P_i in two ways:

1. by means of an $\mathsf{establish\text{-}session}(P_i, P_j, s)$ request, where P_j is another party with whom the key is to be established, and s is a session-id string which uniquely identifies a session between the participants.
2. by means of an *incoming message* m with a specified sender P_j.

Upon activation, the parties perform some computations, update their internal state, and may output messages together with the identities of the intended receivers. Two sessions (P_i, P_j, s) and (P_i', P_j', s') are said to be *matching sessions* if $P_i = P_j'$, $P_j = P_i'$, and $s = s'$, i.e. if their session-ids are identical and they recognised each other as their respective communicating partner for the session. In the analysis of the protocols in this paper, we define the session-id as the concatenation of the messages sent and received by the party. In addition to the activation of parties, $\mathcal{A}$ can perform the following queries:

1. $\mathsf{corrupt}(P_i)$. With this query $\mathcal{A}$ learns the long term key of P_i.
2. $\mathsf{session\text{-}key}(P_i, P_j, s)$. This query returns the session key (if any) accepted by P_i during a given session s with P_j.

3. session-state(P_i, P_j, s). This query returns all the internal state information of party P_i associated to a particular session s with P_j, but does not include the long term key of P_i.
4. session-expiration(P_i, P_j, s). This query is used for defining *forward secrecy* and erases from memory the session key on a completed session. The session is thereafter said to be expired.
5. test-session(P_i, P_j, s). To respond to this query, a random bit b is selected. If $b = 1$ then the session key is output. Otherwise, a random key is output chosen from the probability distribution of keys generated by the protocol. This query can only be issued to a session that has not been *exposed*. A session is exposed if the adversary performs any of the following actions:
   - a session-state or session-key query to this session or to the matching session, or
   - a corrupt query to either partner before the session expires at that partner.

Security is defined based on a game played by the adversary. In this game $\mathcal{A}$ interacts with the protocol. In a first phase of the game, $\mathcal{A}$ is allowed to activate sessions and perform corrupt, session-key, session-state and session-expiration queries as described above. The adversary then performs a test-session query to a party and session of its choice. The adversary is not allowed to expose the test session. $\mathcal{A}$ may then continue with its regular actions with the exception that no more test-session queries can be issued. Eventually, $\mathcal{A}$ outputs a bit b' as its guess on whether the returned value to the test-session query was the session key or a random value, then halts. $\mathcal{A}$ wins the game if $b = b'$. The definition of security follows.

Definition 6. *A key establishment protocol π is called* session key (SK-) secure *with perfect forward secrecy (PFS) if the following properties are satisfied for any adversary $\mathcal{A}$.*

1. *If two uncorrupted parties complete matching sessions then they both output the same key;*
2. *The probability that $\mathcal{A}$ guesses correctly the bit b is no more than $\frac{1}{2}$ plus a negligible function in the security parameter.*

We define the advantage of $\mathcal{A}$ to be

$$\mathbf{Adv}_{\mathcal{A}}^{\mathrm{sk}} = \left|2\Pr[b = b'] - 1\right|.$$

Hence the second requirement will be met if the advantage of $\mathcal{A}$ is negligible. Canetti and Krawczyk also provide a definition of SK-security *without PFS*. The only difference with respect to the above definition is that now the adversary is not allowed to expire sessions.

Krawczyk [26] showed that forward secrecy in the usual sense cannot be achieved in a two-pass protocol such as the ones that we consider. Therefore we restrict our concern to what Krawczyk calls *weak forward secrecy (WFS)*, in which the adversary is forbidden from taking an active part in the test session. We will also consider *partial WFS*, where we further restrict the adversary to corrupt at most one party to the test session. In the ID-based setting, WFS implies *key escrow freeness*, i.e. it protects against attacks in which the Key Generation Centre, who knows all the long term keys of all the parties, tries to (passively) eavesdrop in the communications of any two parties.

The original CK model does not consider *key compromise impersonation* (KCI) attacks, where the adversary, after compromising the long-term key of a party A, engages in a

successful protocol run with A posing as a third party B, i.e. A accepts a session key in the belief that it is shared with B, when in fact is shared with the adversary. Thus in a KCI attack there is no matching session to the test session. To model KCI resistance for our protocols we modify the definition of security to allow the adversary to corrupt the owner A of the test session (A, B, s).

3.1 Difference between the eCK and CK models

The essential difference between the eCK as defined in [27] and the CK model is that the eCK model substitutes the session-state query with a new ephemeral-key reveal query, which reveals the randomness used in the specified session. In addition, a new definition of freshness is formulated in the eCK model that permits ephemeral-key reveal queries on the test session. The important point to note is that the ephemeral-key does not include session state that has been computed using the long-term secret of the party. This is not the case in the CK model where, in principle, the adversary is allowed access to all the inputs (including the randomness, but excluding the long-term secret itself) and the results of all the computations done by a party as part of a session. In other words, there is information that is available to the adversary in the eCK model that is not available in the CK model, but the same is true in the opposite direction. Thus, strictly speaking, the security afforded by the two models cannot be compared.

Furthermore, it is arguable whether the differences between the two models are meaningful in reality. An attack where the adversary can obtain the randomness but not the rest of the session state seems rather special. In any case, we can generically defend against that type of attack by "pseudo-randomising" the random input using the long term secret, which is what protocols such as Okamoto's [30] and NAXOS [27] that have considered the eCK model do. That is, if r is the random input used by a session and s is the long term key of the party, the protocol first uses a pseudo-random function f to compute $r' = f_s(r)$, and uses r' instead of r. This way, even though r is available to the adversary through ephemeral-key reveal, r' remains protected.

4 Generic 2×KEM Protocols

In this section, we present Protocols 1 and 2, two generic protocols based on the use of any CCA-secure IB-KEM. The first, Protocol 1, is the most efficient of the two, and provides KCI resistance, but does not provide forward secrecy. The basic idea of Protocol 1 is very simple: the two parties simply send each other a random secret value using the IB-KEM and then derive a session key from the combined secrets using a randomness extractor and expander. Protocol 2 adds an independent Diffie-Hellman exchange in a group generated by f to achieve (weak) forward secrecy. It also achieves KCI resistance. The description of both protocols for the PKI-based setting is the same except that the identities are substituted with the public keys of the parties.

The protocol messages and actions are symmetrical for the parties in our protocols. It is assumed that the IB-KEM is defined to output a random key if a ciphertext is not valid. Because the protocols complete in one round, the actual order in which the two parties A and B exchange their messages is irrelevant. In the descriptions provided we let A be the one party such that $id_A < id_B$, using some agreed order relation, e.g. lexicographic order.

In defining the session id s we have assumed that the randomness expander is able to accept inputs at least as long as s. If this were not the case, a collision resistant hash function can be used in order to shorten the length of the input to the expander. Our security analysis can be easily modified to accept this change.

Note that each party must check that the identity of the party in its incoming message is actually the identity of its intended partner. Furthermore, the decapsulated IB-KEM key must be securely erased in the same activation in which it is decapsulated. Thus we are making the restriction that session-state reveal queries do not return decapsulated keys. Note however that once the key is decapsulated it can be inmediately used to compute the session key, so there is no need to store decapsulated keys. This restriction is critical, otherwise the protocol can be trivially broken by the adversary, as follows. Let (A, C_A^*, B, C_B^*) be the transcript of an observed protocol run that the adversary $\mathcal{A}$ seeks to compromise. $\mathcal{A}$ initiates a new session with B by sending D, C_A^* to B, i.e. $\mathcal{A}$ pretends to be D and replays the target ciphertext C_A^*. The adversary could then issue a session-state reveal for the new session to B, thus obtaining the decryption C_A^*. Using the same strategy with A, the adversary could find out the decryption of C_B^*, which would then allow the adversary to compute the session key corresponding to the session (A, C_A^*, B, C_B^*). We emphasize that all other session state can be revealed as part of a session-state query, in particular, encapsulated keys (at the party that generated them) and DH exponentials. Despite of this, in the description of our protocols, we explicitly ask for all intermediate state to be erased once the session key is computed. It would seem artificial to specify that only the decapsulated key be deleted, when there is no need to store anything apart from the session key and the session id.

Interestingly, Protocol 2 does not require that the parties check group membership of the Diffie-Hellman exponentials Y_A and Y_B. This is because the security of the protocol does not depend on them except for proving weak forward secrecy, where the adversary is passive, in which case these values are assumed to be correctly generated. We can intuitively see that the security of Protocol 2 is independent of the Diffie-Hellman exchange when the adversary does not corrupt the owners to the test session. To do so, let us assume that the adversary is able to somehow choose the values Y_A and Y_B itself (but the rest of the protocol is executed by the parties normally). The session key is computed as $\mathsf{Expd}_{K''_B}(s) \oplus \mathsf{Expd}_{K''_A}(s) \oplus \mathsf{Expd}_{K''_{AB}}(s)$. The adversary effectively chooses the subkey K''_{AB}, thus the goal of the adversary is reduced to distinguishing $\mathsf{Expd}_{K''_B}(s) \oplus \mathsf{Expd}_{K''_A}(s)$ from random. This is same goal as a that of an adversary against Protocol 1, the difference being that while in Protocol 1, s is fixed for A, C_A, B, C_B, here s depends also on Y_A and Y_B. However a crucial property of Protocol 2 is that each different choice of Y_A, Y_B defines a different session id s, therefore $\mathsf{Expd}_{K''_A}(s) \oplus \mathsf{Expd}_{K''_B}(s)$ will also be pseudo-random across different sessions, even if K''_A and K''_B are fixed.

For the same reason, Protocol 2 is immune to malleability attacks where an active adversary tries to take advantage of the malleability of the Diffie-Hellman key-exchange part of the protocol. An example of such attack is as follows. A sends A, C_A, Y_A to B, who outputs B, C_B, Y_B. The adversary intercepts the latter message and changes it to B, C_B, Y_B^r where r is chosen by the adversary. Thus the key as computed by B is $K_B = \mathsf{Expd}_{K''_B}(s) \oplus \mathsf{Expd}_{K''_A}(s) \oplus \mathsf{Expd}_{K''_{AB}}(s)$, whereas A computes $K_B = \mathsf{Expd}_{K''_B}(\bar{s}) \oplus \mathsf{Expd}_{K''_A}(\bar{s}) \oplus \mathsf{Expd}_{(K''_{AB})^r}(\bar{s})$. Even though the Diffie-Hellman subkeys K''_{AB} and $(K''_{AB})^r$ are related, the two session keys are indistinguishable from random to the adversary.

In the description of both protocols we have assumed that the same public parameters are used by both parties. This is however not necessary. Each party could be using different public parameters, IB-KEMs, or even one party could be using a IB-KEM and the other a PKI-based KEM.

A		B
$(C_A, K'_A) \in_{\mathrm{R}} \mathsf{Enc}(pk, id_B)$		$(C_B, K'_B) \in_{\mathrm{R}} \mathsf{Enc}(pk, id_A)$
	$\xrightarrow{A, C_A}$	
	$\xleftarrow{B, C_B}$	
$K'_B = \mathsf{Dec}(pk, d_{id_A}, C_B)$		$K'_A = \mathsf{Dec}(pk, d_{id_B}, C_A)$
$K''_A = \mathsf{Exct}_\kappa(K'_A); K''_B = \mathsf{Exct}_\kappa(K'_B)$		$K''_B = \mathsf{Exct}_\kappa(K'_B); K''_A = \mathsf{Exct}_\kappa(K'_A)$
$s = A\|\|C_A\|\|B\|\|C_B$		$s = A\|\|C_A\|\|B\|\|C_B$
$K_A = \mathsf{Expd}_{K''_A}(s) \oplus \mathsf{Expd}_{K''_B}(s)$		$K_B = \mathsf{Expd}_{K''_B}(s) \oplus \mathsf{Expd}_{K''_A}(s)$
Erase all state except (K_A, s)		Erase all state except (K_B, s)
'Established (A, B, s, K_A)'		'Established (B, A, s, K_B)'

Protocol 1: Generic 2×KEM

A		B
$y_A \in_{\mathrm{R}} \mathbb{Z}^*_{p'}; Y_A = f^{y_A}$		$y_B \in_{\mathrm{R}} \mathbb{Z}^*_{p'}; Y_B = f^{y_B}$
$(C_A, K'_A) \in_{\mathrm{R}} \mathsf{Enc}(pk, id_B)$		$(C_B, K'_B) \in_{\mathrm{R}} \mathsf{Enc}(pk, id_A)$
	$\xrightarrow{A, C_A, Y_A}$	
	$\xleftarrow{B, C_B, Y_B}$	
$K'_B = \mathsf{Dec}(pk, d_{id_A}, C_B)$		$K'_A = \mathsf{Dec}(pk, d_{id_B}, C_A)$
$K''_A = \mathsf{Exct}_\kappa(K'_A); K''_B = \mathsf{Exct}_\kappa(K'_B)$		$K''_B = \mathsf{Exct}_\kappa(K'_B); K''_A = \mathsf{Exct}_\kappa(K'_A)$
$K''_{AB} = \mathsf{Exct}_\kappa(Y_B^{y_A})$		$K''_{BA} = \mathsf{Exct}_\kappa(Y_A^{y_B})$
$s = A\|\|C_A\|\|Y_A\|\|B\|\|C_B\|\|Y_B;$		$s = A\|\|C_A\|\|Y_A\|\|B\|\|C_B\|\|Y_B;$
$K_A = \mathsf{Expd}_{K''_A}(s) \oplus \mathsf{Expd}_{K''_B}(s)$		$K_B = \mathsf{Expd}_{K''_B}(s) \oplus \mathsf{Expd}_{K''_A}(s)$
$\oplus \mathsf{Expd}_{K''_{AB}}(s)$		$\oplus \mathsf{Expd}_{K''_{BA}}(s)$
Erase all state except (K_A, s)		Erase all state except (K_B, s)

Protocol 2: Generic 2×KEM + Diffie-Hellman

We are now in a position to state the security theorems for the two protocols. For both these theorems we use the following notation:

- $\{\mathsf{Expd}_K(\cdot)\}_{K \in \mathbb{U}_1} : \{0,1\}^\sigma \to \mathbb{U}_2$ is a pseudorandom function family, (as described in Definition 3),
- $\mathsf{Exct}_\kappa(\cdot) : \mathbb{K} \to \mathbb{U}_1$ is chosen uniformly at random from a strong (m, ϵ)-strong randomness extractor for appropriate m and ϵ (as described in Definition 2),
- n_{orac} is the total number of oracles (i.e. sessions) created by $\mathcal{B}$ against the protocol, and
- $\frac{1}{p}$ is the maximum probability that $C_1 = C_2$ where $(C_1, K_1) \in_{\mathrm{R}} \mathsf{Enc}(pk, id)$ and $(C_2, K_2) \in_{\mathrm{R}} \mathsf{Enc}(pk, id)$ for any identity (if $C_1 = C_2$ then $K_1 = K_2$ also since both ciphertexts decrypt to the same value).

Theorem 1. *Let $\mathcal{B}$ be any adversary against Protocol 1. Then the advantage of $\mathcal{B}$ against the SK-security (with partial WFS and KCI resistance) of Protocol 1 is:*

$$\mathbf{Adv}^{\mathrm{sk}}_{\mathcal{B}}(k) \leq \frac{n^2_{\mathrm{orac}}}{p} + 2n_{\mathrm{orac}} \left(\mathbf{Adv}^{\mathrm{ib-kem-cca}}_{\mathcal{E},\mathcal{A}}(k) + \epsilon + \mathbf{Adv}^{\mathrm{p-rand}}_{\mathcal{F},\mathcal{C}}(k) \right)$$

Theorem 2. *Let $\mathcal{B}$ be any adversary against Protocol 2. Then the advantage of $\mathcal{B}$ against the SK-security (with WFS and KCI resistance) of Protocol 2 is:*

$$\mathbf{Adv}^{\mathrm{sk}}_{\mathcal{B}}(k) \leq \max\left(2n^2_{\mathrm{orac}}\mathbf{Adv}^{\mathrm{ddh}}_{F,\mathcal{D}}(k) + 2\epsilon + 2\mathbf{Adv}^{\mathrm{p-rand}}_{\mathcal{F},\mathcal{C}}(k)\ ,\right.$$
$$\left.\frac{n^2_{\mathrm{orac}}}{p} + 2n_{\mathrm{orac}}\left(\mathbf{Adv}^{\mathrm{ib-kem-cca}}_{\mathcal{E},\mathcal{A}}(k) + \epsilon + \mathbf{Adv}^{\mathrm{p-rand}}_{\mathcal{F},\mathcal{C}}(k)\right)\right).$$

The proofs of Theorems 1 and 2 can be found in Appendix C. We remark that, despite the simplicity of the protocols, proving their security turns out to be less simple than one might expect.

4.1 On Alternative Session Key Derivation

When a particular KEM such as those of Kiltz [20] and Kiltz-Galindo [22] (shown in Appendix A) is used in conjunction with the protocol, it may be tempting to try to achieve forward secrecy by treating K'_A and K'_B as Diffie-Hellman values and deriving K''_A and K''_B by combining K'_A and K'_B in the usual Diffie-Hellman manner and then extracting the randomness from this combined key. However, such a modification invalidates the proof, since the adversary $\mathcal{A}$ against the KEM would no longer be able to generate the proper test session key when $\mathcal{B}$, the adversary against the protocol, generates the input to the test session. The problem could be overcome by adding authentication to the messages, but this would increase the number of messages and rounds required by the protocol.

Similarly, when K'_A, K'_B and $Y_B^{y_A}$ are all elements of the same algebraic group, it may be tempting to set $K''_A = \mathsf{Exct}_\kappa(K'_A K'_B Y_B^{y_A})$, and changing K''_B similarly. This way, depending in the actual KEM used, it may be possible to optimize the computation of the product $K'_A K'_B Y_B^{y_A}$, using multi-exponentiation algorithms [6]. Again, such a modification will invalidate the proof. Assuming the test session owner is A and the intended partner B, then the test session key returned to $\mathcal{B}$ would be $\mathsf{Expd}_{\mathsf{Exct}_\kappa(K'_A K'_B Y_B^{y_A})}(s')$ where $\mathcal{B}$ may know $K'_B Y_B^{y_A}$. However, if $\mathcal{B}$ uses the test session output C_A as input to other sessions with B, $\mathcal{B}$ may learn values of the form α_i, $\mathsf{Expd}_{\mathsf{Exct}_\kappa(K'_A \alpha_i)}(s'_i)$ for many i ($\mathcal{B}$ can find out α_i if it controls the the other party involved in the protocol run with B). Unfortunately, providing such values to $\mathcal{B}$ no longer fits with the definition of the randomness extractor, even if it is extended to allow multiple uses. To overcome this we need a different key derivation function. In particular we could use the key derivation function defined by Okamoto in his recent protocol [30], which requires a *pseudo-random function with pair-wise independent random sources*. This is however a non-standard assumption, stronger than the pseudo-randomenss of Definition 3. In any case, such key derivation function is only useful for some KEMs (when K'_A, K'_B and $Y_B^{y_A}$ are all in the same group and the product $K'_A K'_B Y_B^{y_A}$ can be optimised).

5 Protocol Comparison: ID-based case

We now compare Protocols 1 and 2 with that of Boyd et al. [8] (BMP) which is one of the most efficient listed by Chen et al. in their survey of protocols [11, Table 6]. Unlike our protocols, which consist of two passes and a single round and only provide implicit authentication, BMP is a three-round three-pass protocol and provides explicit authentication. Thus it may appear that BMP is not a good choice to compare our protocols with. However if desired, our protocols can be modified to provide explicit authentication using

the well-known key confirmation method discussed by Krawczyk [26, Section 8] at a cost of an extra pseudo-random function computation per party and the addition of a third message.

Table 1 summarises the properties of the protocols under consideration. The costs per party given in the table for Protocols 1 and 2 assume the use of Kiltz's IB-KEM. We note that the BMP protocol does not have a proof of security in the standard model, unlike Protocols 1 and 2. Protocol 2 is the only one for which we have been able to prove both weak forward security (FS) and KCI resistance in the standard model.

	weak FS	KCI	Standard Model	Cost per party
Protocol 1	✘	✓	✓	56
Protocol 2	✓	✓	✓	59
BMP [8]	✓	✘	✘	23

Table 1. Security and efficiency comparison (IB setting)

To compare the efficiency of the protocols we use the costs per operation provided by Chen et al. [11] for Type 3 pairings with a security parameter of 128, which are the most efficient type of pairings for security levels higher than 80 bits. The values are shown in Table 2, which also shows the costs of Kiltz, Kiltz-Galindo and Gentry IB-KEMs. These figures require 256 bits to represent an element of $\mathbb{G}_1$, 512 bits to represent an element of $\mathbb{G}_2$, and 3072 bits to represent an element of $\mathbb{G}_T$. As suggested by Chen et al., we assume that all elements of the ciphertext are checked to determine that they lie in the correct subgroup to avoid attacks such as the small subgroup attack.

All of these IB-KEMs were originally proposed to use Type 1 pairings, and so to obtain the costs we have had to convert the three IB-KEMs to work with Type 3 pairings. The modified schemes can be found in Appendix A together with a discussion on their security and efficiency.

In BMP each party sends only one element of $\mathbb{G}_1$ to the other, so the bandwidth is smaller than using Kiltz's IB-KEMs with Protocol 1. Each party computes one pairing and two exponentiations in $\mathbb{G}_1$, as well as a subgroup check of one element in $\mathbb{G}_1$. Therefore the total cost per party is 23 time units, as opposed to the 56 units for the Kiltz IB-KEM with Protocol 1. This means that we have achieved identity based key exchange in the standard model in less than 2.5 times the cost in the random oracle model using the size of curve given above. Given the better security guarantees of the standard model, this extra cost may be considered quite reasonable.

The efficiency of Protocol 2 will be worse than that of Protocol 1, but depending on the choice of the group $\langle f \rangle$, it may not be much worse. For example, if the DDH assumption holds in $\mathbb{G}_1$ (this will require $\mathbb{G}_1 \neq \mathbb{G}_2$ and no efficiently computable homomorphism from $\mathbb{G}_1$ to $\mathbb{G}_2$), only 3 extra time units would be required per party (e.g. for party A, one to generate Y_A, one to perform a subgroup check on Y_B, and one to find $Y_B^{y_A}$). The increase in message size would be an extra 256 bits per message.

6 Protocol Comparison: PKI-based case

We now consider our two generic protocols in the traditional PKI-based setting and compare them with existing protocols. Table 3 shows the computational cost of Protocol 1 and

	Type 3 cost	Kiltz			Kiltz-Galindo			Gentry		
		Enc	Dec	KeyDer	Enc	Dec	KeyDer	Enc	Dec	KeyDer
$\mathbb{G}_1$ exp, multi-exp.	1, 1.5	-,1	-,1	2,-	1,1	1,1	1,-	-,1	-,-	-,-
$\mathbb{G}_2$ exp, multi-exp.	3, 4.5	1,-	1,-	1,-	1,-	2,-	1,-	-,-	-,1	-,3
$\mathbb{G}_T$ exp, multi-exp.	3, 4.5	1,-	-,-	-,-	1,-	-,-	-,-	3,1	-,1	-,-
Pairing	20	-	2	-	-	3	-	-	1	-
$\mathbb{G}_1$ subgroup check	1	-	1	-	-	2	-	-	1	-
$\mathbb{G}_2$ subgroup check	3	-	1	-	-	1	-	-	-	-
$\mathbb{G}_T$ subgroup check	4	-	-	-	-	-	-	-	3	-
Total cost		7.5	48.5	5	8.5	73.5	4	15	42	13.5
Total Enc + Dec cost		56			82			57		

Table 2. Costs of IB-KEMs using Type 3 pairings

2 when instantiated with the recently proposed KEMs of Kiltz [21] and Okamoto [30]. The efficiency of these two KEMs is shown in Table 4. The computational cost figures of both Table 3 and 4 include the cost of performing group membership tests (1 exponentiation per test) and distinguishes regular exponentiations from multi-exponentiations. However we ignore "half-exponentiations" that maybe possible when exponents are the outputs of hash functions. We stress that the shown computational costs are only rough indicative figures. The exact computational costs depend on actual choices of groups. We see that Kiltz's KEM is more efficient than Okamoto's by one regular exponentiation in the decapsulation algorithm. Kiltz's KEM security is based on the Gap Hashed Diffie-Hellman (GHDH) problem, while Okamoto's is based on the DDH problem and the existence of pseudo-random functions with pair-wise independent random sources (πPRF).

Table 3 also shows the costs of the protocols due to Jeong *et al.* [19] and Okamoto [30], which to our knowledge are the only one-round protocols whose security has been proven in the standard model. HMQV [26], whose security has only been shown in the random oracle model, is also included.

Jeong *et al.*'s protocol is the most efficient of all of the compared protocols, but does not provide KCI resistance. Protocol 1 instantiated with Kiltz's KEM results in the cheapest protocol with KCI resistance but only provides partial FS. Of the protocols providing both weak FS and KCI resistance in the standard model, Okamoto's protocol is the cheapest by one regular exponentiation. As discussed in Section 4.1, Okamoto's protocol can be seen as an instantiation of Protocol 2 with Okamoto's KEM but using a different key derivation function. We note that even though Okamoto's protocol is slightly more efficient than Protocol 2 instantiated with the currently most efficient KEM (Kiltz's KEM), Protocol 2 has the advantage of being generic. It is also possible that if a more efficient KEM is devised, then the generic Protocol 2 would be more efficient that Okamoto's. Note that Okamoto's key derivation function poses constraints on the KEM key space and hence cannot be applied generally to all KEMs.

Finally, we note that Protocol 2 is reasonably efficient when compared with HMQV. In its most optimised form (where there is no subgroup membership checking and considering short-exponents) HMQV requires around 2.2 exponentiations. We can roughly approximate 1 multi-exponentiation to 1.2 regular exponentiations [6], which makes the cost of Protocol 2-Kiltz 7.4 regular exponentiations.

	weak FS	KCI	Standard Model	Cost (exp, multi-exp)
Protocol 1 - Kiltz	✘	✓	✓	3,2
Protocol 1 - Okamoto	✘	✓	✓	4,2
Protocol 2 - Kiltz	✓	✓	✓	5,2
Protocol 2 - Okamoto	✓	✓	✓	6,2
Okamoto	✓	✓	✓	4,2
Jeong-Katz-Lee	✓	✘	✓	3,-
HMQV	✓	✓	✘	4,-

Table 3. Security and efficiency comparison (PKI setting)

	Enc (exp, multi-exp)	Dec (exp, multi-exp)	Security Assumption	Ciphertext (#group elements)
Kiltz	2,1	1,1	GHDH	2
Okamoto	2,1	2,1	DDH+πPRF	2

Table 4. Costs of KEMs

7 Conclusion

We have proven secure two generic protocols that may be used with any KEM to achieve secure key exchange in the standard model, in either the ID-based setting or the normal public key setting.

In addition, we provided a detailed analysis of the protocols' efficiency on Type 3 curves; this necessitated the extension of the IB-KEMs of Kiltz [20], Kiltz-Galindo [22] and Gentry [18] to use ordinary elliptic curves. We found that both our protocols take approximately 2.5 times as long as the protocol of Boyd, Mao, and Paterson [8] (which is only proven secure in the random oracle model) when both protocols are implemented on elliptic curves with a 128 bit security level.

The PKI versions of our protocols also compare favourably with the existing ones of Jeong *et al.* [19] and Okamoto [30]. Protocol 2 provides more security than Jeong's protocol and the same as Okamoto's. When instantiated with Kiltz' PKI-based KEM [21] Protocol 2 is slightly less efficient than Okamoto's. However, Protocol 2 has the advantage of being generic, i.e. it can be used together with any KEM which is CCA secure and our security analysis still applies.

References

1. M. Abe, R. Gennaro, K. Kurosawa, and V. Shoup. Tag-KEM/DEM: A new framework for hybrid encryption and a new analysis of Kurosawa-Desmedt KEM. In *Advances in Cryptology — EUROCRYPT 2005 Proceedings*, volume 3494 of *Lecture Notes in Computer Science*, pages 128–146. Springer, 2005. Full version at `http://eprint.iacr.org/2005/027`, revised October 11, 2006.
2. M. Bellare. New proofs for NMAC and HMAC: Security without collision-resistance. In *Advances in Cryptology—CRYPTO 2006*, volume 4117 of *Lecture Notes in Computer Science*, pages 602–619. Springer, 2006.
3. M. Bellare, A. Boldyreva, and A. Palacio. An uninstantiable random-oracle-model scheme for a hybrid-encryption problem. In *Advances in Cryptology — EUROCRYPT 2004 Proceedings*, volume 3027 of *Lecture Notes in Computer Science*, pages 171–188, 2004.
4. M. Bellare, R. Canetti, and H. Krawczyk. A modular approach to the design and analysis of authentication and key exchange protocols. In *Proceedings of the thirtieth annual ACM symposium on Theory of computing*, pages 419–428. ACM Press, 1998. Full version at `http://www-cse.ucsd.edu/users/mihir/papers/key-distribution.html`.

5. M. Bellare, J. Kilian, and P. Rogaway. The security of the cipher block chaining message authentication code. *Journal of Computer and System Sciences*, 61(3):362–399, December 2000. Full paper at `http://www-cse.ucsd.edu/~mihir/papers/cbc.html`.
6. D. J. Bernstein. Pippenger's exponentiation algorithm. Available from `http://cr.yp.to/papers.html`, 2001.
7. D. Boneh and M. Franklin. Identity-based encryption from the Weil pairing. In J. Kilian, editor, *Advances in Cryptology - CRYPTO 2001, 21st Annual International Cryptology Conference*, volume 2139 of *Lecture Notes in Computer Science*, pages 213–229. Springer Verlag, 2001.
8. C. Boyd, W. Mao, and K. G. Paterson. Key agreement using statically keyed authenticators. In *Applied Cryptography and Network Security: Second International Conference, ACNS 2004*, volume 3089 of *Lecture Notes in Computer Science*, pages 248–262. Springer, 2004.
9. R. Canetti, O. Goldreich, and S. Halevi. The random oracle methodology, revisited. In *Proceedings of the 30th Annual ACM Symposium on Theory of Computing—STOC 98*, pages 209–218, New York, 1998. ACM Press.
10. R. Canetti and H. Krawczyk. Analysis of key-exchange protocols and their use for building secure channels. In *Advances in Cryptology – Eurocrypt 2001*, volume 2045 of *Lecture Notes in Computer Science*, pages 453–474. Springer-Verlag, 2001. `http://eprint.iacr.org/2001/040.ps.gz`.
11. L. Chen, Z. Cheng, and N. P. Smart. Identity-based key agreement protocols from pairings. Cryptology ePrint Archive, Report 2006/199, 2006. `http://eprint.iacr.org/2006/199`.
12. O. Chevassut, P.-A. Fouque, P. Gaudry, and D. Pointcheval. Key derivation and randomness extraction. Cryptology ePrint Archive, Report 2005/061, 2005. `http://eprint.iacr.org/2005/061`.
13. A. W. Dent. Adapting the weaknesses of the random oracle model to the generic group model. In *Advances in Cryptology — ASIACRYPT 2002 Proceedings*, Lecture Notes in Computer Science, pages 100–109. Springer, 2002.
14. A. W. Dent. A note on game-hopping proofs. Cryptology ePrint Archive, Report 2006/260, 2006. `http://eprint.iacr.org/2006/260`.
15. Y. Dodis. *Exposure-Resilient Cryptography*. PhD thesis, Department of Electrical Engineering and Computer Science, Massachusetts Institute of Technology, Aug. 2000. `http://theory.lcs.mit.edu/~yevgen/academic.html`.
16. Y. Dodis, R. Gennaro, J. Håstad, H. Krawczyk, and T. Rabin. Randomness extraction and key derivation using the CBC, cascade and HMAC modes. In *Advances in Cryptology — CRYPTO 2004 Proceedings*, volume 3152 of *Lecture Notes in Computer Science*, pages 494–510. Springer, 2004.
17. R. Gennaro, H. Krawczyk, and T. Rabin. Secure hashed Diffie-Hellman over non-DDH groups. In *Advances in Cryptology — EUROCRYPT 2004 Proceedings*, volume 3027 of *Lecture Notes in Computer Science*, pages 361–381. Springer, 2004. Full version in: Cryptology ePrint Archive (`http://eprint.iacr.org/2004/099`), Report 2004/099.
18. C. Gentry. Practical identity-based encryption without random oracles. In *Advances in Cryptology — EUROCRYPT 2006 Proceedings*, volume 4004 of *Lecture Notes in Computer Science*, pages 445–464. Springer, 2006.
19. I. R. Jeong, J. Katz, and D. H. Lee. One-round protocols for two-party authenticated key exchange. In *Applied Cryptography and Network Security, Second International Conference, ACNS 2004*, volume 3089 of *Lecture Notes in Computer Science*, pages 220–232. Springer, 2004.
20. E. Kiltz. Direct chosen-ciphertext secure identity-based encryption in the standard model with short ciphertexts. Cryptology ePrint Archive, Report 2006/122, 2006. `http://eprint.iacr.org/2006/122`.
21. E. Kiltz. Chosen-ciphertext secure key-encapsulation based on gap hashed diffie-hellman. In T. Okamoto and X. Wang, editors, *Public Key Cryptography*, volume 4450 of *Lecture Notes in Computer Science*, pages 282–297. Springer, 2007.
22. E. Kiltz and D. Galindo. Direct chosen-ciphertext secure identity-based key encapsulation without random oracles. Cryptology ePrint Archive, Report 2006/034, 2006. `http://eprint.iacr.org/2006/034`; full version of [23] and [24].
23. E. Kiltz and D. Galindo. Direct chosen-ciphertext secure identity-based key encapsulation without random oracles. In *11th Australasian Conference on Information Security and Privacy—ACISP 2006*, volume 4058 of *Lecture Notes in Computer Science*, pages 336–347. Springer, 2006. full version at `http://eprint.iacr.org/2006/034`.
24. E. Kiltz and D. Galindo. Threshold chosen-ciphertext secure identity-based key encapsulation without random oracles. In *Proceedings of the Fifth Conference on Security and Cryptography for Networks—SCN 2006*, volume 4116 of *Lecture Notes in Computer Science*, pages 173–185. Springer, 2006. full version at `http://eprint.iacr.org/2006/034`.
25. H. Krawczyk. SKEME: A Versatile Secure Key Exchange Mechanism for Internet. *Proceedings of SNDSS*, 96:114, 1996.
26. H. Krawczyk. HMQV: A high-performance secure diffie-hellman protocol. In *Advances in Cryptology — CRYPTO 2005*, volume 3621 of *Lecture Notes in Computer Science*, pages 546–566. Springer, 2005.

27. B. A. LaMacchia, K. Lauter, and A. Mityagin. Stronger security of authenticated key exchange. In W. Susilo, J. K. Liu, and Y. Mu, editors, *ProvSec*, volume 4784 of *Lecture Notes in Computer Science*, pages 1–16. Springer, 2007.
28. Y. Mansour, N. Nisan, and P. Tiwari. The computational complexity of universal hashing. In *Proceedings of the Twenty Second Annual ACM Symposium on Theory of Computing—STOC '90*, pages 235–243. ACM Press, 1990.
29. N. Nisan and D. Zuckerman. Randomness is linear in space. *J. Comp. and Sys. Sci.*, 52(1):43–52, 1996.
30. T. Okamoto. Authenticated key exchange and key encapsulation in the standard model. In *Advances in Cryptology – ASIACRYPT 2007*, volume 4833 of *Lecture Notes in Computer Science*, pages 474–484. Springer, 2007. Full version in Cryptology ePrint Archive, Report 2007/473, `http://eprint.iacr.org/`.
31. R. Sakai, K. Ohgishi, and M. Kasahara. Cryptosystems based on pairing. In *2000 Symposium on Cryptography and Information Security (SCIS2000)*, January 2000.
32. S. Wang, Z. Cao, and K.-K. R. Choo. New identity-based authenticated key agreement protocols from pairings (without random oracles). Cryptology ePrint Archive, Report 2006/446, 2006. `http://eprint.iacr.org/`.
33. B. Waters. Efficient identity-based encryption without random oracles. In *Advances in Cryptology - EUROCRYPT 2005*, volume 3494 of *Lecture Notes in Computer Science*, pages 114–127. Springer, 2005.

A Evaluation of IB-KEMs

In this section, we evaluate the efficiency of the IB-KEMs of Kiltz [20], Gentry [18] and Kiltz–Galindo [22, 23] when implemented using asymmetric pairings. All of these IB-KEMs have been proposed and proven secure under the assumption that the scheme uses a bilinear map on a supersingular elliptic curve which provides a mapping from a pair in the same group $\hat{e} : \mathbb{G}_1 \times \mathbb{G}_1 \rightarrow \mathbb{G}_T$. Chen et al. [11] identify four different types of pairing, with the above case corresponding to Type 1, and note that although it is suitable for 80-bit security levels, its performance degrades significantly for higher security levels. Here we focus on Type 3 pairing systems, which are the most efficient for higher security levels. Type 2, 3 and 4 pairing systems all use a bilinear map $\hat{e} : \mathbb{G}_1 \times \mathbb{G}_2 \rightarrow \mathbb{G}_T$. We show how to transform each of the suggested IB-KEMs to use a Type 2 or 3 pairing system, and Appendix B shows how the modified schemes may still be proven secure in a similar fashion to the original ones. We point out that one of the main differences between Type 3 and Types 2 and 4 is that there is no efficiently computable homomorphism ψ from $\mathbb{G}_2$ to $\mathbb{G}_1$ in Type 3 pairing systems, and that instead of $|\mathbb{G}_2| = |\mathbb{G}_1|$, Type 4 pairing systems have $|\mathbb{G}_2| = |\mathbb{G}_1|^2$, which does not appear desirable for the suggested schemes. Also, the existence (or not) of ψ affects the assumptions made in the proofs of security.

We use the following notation. For any i, $\mathbb{G}_i^* = \mathbb{G}_i - \{1\}$ where 1 is the neutral element. $|\mathbb{G}_1| = |\mathbb{G}_2| = p$ for some prime p; g_* is a generator of $\mathbb{G}_1$ and g is a generator of $\mathbb{G}_2$. In the case of Type 2, $g_* = \psi(g)$. $\hat{e} : \mathbb{G}_1 \times \mathbb{G}_2 \rightarrow \mathbb{G}_T$ is a bilinear map. $\mathsf{H} : \{0,1\}^n \rightarrow \mathbb{G}_1$ is the hash function originally defined in Waters' scheme [33] and commonly called simply Waters' hash. It is defined as $\mathsf{H}(id) = h_0 \prod_{i=1}^{n} h_i^{id_i} \in \mathbb{G}_1$ where $id = (id_1, \ldots, id_n) \in \{0,1\}^n$. $\mathsf{HGen}(\mathbb{G}_1, n)$ generates the hash key by choosing $n+1$ random group elements $h_0, h_1, \ldots, h_n \in_{\mathrm{R}} \mathbb{G}_1$ and returning them.

Figure 1 describes the modified version of each of the three suggested IB-KEMs: those by Kiltz [20], Kiltz and Galindo [22, 23] and Gentry [18]. Gentry's IBE has been modified so that the message of the IBE (chosen at random) is considered the key of the IB-KEM. We assume it requires an exponentiation in $\mathbb{G}_T$ to generate this random message. Calculation of the key in the decryption algorithm has also been modified to reduce the number of pairings; this is possible since in the IB-KEM a random key is returned if the ciphertext

Kiltz

$\mathsf{KeyGen}(1^k)$:
$\alpha, u \in_R \mathbb{G}_1^*; z = \hat{e}(\alpha, g); \mathsf{H} = \mathsf{HGen}(\mathbb{G}_1, n)$
$pk = (u, \mathsf{H}, g, z) \in \mathbb{G}_1^{n+2} \times \mathbb{G}_2 \times \mathbb{G}_T$
Return (pk, α)

$\mathsf{KeyDer}(pk, \alpha, id)$:
$s \in_R \mathbb{Z}_p$
Return $(\alpha \mathsf{H}(id)^s, g^s, u^s)$

$\mathsf{Enc}(pk, id)$:
$r \in_R \mathbb{Z}_p^*$
$c_1 = g^r \in \mathbb{G}_2; t = \mathsf{TCR}(c_1)$
$c_2 = (\mathsf{H}(id) u^t)^r \in \mathbb{G}_1$
$K = z^r \in \mathbb{G}_T$
$C = (c_1, c_2) \in \mathbb{G}_2 \times \mathbb{G}_1$
Return (C, K)

$\mathsf{Dec}(pk, d, C)$:
Parse C as (c_1, c_2)
Parse d as (d_1, d_2, d_3)
$t = \mathsf{TCR}(c_1)$
$v \in_R \mathbb{Z}_p^*$
Return $\frac{\hat{e}\left(d_1 d_3^t \left(\mathsf{H}(id) u^t\right)^v, c_1\right)}{\hat{e}(c_2, g^v d_2)}$

Kiltz-Galindo

$\mathsf{KeyGen}(1^k)$:
$\alpha, u_1, u_2 \in_R \mathbb{G}_1^*; z = \hat{e}(\alpha, g); \mathsf{H} = \mathsf{HGen}(\mathbb{G}_1, n)$
$pk = (u_1, u_2, \mathsf{H}, g, z) \in \mathbb{G}_1^{n+3} \times \mathbb{G}_2 \times \mathbb{G}_T$
Return (pk, α)

$\mathsf{KeyDer}(pk, \alpha, id)$:
$s \in_R \mathbb{Z}_p$
Return $(\alpha \mathsf{H}(id)^s, g^s)$

$\mathsf{Enc}(pk, id)$:
$r \in_R \mathbb{Z}_p^*$
$c_1 = g^r \in \mathbb{G}_2; t = \mathsf{TCR}(c_1)$
$c_2 = \mathsf{H}(id)^r \in \mathbb{G}_1$
$c_3 = (u_1^t u_2)^r \in \mathbb{G}_1$; $K = z^r \in \mathbb{G}_T$
$C = (c_1, c_2) \in \mathbb{G}_2 \times \mathbb{G}_1^2$
Return (C, K)

$\mathsf{Dec}(pk, d, C)$:
Parse C as (c_1, c_2, c_3)
Parse d as (d_1, d_2)
$t = \mathsf{TCR}(c_1)$
$v_1, v_2 \in_R \mathbb{Z}_p^*$
Ret. $\frac{\hat{e}\left(d_1 \left(u_1^t u_2\right)^{v_1} \mathsf{H}(id)^{v_2}, c_1\right)}{\hat{e}(c_2, d_2 g^{v_2}) \cdot \hat{e}(c_3, g^{v_1})}$

Gentry

$\mathsf{KeyGen}(1^k)$:
$\alpha \in_R \mathbb{Z}_p; g_1 = g_*^\alpha \in \mathbb{G}_1; u \in_R \mathbb{G}_1;$
$h_1, h_2, h_3 \in_R \mathbb{G}_2; z_0 = \hat{e}(g_*, g);$
$z_i = \hat{e}(g_*, h_i)$ for $1 \le i \le 3$;
$\mathsf{H}' \in_R$ universal hash fn. family
$pk = (g_*, g_1, u, g, h_1, h_2, h_3, z_0, z_1, z_2, z_3, \mathsf{H}') \in \mathbb{G}_1^3 \times \mathbb{G}_2^4 \times \mathbb{G}_T^4 \times \{0,1\}^k$
Return (pk, α)

$\mathsf{KeyDer}(pk, \alpha, id)$:
$s_1, s_2, s_3 \in_R \mathbb{Z}_p$
$d_i = \left(h_i g^{-s_i}\right)^{1/(\alpha - id)}$ for $1 \le i \le 3$
$d = \{(s_i, d_i) : i \in (1,2,3)\}$
Return d
If $id = \alpha$ abort. Always use same s_i for same identity.

$\mathsf{Enc}(pk, id)$:
$r \in_R \mathbb{Z}_p; K \in_R \mathbb{G}_T$
$c_1 = g_1^r g_*^{-r id} \in \mathbb{G}_1; c_2 = z_0^r \in \mathbb{G}_T$
$c_3 = K z_1^{-r} \in \mathbb{G}_T; \beta = \mathsf{H}'(c_1, c_2, c_3)$
$c_4 = z_2^r z_3^{r\beta} \in \mathbb{G}_T$
$C = (c_1, c_2, c_3, c_4) \in \mathbb{G}_1 \times \mathbb{G}_T^3$
Return (C, K)

$\mathsf{Dec}(pk, d, C)$:
Parse C as (c_1, c_2, c_3, c_4)
Parse d as $\{(s_i, d_i) : i \in (1,2,3)\}$
$\beta = \mathsf{H}'(c_1, c_2, c_3)$
$v \in_R \mathbb{Z}_p^*$
Ret. $\frac{\hat{e}\left(c_1, d_1 \left(d_2 d_3^\beta\right)^v\right) c_3 c_2^{s_1 + v(s_2 + \beta s_3)}}{c_4^v}$

Costs

	Type 3	Kiltz			Kiltz-Galindo			Gentry		
	cost	Enc	Dec	KeyDer	Enc	Dec	KeyDer	Enc	Dec	KeyDer
$\mathbb{G}_1$ exp, multi-exp.	1, 1.5	-,1	-,1	2,-	1,1	1,1	1,-	-,1	-,-	-,-
$\mathbb{G}_2$ exp, multi-exp.	3, 4.5	1,-	1,-	1,-	1,-	2,-	1,-	-,-	-,1	-,3
$\mathbb{G}_T$ exp, multi-exp.	3, 4.5	1,-	-,-	-,-	1,-	-,-	-,-	3,1	-,1	-,-
Pairing	20	-	2	-	-	3	-	-	1	-
$\mathbb{G}_1$ subgroup check	1	-	1	-	-	2	-	-	1	-
$\mathbb{G}_2$ subgroup check	3	-	1	-	-	1	-	-	-	-
$\mathbb{G}_T$ subgroup check	4	-	-	-	-	-	-	-	3	-
Total cost		7.5	48.5	5	8.5	73.5	4	15	42	13.5
Total Enc + Dec cost		56			82			57		

Fig. 1. IB-KEMs generalized to use $\hat{e} : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$

is not consistent, and so checking for consistency can be incorporated into calculation of the key, as in the Kiltz–Galindo IB-KEMs.

The costs per operation shown in Figure 1 are from Chen et al. [11] for Type 3 curves with a security parameter of 128 to compare the cost of each IB-KEM on the same sized group (also shown in Table 2). A security parameter of 128 bits requires 256 bits to represent an element of $\mathbb{G}_1$, 512 bits to represent an element of $\mathbb{G}_2$, and 3072 bits to represent an element of $\mathbb{G}_T$. The number of operations required by each scheme is similar to that in Kiltz's comparison of IBEs, with the exception of the inclusion in these figures of subgroup membership tests, a more efficient key derivation for Gentry's scheme, and a separate listing of exponentiations for each of the three groups. When two or more pairings are required to calculate a key, it is possible that the 20 units estimated per pairing is too high, since a significant portion of the pairing computation involves finding a unique representation of the output, and this would only be necessary once the different pairing outputs had been combined.

The fastest IB-KEM proposed so far is that of Kiltz, followed closely by that of Gentry. Kiltz's scheme uses a slightly non-standard assumption, namely the mBDDH (modified bilinear decisional Diffie-Hellman) assumption. The assumption may be modified to allow $\mathbb{G}_1 \neq \mathbb{G}_2$ when ψ is not efficiently computable and stated as follows. Given $(g_*, g_*^a, g_*^b, g_*^{(b^2)}, g_*^c,\ g, g^a, g^b, g^c, W)$, it is hard to distinguish whether $W = \hat{e}(g_*, g)^{abc}$ or not. The Kiltz-Galindo scheme is based on the standard BDDH assumption: given $(g_*, g_*^a, g_*^b, g_*^c, g, g^b, g^c, W)$, it is hard to distinguish whether $W = \hat{e}(g_*, g)^{abc}$ or not. (When $\mathbb{G}_1 = \mathbb{G}_2$, ψ is efficiently computable, or a gap assumption is used in the proof, the powers of g_* may be omitted in both assumptions, provided every power of g_* is given as a power of g instead.) Gentry's scheme is based on a very non-standard assumption, the truncated decision q-ABDHE (augmented bilinear Diffie-Hellman exponent) assumption, whose hardness in relation to the BDDH assumption is unknown. The assumption (modified for the case when $\mathbb{G}_1 \neq \mathbb{G}_2$) states: given $(g_*, g_*^{(\alpha^{q+2})}, g, g^\alpha, g^{(\alpha^2)}, g^{(\alpha^3)}, \ldots, g^{(\alpha^q)}, W)$ it is hard to distinguish whether $W = \hat{e}\left(g_*,\, g^{(\alpha^{q+1})}\right)$ or not. However, by using this assumption, Gentry is able to achieve a tight reduction, whereas the other two schemes do not, losing a factor of about $(n+1)q$ where q is the upper bound of the number of key derivation/decryption queries in the KEM.

One drawback of the Kiltz and Kiltz-Galindo schemes is the use of the Waters hash, H. If we assume that identities are 30 bits long, this contributes 31 elements of $\mathbb{G}_1$ to the system parameters, or $7,936$ bits using the above figures. If this figure is seen to be too large, Gentry's scheme may appear more attractive. However, it requires the use of a universal hash function, which if implemented using the well-known method of multiplying a Toeplitz matrix (one with constant diagonals) by the input to create the output [28], would require a key of $6,655$ bits ($256 + 3072 + 3072$ bits input plus 256 bits output minus 1). Although this may be an improvement on the storage requirements for H, it is not a huge one.

On the other hand, it is possible to make a tradeoff between the size of the Waters hash and the security of the IB-KEM scheme (see e.g. Kiltz [20] for details). The tradeoff chooses a parameter l and considers each identity a sequence of l-bit strings. Then, for identities n bits long, only $(l/n + 1)$ h_i need to be chosen to define the hash function H, and each h_i is raised to the i^{th} l-bit string in the identity to find the hash of an identity. A multiplicative factor of 2^l must then be added to the security reduction of the IB-KEM scheme.

B Extension of Proofs to Ordinary Curves

Given the modified assumptions provided in Appendix A for the three IB-KEM schemes, it is not hard to modify the proofs to accommodate the modified IBKEMs. Each value or calculation defined in the proof must be examined to see whether it should be in $\mathbb{G}_1$, $\mathbb{G}_2$ or $\mathbb{G}_T$. This is fairly straightforward given the descriptions of the modified schemes. Then, wherever necessary, any value g^x for any x should be replaced with g_*^x to put the value or calculation in the correct group. The only exceptions to this rule are the calculations of the value K in Game 7 of Kiltz's proof and Game 8 in Kiltz and Galindo's proof. As originally written, these calculations would require the division of elements from two different groups. However, the calculations can be modified to avoid this problem but provide the same answer. In the Kiltz proof, the calculation must be changed to be:

$$K = \left(\frac{\hat{e}\left(c_2,\, g^a\right)}{\hat{e}\left(g_*^a,\, c_1^{x(id)}\right)} \right)^{(t-t^*)^{-1}} \tag{4}$$

and in the Kiltz-Galindo proof, the calculation must be changed to be:

$$K = \left(\frac{\hat{e}\left(c_3,\, g^b\right)}{\hat{e}\left(g_*^b,\, c_1^d\right)} \right)^{(t-t^*)^{-1}} . \tag{5}$$

C Security proofs

C.1 Analysis Techniques

In each proof, we wish to show that each game cannot be distinguished from the previous one, without breaking a hard problem. More precisely, we wish to show that in each game, the probability that $\mathcal{B}$ outputs 0 when $b = 0$ is (almost) the same as in the previous game and the probability that $\mathcal{B}$ outputs 0 when $b = 1$ is (almost) the same as in the previous game. Then, in the final game, we wish to show that from the point of view of $\mathcal{B}$, the output of the test session is independent of all other values in $\mathcal{B}$'s view. Therefore, in the last game, $\mathcal{B}$ has no advantage in distinguishing whether $b = 0$ or $b = 1$. Since it will be shown that each game cannot be distinguished from a previous one, this means that in Game 0 $\mathcal{B}$ cannot tell the difference between $b = 0$ and $b = 1$ either. This is exactly what is necessary to show that the protocol is secure.

We use the following notation.

$$\sigma_i = \text{The event that } \mathcal{B} \text{ guesses the value of } b \text{ correctly in Game } i \tag{6}$$

$$\tau_i = |2\Pr[\sigma_i] - 1| = \text{Advantage of } \mathcal{B} \text{ in Game } i \tag{7}$$

We observe that if Games i and $i + 1$ are identical when event E does not occur, and if there is a probability of $\frac{1}{2}$ that $\mathcal{B}$ is correct in Game $i + 1$ when E does occur, then:

$$\Pr[\sigma_{i+1}] = \Pr[\sigma_{i+1}|E]\Pr[E] + \Pr[\sigma_{i+1}|\neg E]\Pr[\neg E] \tag{8}$$

$$= \frac{1}{2}\Pr[E] + \Pr[\sigma_i|\neg E]\Pr[\neg E] \tag{9}$$

$$\Pr[\sigma_i] - \Pr[\sigma_{i+1}] = \Pr[\sigma_i|E]\Pr[E] - \frac{1}{2}\Pr[E] \tag{10}$$

$$|\Pr[\sigma_i] - \Pr[\sigma_{i+1}]| \leq \frac{1}{2}\Pr[E] \tag{11}$$

We also use the following game hopping technique suggested by Dent [14]. Consider an event E that may occur during $\mathcal{B}$'s execution such that E is detectable by the simulator, E is independent of σ_i, Game i and Game $i+1$ are identical unless E occurs, and $\Pr[\sigma_{i+1}|E] = \frac{1}{2}$. Then we have:

$$\Pr[\sigma_{i+1}] = \Pr[\sigma_{i+1}|E]\Pr[E] + \Pr[\sigma_{i+1}|\neg E]\Pr[\neg E] \tag{12}$$
$$= \frac{1}{2}\Pr[E] + \Pr[\sigma_i|\neg E]\Pr[\neg E] \tag{13}$$
$$= \frac{1}{2}\left(1 - \Pr[\neg E]\right) + \Pr[\sigma_i]\Pr[\neg E] \tag{14}$$
$$= \frac{1}{2} + \Pr[\neg E]\left(\Pr[\sigma_i] - \frac{1}{2}\right) \tag{15}$$
$$\text{Hence, } \tau_{i+1} = 2\left|\Pr[\sigma_{i+1}] - \frac{1}{2}\right| = 2\left|\Pr[\neg E]\left(\Pr[\sigma_i] - \frac{1}{2}\right)\right| \tag{16}$$
$$= \Pr[\neg E]\tau_i \tag{17}$$

C.2 Proof of Theorem 1

We now proceed to prove Theorem 1. The proof has two parts; the first part proves the security of Protocol 1 when the partner to the test session is not corrupted. The second part proves the security of Protocol 1 when the partner to the test session is corrupted (in this case, we require the test session to have a matching session by the time $\mathcal{B}$ finishes). Remember that we are only considering partial forward secrecy, and therefore $\mathcal{B}$ does not corrupt both the owner of the test session and the corresponding partner.

Throughout the proof, we call each session to be activated at a party an oracle. We denote the oracles with which $\mathcal{B}$ interacts Π^i_X where X is the name of a party and i is the number of the oracle. We number the oracles such that Π^i_X is the i^{th} oracle created by $\mathcal{B}$ out of all oracles created by $\mathcal{B}$ (i.e. if Π^i_X and Π^j_Y are two oracles, then $i = j$ implies $X = Y$). Also, for any party, X, the identity of that party is denoted e_X. We consider the following series of games with $\mathcal{B}$.

Case 1: Partner to the test-session is not corrupted In this case, the partner to the test session is not corrupted, but the owner of the test session may be, either prior to the session (as in a KCI attack), or after the session expires (as in a forward secrecy attack). This part of the proof uses the following series of games with $\mathcal{B}$.

Game 0. This game is the same as a real interaction with the protocol. A random bit b is chosen, and when $b = 0$, the real key is returned in answer to the test session query, otherwise a random key from $\mathbb{U}_2$ is returned.

Game 1. This game is the same as the previous one, except that if two different sessions output exactly the same message and have the same intended partner, the protocol halts.

Game 2. This game is the same as the previous one, except that before the adversary begins, a random value $m \in_{\text{R}} \{1, 2, \ldots, n_{\text{orac}}\}$ is chosen. We call the m^{th} oracle to be activated the target oracle. If the target oracle is not the test oracle, the protocol halts and $\mathcal{B}$ fails and outputs a random bit. We denote the input message to the target oracle with C, the corresponding output message with C^*, the target oracle's owner with T and the target oracle's intended partner with T^*. Note that there may not be a matching test session activated at T^*.

Game 3. In this game, a random value K'^* is chosen. Whenever C^* is used as input to an oracle owned by T^*, the calculation of the key is modified so that K'^* is used in place of $\mathsf{Dec}(pk, d_{T^*}, C^*)$; the message output by this oracle is calculated as usual. Similarly, K'^* is used instead of K'_T in the calculation of the session key by T.
The rest of the Game 3 is the same as Game 2. If $b = 1$, a random key from $\mathbb{U}_2$ is returned. Otherwise, K'^* is used in the computation of the test session as described above.

Game 4. This game is the same as the previous one, except that a random value $K''^* \in_{\mathrm{R}} \mathbb{U}_1$ is chosen and the use of $\mathsf{Exct}_\kappa(K'^*)$ is replaced with K''^*.

Game 5. This game is the same as the previous one, except that whenever the value $\mathsf{Expd}_{K''^*}(s')$ for any s' would be used in generating keys, a random value from $\mathbb{U}_2$ is used instead (the same random value is used for the same value of s'; a different random value is chosen for different s').

Analysis of Games 0 to 2: Let p_{sameMsg} be the probability of two or more sessions outputting the same message. We have

$$1 - p_{\mathrm{sameMsg}} > 1 - \frac{n_{\mathrm{orac}}^2}{p.}$$

Then, from (11)

$$|\Pr[\sigma_0] - \Pr[\sigma_1]| < \frac{n_{\mathrm{orac}}^2}{2p} \quad \text{when } \tfrac{1}{2} > \tfrac{n_{\mathrm{orac}}-1}{p} > 0 \tag{18}$$

This can be used to bound τ_0 as follows:

$$\tau_0 = |2\Pr[\sigma_0] - 1| \leq 2\left(|\Pr[\sigma_0] - \Pr[\sigma_1]| + \left|\Pr[\sigma_1] - \frac{1}{2}\right|\right) \tag{19}$$

$$\leq \frac{n_{\mathrm{orac}}^2}{p} + \tau_1 \tag{20}$$

In Game 2, the probability of the protocol halting due to an incorrect choice of m is $1 - \frac{1}{n_{\mathrm{orac}}}$. Whether or not an abortion would occur in this game could be detected in the previous game if it also chose m in the same way. Therefore, we may use equation (17) to find:

$$\tau_2 = \frac{1}{n_{\mathrm{orac}}}\tau_1 \quad \Rightarrow \quad n_{\mathrm{orac}}\tau_2 = \tau_1 \tag{21}$$

and (20) gives

$$\tau_0 \leq \frac{n_{\mathrm{orac}}^2}{p} + n_{\mathrm{orac}}\tau_2 \tag{22}$$

Analysis of Game 3: We now construct adversary $\mathcal{A}$ against the security of the IB-KEM, using $\mathcal{B}$. $\mathcal{A}$ is constructed such that when it receives the real key for the IB-KEM scheme, the view of $\mathcal{B}$ is the same as in Game 2, but if $\mathcal{A}$ receives a random IB-KEM key, the view of $\mathcal{B}$ is the same as in Game 3. Then, by the security of the IB-KEM scheme, we can claim these games are indistinguishable.

To begin, $\mathcal{A}$ is given the master public key pk. $\mathcal{A}$ passes this value as well as the description of $\mathsf{Exct}_\kappa(\cdot)$, its key κ and $\{\mathsf{Expd}_K(\cdot)\}_{K\in\mathbb{U}_1}$ to $\mathcal{B}$. Recall that $\mathcal{A}$ has access to the corresponding oracles $\mathcal{O}_{\mathsf{KeyDer}}(\cdot)$ and $\mathcal{O}_{\mathsf{Dec}}(\cdot,\cdot)$.

$\mathcal{A}$ runs as described in Game 2, except that when the target session is activated, $\mathcal{A}$ outputs e_{T^*} as the identity on which it wants to be tested. $\mathcal{A}$ receives a ciphertext C^* for T^* and key K'^*, which may be the decryption of C^* or may be a random IB-KEM key, each with equal probability. $\mathcal{A}$ then uses C^* as the output of the target session, modifies the calculation of keys so that K'^* is used in place of $\mathsf{Dec}(pk, \mathsf{KeyDer}(pk, \alpha, e_{T^*}), C^*)$, and uses K'^* instead of K'_T to find the answer to the test session query when $b = 0$.

All legitimate queries made by $\mathcal{B}$ can still be answered by $\mathcal{A}$ using its oracles in as follows.

- A corrupt query on some identity e_X may be answered with $\mathcal{O}_{\mathsf{KeyDer}}(e_X)$(recall that no corrupt query is made by $\mathcal{B}$ on the partner to the test session, T^*).
- $\mathcal{A}$ must maintain the session state of each oracle so that it may be returned in answer to session state reveal queries (session state reveal is not allowed on the test session or its matching session).
- Any message C_X to any party (including T^*) with identity e_X may be decrypted using $\mathcal{O}_{\mathsf{Dec}}(e_X, C_X)$ to generate keys for reveal session key queries and the test query.

When $\mathcal{B}$ halts and outputs its bit b', $\mathcal{A}$ halts and outputs $1 - b'$. The probability that $\mathcal{A}$ is correct is $\Pr[\sigma_2]$ when K'^* is the real key for the IB-KEM message, and $1 - \Pr[\sigma_3]$ when K'^* is not the key for the IB-KEM message. We can then find that:

$$\mathbf{Adv}^{\mathrm{ib-kem-cca}}_{\mathcal{E},\mathcal{A}}(k) = \left| 2\left(\frac{1}{2}\left(\Pr[\sigma_2] + 1 - \Pr[\sigma_3]\right)\right) - 1 \right| \tag{23}$$

$$= \Pr[\sigma_2] - \Pr[\sigma_3] \tag{24}$$

$$\tau_2 = |2\Pr[\sigma_2] - 1| \leq |2\Pr[\sigma_2] - 2\Pr[\sigma_3]| + |2\Pr[\sigma_3] - 1| \tag{25}$$

$$\tau_2 \leq 2\mathbf{Adv}^{\mathrm{ib-kem-cca}}_{\mathcal{E},\mathcal{A}}(k) + \tau_3 \tag{26}$$

Analysis of Game 4 We now consider an adversary, $\mathcal{D}$, against the security of the randomness extraction function. This adversary runs a copy of $\mathcal{B}$ and interacts with $\mathcal{B}$ in such a manner that it is the same as when $\mathcal{B}$ interacts with either Game 3 or 4. $\mathcal{D}$ receives a key κ for the randomness extraction function and a value R_1 such that either $R_1 = \mathsf{Exct}_\kappa(X)$ for some $X \in_{\mathrm{R}} \mathbb{K}$ or $R_1 \in_{\mathrm{R}} \mathbb{U}_1$. $\mathcal{D}$ sets κ to be the public parameter used to key the randomness extraction function, and chooses the other public parameters according to the protocol. $\mathcal{D}$ runs as described for Game 4, except that $\mathcal{D}$ uses R_1 in place of K''^*. When $\mathcal{B}$ outputs it guess of the bit b, $\mathcal{D}$ outputs that $R_1 = \mathsf{Exct}_\kappa(X)$ for some X if $\mathcal{B}$ is correct, and $\mathcal{D}$ outputs that $R_1 \in_{\mathrm{R}} \mathbb{U}_1$ otherwise. The probability that $\mathcal{D}$ is correct is $\frac{1}{2}\left(\Pr[\sigma_3] + 1 - \Pr[\sigma_4]\right)$. By the security of the randomness extraction function, we have:

$$\epsilon \geq |2\Pr[\mathcal{D} \text{ correct}] - 1| = |\Pr[\sigma_3] - \Pr[\sigma_4]| \tag{27}$$

$$\tau_3 = |2\Pr[\sigma_3] - 1| \tag{28}$$

$$\leq |2\Pr[\sigma_3] - 2\Pr[\sigma_4]| + |2\Pr[\sigma_4] - 1| \tag{29}$$

$$\leq 2\epsilon + \tau_4 \tag{30}$$

Analysis of Game 5 Now, we consider another adversary, $\mathcal{D}'$, this time against the randomness expansion (or pseudorandom) function family $\{\mathsf{Expd}_K(\cdot)\}_{K \in \mathbb{U}_1}$. We define $\mathcal{D}'$ to run a copy of $\mathcal{B}$, and to interact with $\mathcal{B}$ in such a manner that it is the same as when $\mathcal{B}$ interacts with with either Game 4 or 5. $\mathcal{D}'$ receives the definition of the function family $\{\mathsf{Expd}_K(\cdot)\}_{K \in \mathbb{U}_1}$, and an oracle $\mathcal{O}(\cdot)$ which is either $\mathsf{Expd}_K(\cdot)$ for some value of K unknown

to $\mathcal{D}'$ or a truly random function. $\mathcal{D}'$ runs a copy of the protocol for $\mathcal{B}$ in the same way as described for Game 4, except that whenever the value $\mathsf{Expd}_{K''^*}(s')$ for any s' would be used in generating keys, $\mathcal{D}'$ uses the value $\mathcal{O}(s')$ instead. When $\mathcal{B}$ outputs it guess of the bit b, $\mathcal{D}'$ outputs that its oracle is a member of the given function family if $\mathcal{B}$ is correct, and $\mathcal{D}'$ outputs that its oracle is a truly random function otherwise. The probability that $\mathcal{D}'$ is correct is $\frac{1}{2}(\Pr[\sigma_4] + 1 - \Pr[\sigma_5])$. By the security of the randomness expansion function we have:

$$\mathbf{Adv}^{\mathrm{p-rand}}_{\mathcal{F},\mathcal{C}}(k) \geq \left|2\Pr[\mathcal{D}' \text{ correct}] - 1\right| = |\Pr[\sigma_4] - \Pr[\sigma_5]| \tag{31}$$

$$\tau_4 = |2\Pr[\sigma_4] - 1| \tag{32}$$

$$\leq |2\Pr[\sigma_4] - 2\Pr[\sigma_5]| + |2\Pr[\sigma_5] - 1| \tag{33}$$

$$\leq 2\mathbf{Adv}^{\mathrm{p-rand}}_{\mathcal{F},\mathcal{C}}(k) + \tau_5 \tag{34}$$

In Game 5, let us denote the key returned in the test session query with $R_1 \oplus \mathsf{Expd}_{K''_{T^*}}(s')$ when $b = 0$, and R_2 when $b = 1$, where R_1 and R_2 are chosen uniformly at random from $\mathbb{U}_2$. Now, R_2 is chosen independently of all other values in the protocol, so $\mathcal{B}$ can gain no information about R_2 directly; $\mathcal{B}$ can only gain information about R_2 by determining whether $b = 0$ or $b = 1$. Furthermore, when $b = 0$, unless $\mathcal{B}$ can gain some information about R_1, the response to the test session query also looks random and is therefore indistinguishable from the case when $b = 1$.

To gain information about R_1 from a source other than the test session query response, $\mathcal{B}$ must obtain the key of a session that has also used R_1 in the generation of its key. Now, if R_1 is used in the generation of a session's key, then that session must have had the same session identifier, and hence exchanged the same messages as the test session. Therefore, the session is either owned by T^* with intended partner T and received C^* as part of its input or is owned by T with intended partner T^* and had C^* as part of its output. However, such a session owned by T^* will match the test session and so not be subject to reveal key queries.Hence, $\mathcal{B}$ can gain no information about R_1, and so $\mathcal{B}$ can gain no information about b in Game 5, and therefore

$$\tau_5 = 0$$

Combining the results in equations (22), (26), (30) and (34) we conclude:

$$\tau_0 \leq \frac{n^2_{\mathrm{orac}}}{p} + 2n_{\mathrm{orac}}\left(\mathbf{Adv}^{\mathrm{ib-kem-cca}}_{\mathcal{E},\mathcal{A}}(k) + \epsilon + \mathbf{Adv}^{\mathrm{p-rand}}_{\mathcal{F},\mathcal{C}}(k)\right) \tag{35}$$

Case 2: Partner to the test session is corrupted Recall that we only consider partial weak forward secrecy for Protocol 1. Consequently we require that

1. the owner of the test session be not corrupted by the adversary,
2. the adversary be passive in the protocol corresponding to the test session; that is there exists a matching session to the test session at the intended partner by the time that the test session query is issued by the adversary; and
3. the partner to the test session be corrupted only after the matching session expires.

For this part of the proof we set up the following series of 6 games with $\mathcal{B}$. Game 0 and 1 are the same as in Case 1. Game 2 and 3 are analogous to Game 2 and 3 in Case 1 except that now our target oracle is the partner to the test session, and it is its input to the session key that is substituted by a random value. Game 4 and 5, which are used to prove the security of the session key derivation mechanism via randomness extraction and expansion also remain essentially the same.

Game 0. This game is the same as a real interaction with the protocol. A random bit b is chosen, and when $b = 0$, the real key is returned in answer to the test session query, otherwise a random key from $\mathbb{U}_2$ is returned.

Game 1. This game is the same as the previous one, except that if two different sessions output exactly the same message and have the same intended partner, the protocol halts.

Game 2. This game is the same as the previous one, except that before the adversary begins, a random value $m \in_{\mathrm{R}} \{1, 2, \ldots, n_{\mathrm{orac}}\}$ is chosen. We call the m^{th} oracle to be activated the target oracle. If the target oracle is not the *partner* to the test oracle, the protocol halts and $\mathcal{B}$ fails and outputs a random bit. We denote the input message to the target oracle with C^*, the corresponding output message with C, the target oracle's owner with T^* and the target oracle's intended partner with T.

Game 3. In this game, a random value $K' \in_{\mathrm{R}} \mathbb{K}$ is chosen. Further a bit $c \in_R \{0, 1\}$ is chosen as a guess as to whether $\mathcal{B}$ corrupts T or T^*. Whenever C is used as input to an oracle owned by T, the calculation of the key is modified so that K' is used in place of $\mathsf{Dec}(pk, d_T, C)$; the message output by this oracle is calculated as usual. Similarly, K' is used instead of K'_T in the calculation of the session key by T^*.
The rest of the Game 3 is the same as Game 2. if $b = 1$, a random key from $\mathbb{U}_2$ is returned. Otherwise, K' is used in the computation of the test session as described above.

Game 4. This game is the same as the previous one, except that a random value $K'' \in_{\mathrm{R}} \mathbb{U}_1$ is chosen and the use of $\mathsf{Exct}_\kappa(K')$ is replaced with K''.

Game 5. This game is the same as the previous one, except that whenever the value $\mathsf{Expd}_{K''}(s')$ for any s' would be used in generating keys, a random value from $\mathbb{U}_2$ is used instead (the same random value is used for the same value of s'; a different random value is chosen for different s').

In the analysis that follows, we denote with σ'_i the event that the adversary is successful in Game i and with τ'_i the corresponding advantage.

Analysis of Games 0 to 2: This analysis is the same as the that of Games 0 to 2 in Case 1. Thus,

$$\tau'_0 \leq \frac{n^2_{\mathrm{orac}}}{p} + n_{\mathrm{orac}}\tau'_2 \tag{36}$$

Analysis of Game 3: This analysis is very similar to that of Game 3 in Case 1. We construct adversary $\mathcal{A}$ against the security of the IB-KEM, using $\mathcal{B}$. $\mathcal{A}$ is given the master public key pk and passes this value as well as the description of $\mathsf{Exct}_\kappa(\cdot)$, its key κ and $\{\mathsf{Expd}_K(\cdot)\}_{K\in\mathbb{U}_1}$ to $\mathcal{B}$. $\mathcal{A}$ runs as described in Game 2, except that when the target oracle is activated, $\mathcal{A}$ outputs e_T as the identity on which it wants to be tested. $\mathcal{A}$ receives a ciphertext C for T and key K', which may be the decryption of C or may be a random IB-KEM key, each with equal probability. $\mathcal{A}$ then uses C as the output of the target oracle (and hence input to the test session). $\mathcal{A}$ modifies the calculation of keys so that K' is used in place of $\mathsf{Dec}(pk, \mathsf{KeyDer}(pk, \alpha, e_T), C)$, and uses K' instead of K'_T to find the answer to the test session query when $b = 0$.

All legitimate queries made by $\mathcal{B}$ can still be answered by $\mathcal{A}$ using its oracles as follows.

- A corrupt query on some identity e_X may be answered with $\mathcal{O}_{\mathsf{KeyDer}}(e_X)$(recall that no corrupt query is made by $\mathcal{B}$ on the owner of the test session, T).

– $\mathcal{A}$ must maintain the session state of each oracle so that it may be returned in answer to session state reveal queries (session state reveal is not allowed on the test session or its matching session).
– Any message C_X to any party (including T) with identity e_X may be decrypted using $\mathcal{O}_{\mathsf{Dec}}(e_X, C_X)$ to generate keys for reveal session key queries and the test query.

When $\mathcal{B}$ halts and outputs its bit b', $\mathcal{A}$ halts and outputs $1 - b'$. As in Case 1, the probability that $\mathcal{A}$ is correct is $\Pr[\sigma_2']$ when K' is the real key for the IB-KEM message, and $1 - \Pr[\sigma_3']$ when K' is not the key for the IB-KEM message. Hence,

$$\tau_2' \leq 2\mathbf{Adv}^{\mathrm{ib-kem-cca}}_{\mathcal{E},\mathcal{A}}(k) + \tau_3' \tag{37}$$

Analysis of Game 4 This is the same as in Case 1. Thus,

$$\tau_3' \leq 2\epsilon + \tau_4' \tag{38}$$

Analysis of Game 5 This is the same as in Case 1. Thus,

$$\tau_4' \leq 2\mathbf{Adv}^{\mathrm{p-rand}}_{\mathcal{F},\mathcal{C}}(k) + \tau_5' \tag{39}$$

Combining Results Again using the same reasoning as in Case 1, we conclude that $\tau_5' = 0$, and therefore combining equations (36), (37), (38) and (39) we have:

$$\tau_0' \leq \frac{n^2_{\mathrm{orac}}}{p} + 2n_{\mathrm{orac}}\left(\mathbf{Adv}^{\mathrm{ib-kem-cca}}_{\mathcal{E},\mathcal{A}}(k) + \epsilon + \mathbf{Adv}^{\mathrm{p-rand}}_{\mathcal{F},\mathcal{C}}(k)\right) \tag{40}$$

This is the same advantage as in Case 1 and hence Therorem 1 follows.

C.3 Proof of Theorem 2:

The security difference between Protocol 1 and 2 is that the latter provides full WFS, i.e. in addition to the adversarial capabilities considered in the proof of Theorem 1, we now allow the adversary to corrupt both parties to the test session. It is natural then to consider the proof of Theorem 2 in two parts: the first part where the adversary does not corrupt both parties to the test session, and the second part where it does. Then, the first part is essentially identical to the proof of Theorem 1. The only difference is that in the analysis of Game 3 (in both Case 1 and 2) $\mathcal{A}$ needs to simulate the extra Diffie-Hellman values, which $\mathcal{A}$ can easily do for all sessions, including the test session. (Note that $\mathcal{A}$ will always choose at least one of Y_A or Y_B, hence it can always compute K''_{AB}).

We deal now with the second part of the proof, where the adversary corrupts the two partners to the test session. Note however that the adversary is restricted to being passive during the protocol run corresponding to the test session – a consequence of only being able to achieve weak forward secrecy in one round. As we will see below this allows us to inject a challenge Decisional Diffie-Hellman triplet into the test session.

The second part of the proof allows any party to be corrupted. It considers the following four games with $\mathcal{B}$.

Game 0. This game is the same as a real interaction with the protocol. A random bit b is chosen, and when $b = 0$, the real key is returned in answer to the test session query, otherwise a random key from $\mathbb{U}_2$ is returned.

Game 1. This game is the same as the previous one, except that before the adversary begins, random values $j, j^* \in_{\mathrm{R}} \{1, 2, \ldots, n_{\mathrm{orac}}\}$ are chosen. Let T and T^* be the owners of the j^{th} and $j^{*\mathrm{th}}$ sessions respectively. if the j^{th} session at T is not the test session or if the output of the $j^{*\mathrm{th}}$ session at T^* is not used as input to the test session, then the protocol halts and $\mathcal{B}$ fails and outputs a random bit. Furthermore, the session key of the test session is calculated as usual except that $\mathsf{Exct}_\kappa(h)$ for $h \in_R \langle f \rangle$ replaces K''_{TT^*}. The test session's matching session has its key set to the same value (i.e. a random test session key is always returned, no matter what the value of b).

Game 2. This game is the same as the previous one except that $\mathsf{Exct}_\kappa(h)$ is replaced with $K'' \in_R \mathbb{U}_1$

Game 3. This game is the same as the previous one, except that whenever the value $\mathsf{Expd}_{K''}(s')$ for any s' would be used in generating keys, a random value from $\mathbb{U}_2$ is used instead (the same random value is used for the same value of s'; a different random value is chosen for different s').

Analysis Game 1 We now construct adversary $\mathcal{A}$ against the DDH problem, using $\mathcal{B}$. $\mathcal{A}$ is constructed such that provided $\mathcal{A}$ does not have to abort the protocol as specified in Game 1 then if $\mathcal{A}$'s input is from $\mathcal{DH}_F$, the view of $\mathcal{B}$ is the same as in Game 0, but if $\mathcal{A}$'s input is from $\mathcal{R}_F$, the view of $\mathcal{B}$ is the same as in Game 1. Then, by Assumption 1, we can claim these games are indistinguishable.

To begin, $\mathcal{A}$ generates all the protocol parameters and passes the public parameters to $\mathcal{B}$.

Let (f^a, f^b, h) be $\mathcal{A}$'s challenge DDH inputs (with h either f^c or f^{ab}). When the j^{th} and the $j^{*\mathrm{th}}$ sessions are activated, $\mathcal{A}$ uses its inputs f^a and f^b instead of the values Y_T and Y_{T^*} when it generates the outputs of these sessions. Apart from this change, all session inputs and outputs are generated according to the protocol specification.

When the test session query is made, $\mathcal{A}$ uses $\mathsf{Exct}_\kappa(h)$ in place of K''_{TT^*} when calculating the real test session key.

$\mathcal{A}$ is able to answer all other queries correctly since it knows all of the system parameters and all of the session states. $\mathcal{A}$ outputs 1 if $\mathcal{B}$ is correct, 0 otherwise.

The probability that $\mathcal{A}$ will not have to abort the protocol as described in Game 1 is $\frac{1}{n^2_{\mathrm{orac}}}$. Hence, by (17) and using a similar logic to that shown in (31) to (34) we have that:

$$\tau_0 \leq 2n^2_{\mathrm{orac}} \mathbf{Adv}^{\mathrm{ddh}}_{F,\mathcal{D}}(k) + \tau_1 \tag{41}$$

Analysis Game 2 The analysis of Game 2 is the same as that of Game 4 of Case 1. Thus,

$$\tau_1 \leq 2\epsilon + \tau_2 \tag{42}$$

Analysis Game 3 The analysis of Game 3 is the same as that of Game 5 of Case 1 and 2. Thus,

$$\tau_2 \leq 2\mathbf{Adv}^{\mathrm{p-rand}}_{\mathcal{F},\mathcal{C}}(k) + \tau_3 \tag{43}$$

Since the test session key is masked with a random value, and that value is independent of all other sessions, $\mathcal{B}$ has no advantage in Game 3 ($\tau_3 = 0$). The value is independent because only the test session has the test session's session id.

Combining Results: Let E be the event that the test session has a matching session by the time $\mathcal{B}$ finishes, and let σ be the event that $\mathcal{B}$ guesses b correctly in Protocol 2. Then we have:

$$\begin{aligned}\mathbf{Adv}^{\mathrm{sk}}_{\mathcal{B}}(k) &= |2\Pr[\sigma] - 1| \\ \Pr[\sigma] &= \Pr[\sigma|E]\Pr[E] + \Pr[\sigma|\neg E]\Pr[\neg E] \\ &= \Pr[\sigma|\neg E] + \Pr[E]\left(\Pr[\sigma|E] - \Pr[\sigma|\neg E]\right).\end{aligned}$$

Therefore,

$$\min\left(\Pr[\sigma|\neg E], \Pr[\sigma|E]\right) \leq \Pr[\sigma] \leq \max\left(\Pr[\sigma|\neg E], \Pr[\sigma|E]\right)$$

and

$$\mathbf{Adv}^{\mathrm{sk}}_{\mathcal{B}}(k) \leq \max\left(\mathbf{Adv}^{\mathrm{sk}}_{\mathcal{B}}(k)\,|E,\ \mathbf{Adv}^{\mathrm{sk}}_{\mathcal{B}}(k)\,|\neg E\right)$$

and so we can combine (35), and (40)-(43) to find:

$$\begin{aligned}\mathbf{Adv}^{\mathrm{sk}}_{\mathcal{B}}(k) \leq \max\Big(&2n^2_{\mathrm{orac}}\mathbf{Adv}^{\mathrm{ddh}}_{F,\mathcal{D}}(k) + 2\epsilon + 2\mathbf{Adv}^{\mathrm{p-rand}}_{\mathcal{F},\mathcal{C}}(k)\,, \\ &\frac{n^2_{\mathrm{orac}}}{p} + 2n_{\mathrm{orac}}\left(\mathbf{Adv}^{\mathrm{ib-kem-cca}}_{\mathcal{E},\mathcal{A}}(k) + \epsilon + \mathbf{Adv}^{\mathrm{p-rand}}_{\mathcal{F},\mathcal{C}}(k)\right).\end{aligned}$$