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`

January 4, 2008

Abstract. We consider one-round identity-based 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, 4]. 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 identity-based 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]. 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 resilience, 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 [28] 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. Our focus is on protocols that can be built upon an ID-based encryption infrastructure. Hence we exclude existing certificate-based key exchange protocols, even when they can also be seen as ID-based.

1.1 Contributions

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 [21] 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.

Our first protocol is the most efficient of the two, and provides key-compromise impersonation (KCI) resistance but not 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 use a randomness extractor to derive a session key from the combined secrets. Protocol 2 is based on the first protocol, but adds an independent Diffie-Hellman exchange to achieve forward secrecy. It also achieves KCI resistance.

Our approach can also be quite naturally considered in the normal public key setting, although for brevity we only deal with the ID-based setting in this paper. We note, nevertheless, that in the normal public key setting we obtain efficient, one round, concrete protocol designs in the standard model, which compare favorably with the protocol of Jeong, Katz and Lee [19], which is to our knowledge the only other one-round protocol 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 [24].

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 Protocols 1 and 2 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.

1.2 Related Work

Following the development of practical schemes for identity-based encryption [27, 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 [29], Kiltz [20], Gentry [18] and Kiltz–Galindo [21, 22].

Up until now, all proofs for identity-based key exchange protocols have continued to rely on the ROM, with the exception [28] noted. However, although Wang et al. [28] 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. [28] 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 [6], 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, call TS3, which is one-round and proven secure in the standard model. TS3 is a Diffie-Hellman (DH) key exchanged authenticated using a MAC keyed under the (static) DH of the long term keys of the two users. TS3 provides forward secrecy, but fails to achieve KCI resilience – 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 secure.

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*

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

(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 [16, p.42][26]). *A family of efficiently computable hash functions $\mathcal{H} = \{h_\kappa : \{0,1\}^n \to \{0,1\}^k | \kappa \in \{0,1\}^d\}$ 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 following two distributions are within statistical distance ϵ from each other:*

$$\langle \kappa, h_\kappa(X) \rangle \cong_\epsilon \langle \kappa, R \rangle$$

To implement the randomness extraction function, one could apply the work of Dodis et al. [15] to use CBC-MAC, keyed cascade chaining, or HMAC as an almost universal hash function, and conclude that the function is a good randomness extraction function based on their extension of the leftover hash lemma.

Definition 3 (Pseudorandom Function Family (PRF) [5]). *A function family $F = \{f_\kappa\}_{\kappa \in K}$ is (S, q, ϵ) pseudorandom-secure if a circuit $\mathcal{C}$ of size S which is given oracle access to either F_κ for $\kappa \in_R K$ or a truly random function with the same domain and range as the functions in F, and makes at most q queries to this oracle, has advantage at most ϵ in distinguishing whether it has access to a random member of F or a truly random function. That is,*

$$\mathbf{Adv}_{\mathcal{C}}^{\text{p-rand}}(F) = |\Pr[\mathcal{C}^{F_\kappa(\cdot)}(F) = 1] - \Pr[\mathcal{C}^{\text{Rand}(\cdot)}(F) = 1]| \leq \epsilon$$

Functions that are proven to be pseudorandom include CBC-MAC [2] (provided the underlying block cipher is a secure pseudorandom permutation family and the input length is constant) and HMAC [3] (provided the compression function is a PRF).

Definition 4 ([Target] collision resistant hash function [21, p.5]). *Let $\mathcal{F} = \{\mathsf{H}_s\}_{s \in S}$ be a family of hash functions for security parameter k and with seed $s \in S = S(k)$. $\mathcal{F}$ is said to be* target collision resistant *(*collision resistant *respectively) if, for a hash function $\mathsf{H} = \mathsf{H}_s$ (where the seed is chosen at random from S), given a randomly chosen element x, it is infeasible for an efficient adversary to find a distinct value $y \neq x$ such that $\mathsf{H}(x) = \mathsf{H}(y)$ (respectively, it is infeasible for an efficient adversary to find two distinct values $x \neq y$ such that $\mathsf{H}(x) = \mathsf{H}(y)$). We define $\mathbf{Adv}_{\mathsf{H},\mathcal{A}}^{\text{tcr}}(k)$ ($\mathbf{Adv}_{\mathsf{H},\mathcal{A}}^{\text{cr}}(k)$ respectively) to be the probability that $\mathcal{A}$ finds a collision in H. The hash function family H is said to be* target collision resistant *(*collision resistant *respectively) if the advantage function is a negligible function in k for all polynomial-time adversaries $\mathcal{A}$.*

Assumption 1 (Decisional Diffie-Hellman (DDH) [17, p.5]) *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 (S, ϵ) Decisional Diffie-Hellman (DDH) Assumption holds over $F = \langle f \rangle$ (alternatively, that F is a (S, ϵ) DDH group) if the two distributions $\mathcal{R}_F$ and $\mathcal{DH}_F$ are (S, ϵ)-indistinguishable.

Definition 5 (ID-based KEM). *An IB-KEM $\mathcal{E} = (\mathsf{KeyGen}, \mathsf{KeyDer}, \mathsf{Enc}, \mathsf{Dec})$ consists of four polynomial-time algorithms:*

- $(pk, \alpha) \in_{\mathrm{R}} \mathsf{KeyGen}(1^k)$*, given the security parameter $k \in \mathbb{N}$, returns a master public key, pk, and master secret key α;*
- $d_{id} \in_{\mathrm{R}} \mathsf{KeyDer}(pk, \alpha, id)$ *generates a private key corresponding to the identity id.*
- $(C, K) \in_{\mathrm{R}} \mathsf{Enc}(pk, id)$ *outputs a random key K and an encapsulation (ciphertext) C of the key under the identity id;*
- $K = \mathsf{Dec}(d_{id}, C)$ *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 [21].

Definition 6 (IB-KEM-CCA Security). *The security of an IB-KEM scheme $\mathcal{E} = (\mathsf{KeyGen}, \mathsf{KeyDer}, \mathsf{Enc}, \mathsf{Dec})$ is defined using the following experiment.*

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

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

$\mathcal{O}_{\mathsf{KeyDer}}(id) = \mathsf{KeyDer}(pk, \alpha, id)$ *(where $id \neq id^*$)*
$\mathcal{O}_{\mathsf{Dec}}(id, C) = \mathsf{Dec}(pk, \mathsf{KeyDer}(pk, \alpha, id), C)$ *(where $id \neq id^*$ or $C \neq C^*$)*

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

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

$\mathcal{E}$ 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 [6, 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 as 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: 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.

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 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. $\mathsf{session\text{-}state}(P_i, P_j, s)$. This query returns all the internal state information of party P_i associated to a particular session s with P_j.
4. $\mathsf{session\text{-}expiration}(P_i, P_j, s)$. This query is used for defining *forward secrecy* and erases from memory the session key on completed session.
5. $\mathsf{test\text{-}session}(P_i, P_j, s)$. 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 $\mathsf{session\text{-}state}$ or $\mathsf{session\text{-}key}$ 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 7. *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

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

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 [24] 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*, in which the adversary is forbidden from taking an active part in the test session.

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 believe 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 resilience 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).

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 random extractor $\mathsf{Exct}_\kappa(\cdot) : \mathbb{K} \to \mathbb{U}_1$ with key κ and expander $\{\mathsf{Expd}_K(\cdot)\}_{K \in \mathbb{U}_1} : \{0,1\}^\sigma \to \mathbb{U}_2$. Protocol 2, adds an independent Diffie-Hellman exchange in a group generated by f to achieve forward secrecy. It also achieves KCI resistance.

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.

Use of the collision resistant hash function to find s' is unnecessary if the randomness expander is able to accept inputs at least as long as s; in that case, we could simply set $s' = s$. Also 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, if session-state reveal queries are to be allowed, then a decapsulated IB-KEM key must be securely erased in the same activation in which it is decapsulated, as the decapsulation cannot be simulated in the proof.

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

- $\mathsf{Exct}_\kappa(\cdot)$ is a function chosen uniformly at random from a strong ϵ-randomness extractor, (as described in Definition 2),

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

Protocol 2: Generic 2×KEM + Diffie-Hellman

- the function family $\{\mathsf{Expd}_K(\cdot)\}_{K \in \{0,1\}^{k'}}$ is a $(S, 2n_{\mathrm{orac}}, \epsilon')$-pseudorandom secure function family, (as described in Definition 3),
- n_{orac} is the total number of oracles (i.e. sessions) created by $\mathcal{B}$ against the protocol,
- $\frac{1}{p}$ 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),
- for Protocol 2, the (S'', ϵ'') DDH assumption (see Assumption 1) holds over $\langle f \rangle$, where S'' is the circuit size of $\mathcal{B}$ (and the circuit size of $\mathcal{B}$ includes the circuit size of parties activated by $\mathcal{B}$).

We also need to assume that any adversary $\mathcal{B}$ may not corrupt the test session's intended partner (but may corrupt any other party and perform session-state reveal queries) in order for the proofs to hold.

Theorem 1. *Let $\mathcal{B}$ be any adversary against Protocol 1. Then the advantage of $\mathcal{B}$ against the SK-security of Protocol 1 is:*

$$\begin{aligned}\mathbf{Adv}_{\mathcal{B}}^{\mathrm{sk}}(k) \leq \mathbf{Adv}_{\mathit{CR},\mathcal{A}}^{\mathrm{cr}}(k) + \frac{n_{\mathrm{orac}}^{2}}{p} \\ +2n_{\mathrm{orac}}\left(2\mathbf{Adv}_{\mathcal{E},\mathcal{A}}^{\mathrm{ib-kem-cca}}(k) + \epsilon + \epsilon'\right)\end{aligned}$$

where p, ϵ, ϵ', k and k are all functions of the security parameter k.

Theorem 2. *Let $\mathcal{B}$ be any adversary against Protocol 2. Then the advantage of $\mathcal{B}$ against the SK-security of Protocol 2 is:*

$$\begin{aligned}\mathbf{Adv}_{\mathcal{B}}^{\mathrm{sk}}(k) \leq \max\Bigg(\mathbf{Adv}_{\mathit{CR},\mathcal{A}}^{\mathrm{cr}}(k) + \frac{n_{\mathrm{orac}}^{2}}{p} \\ +2n_{\mathrm{orac}}\left(2\mathbf{Adv}_{\mathcal{E},\mathcal{A}}^{\mathrm{ib-kem-cca}}(k) + \epsilon + \epsilon'\right), 2n_{\mathrm{orac}}^{2}\epsilon''\Bigg).\end{aligned}$$

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 seems to be much less simple than may initially be expected.

5 Protocol Comparison

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].

Table 1 summarises the properties of the protocols under consideration. 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 resilience in the standard model.

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

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

Table 1. Security and efficiency comparison

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.

	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		

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

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 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 [21] and Gentry [18] to use ordinary elliptic curves.We found that both our Protocols takes 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.

Lack of space precluded inclusion of a detailed analysis of our protocols in the normal public key setting, but it is not hard to see that instantiating them with, for example, the Cramer-Shoup scheme [12] yields a protocol with a computational increase of a factor only around 3 when compared with HMQV [24].

References

1. Masayuki Abe, Rosario Gennaro, Kaoru Kurosawa, and Victor 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, 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`.
3. Mihir 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.
4. Mihir Bellare, Alexandra Boldyreva, and Adriana 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.

5. Mihir Bellare, Ran Canetti, and Hugo Krawczyk. Pseudorandom functions revisited: The cascade construction and its concrete security. In *Proceedings of the* 37th *Annual Symposium on the Foundations of Computer Science*, pages 514–523. IEEE, 1996. Full paper available at `http://www-cse.ucsd.edu/users/mihir/papers/cascade.html`.
6. Mihir Bellare, Ran Canetti, and Hugo 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`.
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. Colin Boyd, Wenbo Mao, and Kenneth 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. Ran Canetti, Oded Goldreich, and Shai 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. Ran Canetti and Hugo 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. Ronald Cramer and Victor Shoup. Design and analysis of practical public-key encryption schemes secure against adaptive chosen ciphertext attack. *SIAM Journal on Computing*, 33(1):167–226, 2003.
13. Alexander 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. Alexander W. Dent. A note on game-hopping proofs. Cryptology ePrint Archive, Report 2006/260, 2006. `http://eprint.iacr.org/2006/260`.
15. 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.
16. Yevgeniy Dodis. *Exposure-Resilient Cryptography*. PhD thesis, Department of Electrical Engineering and Computer Science, Massachusetts Institute of Technology, August 2000. `http://theory.lcs.mit.edu/~yevgen/academic.html`.
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. Craig 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. Ik Rae Jeong, Jonathan Katz, and Dong Hoon 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. Eike 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. Eike Kiltz and David 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 [22] and [23].
22. Eike Kiltz and David 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`.
23. Eike Kiltz and David 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`.
24. Hugo 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.
25. 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.
26. N. Nisan and D. Zuckerman. Randomness is linear in space. *J. Comp. and Sys. Sci.*, 52(1):43–52, 1996.
27. R. Sakai, K. Ohgishi, and M. Kasahara. Cryptosystems based on pairing. In *2000 Symposium on Cryptography and Information Security (SCIS2000)*, January 2000.
28. Shengbao Wang, Zhenfu Cao, and Kim-Kwang Raymond Choo. New identity-based authenticated key agreement protocols from pairings (without random oracles). Cryptology ePrint Archive, Report 2006/446, 2006. `http://eprint.iacr.org/`.
29. Brent 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 [21, 22] 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 $\hat{e} : \mathbb{G}_1 \times \mathbb{G}_1 \to \mathbb{G}_T$. 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 $\hat{e} : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$. 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, $g_* = \psi(g)$. $\hat{e} : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$ is a bilinear map. $\mathsf{H} : \{0,1\}^n \to \mathbb{G}_1$ is the hash function originally defined in Waters' scheme [29] 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. $\mathsf{TCR}(\cdot)$ is a target collision resistant hash function (see Definition 4).

Figure 1 describes the modified version of each of the three suggested IB-KEMs: those by Kiltz [20], Kiltz and Galindo [21, 22] 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 is not consistent, and so checking for consistency can be incorporated into calculation of the key, as in the Kiltz–Galindo IB-KEMs.

The cost 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

Kiltz

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

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

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

$\mathsf{Dec}(pk, d, C)$ :
    Parse $C$ as $(c_1, c_2)$
    Parse $d$ as $(d_1, d_2, d_3)$
    $t = \mathsf{TCR}(c_1)$
    $v \in_{\mathrm{R}} \mathbb{Z}_p^*$
    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

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

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

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

$\mathsf{Dec}(pk, d, C)$ :
    Parse $C$ as $(c_1, c_2, c_3)$
    Parse $d$ as $(d_1, d_2)$
    $t = \mathsf{TCR}(c_1)$
    $v_1, v_2 \in_{\mathrm{R}} \mathbb{Z}_p^*$
    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

```
$\mathsf{KeyGen}(1^k)$ :
    $\alpha \in_{\mathrm{R}} \mathbb{Z}_p; g_1 = g_*^{\alpha} \in \mathbb{G}_1; u \in_{\mathrm{R}} \mathbb{G}_1;$
    $h_1, h_2, h_3 \in_{\mathrm{R}} \mathbb{G}_2; z_0 = \hat{e}(g_*, g);$
    $z_i = \hat{e}(g_*, h_i)$ for $1 \le i \le 3$;
    $\mathsf{H}' \in_{\mathrm{R}}$ 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 $(pk, \alpha)$

$\mathsf{KeyDer}(pk, \alpha, id)$ :
    $s_1, s_2, s_3 \in_{\mathrm{R}} \mathbb{Z}_p$
    $d_i = \left(h_i g^{-s_i}\right)^{1/(\alpha - id)}$
      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.

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

$\mathsf{Dec}(pk, d, C)$ :
    Parse $C$ as $(c_1, c_2, c_3, c_4)$
    Parse $d$ as $\{(s_i, d_i) : i \in (1, 2, 3)\}$
    $\beta = \mathsf{H}'(c_1, c_2, c_3)$
    $v \in_{\mathrm{R}} \mathbb{Z}_p^*$
    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 $\mathsf{Enc} + \mathsf{Dec}$ cost		56			82			57		

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

of $\mathbb{G}_T$. The number of operations required by each scheme are similar to those 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 [25], 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. Unfortunately, it is unclear exactly what effect the universal hash function has in Gentry's proof. In particular, it would be nice to know whether the universal hash function may be replaced with an almost universal one, and how close to universal that function must be. If this were the case, it may be possible to apply the work of Dodis et al. [15] to use CBC-MAC, keyed cascade chaining, or HMAC instead of the universal hash function, and so reduce the size of Gentry's system parameters substantially.

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 Proofs of Protocols

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:

$$\begin{aligned}
\Pr[\sigma_{i+1}] &= \Pr[\sigma_{i+1}|E]\Pr[E] + \Pr[\sigma_{i+1}|\neg E]\Pr[\neg E] && (8)\\
&= \frac{1}{2}\Pr[E] + \Pr[\sigma_i|\neg E]\Pr[\neg E] && (9)\\
\Pr[\sigma_i] - \Pr[\sigma_{i+1}] &= \Pr[\sigma_i|E]\Pr[E] - \frac{1}{2}\Pr[E] && (10)\\
|\Pr[\sigma_i] - \Pr[\sigma_{i+1}]| &\le \frac{1}{2}\Pr[E] && (11)
\end{aligned}$$

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:

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

C.2 Proof of Theorem 1

Throughout the proof, we call each session to be activated an oracle. We denote the oracles with which $\mathcal{B}$ interacts Π_X^i where X is the name of a party and i is the number of the oracle. We number the oracles such that Π_X^i is the i^{th} oracle created by $\mathcal{B}$ out of all oracles created by $\mathcal{B}$ (i.e. if Π_X^i and Π_Y^j 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 $\mathcal{B}$.

The first part of the proof uses the following games.

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 a collision in the collision resistant hash function $\mathsf{CR}(\cdot)$ occurs, the protocol halts and $\mathcal{B}$ returns a random bit.

Game 2. 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 and $\mathcal{B}$ returns a random bit.

Game 3. 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 test oracle, the protocol halts and $\mathcal{B}$ fails and outputs a random bit. We denote the output message of the target oracle with C^*, the target oracle's owner with T and the target oracle's intended partner with T^*.

Game 4. In this game, a random value $K'^* \in_{\mathrm{R}} \mathbb{K}$ 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. When the test session query is made to the target oracle, if $b = 1$, a random key from $\mathbb{U}_2$ is returned. Otherwise, K'^* is used instead of K'_T in calculation of the key returned. Apart from the above modifications, this game is the same as the previous one.

Game 5. 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 6. 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 3: In Game 1, if a collision in the hash function $\mathsf{CR}(\cdot)$ is detected and the protocol halts, then it is possible to use $\mathcal{B}$ to break the target collision resistance of the hash function. Since $\mathbf{Adv}^{\mathrm{cr}}_{\mathsf{CR},\mathcal{A}}(k)$ is the probability that an adversary $\mathcal{A}$ finds a collision, from (11) we have:

$$|\Pr[\sigma_0] - \Pr[\sigma_1]| \leq \frac{1}{2}\mathbf{Adv}^{\mathrm{cr}}_{\mathsf{CR},\mathcal{A}}(k) \tag{18}$$

and this can be used to bound τ_0 as follows:

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

$$\leq \mathbf{Adv}^{\mathrm{cr}}_{\mathsf{CR},\mathcal{A}}(k) + \tau_1 \tag{20}$$

To analyse Game 2, we find the probability of two or more sessions outputting the same message, which we label p_{sameMsg}. We begin with the Taylor approximation of e^{-x} and then use it in the analysis.

$$e^{-x} = 1 - x + \frac{x^2}{2!} - \dots \tag{21}$$

$$e^{-x} > 1 - x \quad \text{when } x > 0 \tag{22}$$

$$1 - x > 1 - 2x + \frac{(2x)^2}{2!} > e^{-2x} \text{ when } \tfrac{1}{2} > x > 0 \tag{23}$$

$$1 - p_{\mathrm{sameMsg}} = 1 \cdot \left(1 - \frac{1}{p}\right) \cdot \left(1 - \frac{2}{p}\right) \cdots \left(1 - \frac{n_{\mathrm{orac}} - 1}{p}\right) \tag{24}$$

$$> 1 \cdot e^{-2\frac{1}{p}} \cdot e^{-2\frac{2}{p}} \cdots e^{-2\frac{n_{\text{orac}}-1}{p}} \text{ if } \tfrac{1}{2} > \tfrac{n_{\text{orac}}-1}{p} > 0 \quad (25)$$

$$= e^{-\frac{n_{\text{orac}}(n_{\text{orac}}-1)}{p}} \quad (26)$$

$$> 1 - \frac{n_{\text{orac}}(n_{\text{orac}}-1)}{p} > 1 - \frac{n_{\text{orac}}^2}{p} \quad (27)$$

$$|\Pr[\sigma_1] - \Pr[\sigma_2]| < \frac{n_{\text{orac}}^2}{2p} \quad \text{when } \tfrac{1}{2} > \tfrac{n_{\text{orac}}-1}{p} > 0 \quad \text{from (11)} \quad (28)$$

This can be used to bound τ_1 as follows:

$$\tau_1 = |2\Pr[\sigma_1] - 1| \leq 2\left(|\Pr[\sigma_1] - \Pr[\sigma_2]| + \left|\Pr[\sigma_2] - \frac{1}{2}\right|\right) \quad (29)$$

$$\leq \frac{n_{\text{orac}}^2}{p} + \tau_2 \quad (30)$$

In Game 3, the probability of the protocol halting due to an incorrect choice of m is $1 - \frac{1}{n_{\text{orac}}}$. 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:

$$\tau_3 = \frac{1}{n_{\text{orac}}}\tau_2 \quad \Rightarrow \quad n_{\text{orac}}\tau_3 = \tau_2 \quad (31)$$

and (20) and (30) give $\quad \tau_0 \leq \mathbf{Adv}^{\text{cr}}_{\mathsf{CR},\mathcal{A}}(k) + \frac{n_{\text{orac}}^2}{p} + n_{\text{orac}}\tau_3 \quad (32)$

Analysis of Game 4: 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 3, but if $\mathcal{A}$ receives a random IB-KEM key, the view of $\mathcal{B}$ is the same as in Game 4. 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 the collision resistant hash function $\mathsf{CR}(\cdot)$, and 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 3, 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 as follows. A corrupt query on some identity e may be answered with $\mathcal{O}_{\mathsf{KeyDer}}(e)$(recall that no corrupt query is allowed by $\mathcal{B}$ on T^* since this is the partner to the test session). $\mathcal{A}$ 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). A message M to any party (including T^*) with identity e may be decrypted using $\mathcal{O}_{\mathsf{Dec}}(e, M)$ 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 $\Pr[\sigma_3]$ when K'^* is the real key for the IB-KEM message, and $1-\Pr[\sigma_4]$ when K'^* is not the key for the IB-KEM message. We can then find that:

$$\mathbf{Adv}_{\mathcal{E},\mathcal{A}}^{\mathrm{ib-kem-cca}}(k) = \left| \left(\frac{1}{2} \left(\Pr[\sigma_3] + 1 - \Pr[\sigma_4] \right) \right) - \frac{1}{2} \right| \tag{33}$$

$$= \frac{1}{2} \left(\Pr[\sigma_3] - \Pr[\sigma_4] \right) \tag{34}$$

$$\tau_3 = |2\Pr[\sigma_3] - 1| \leq |2\Pr[\sigma_3] - 2\Pr[\sigma_4]| + |2\Pr[\sigma_4] - 1| \tag{35}$$

$$\tau_3 \leq 4\mathbf{Adv}_{\mathcal{E},\mathcal{A}}^{\mathrm{ib-kem-cca}}(k) + \tau_4 \tag{36}$$

Analysis of Game 5 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 4 or 5. $\mathcal{D}$ 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 5, 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 $\frac{1}{2}\left(\Pr[\sigma_4] + 1 - \Pr[\sigma_5]\right)$. By the security of the randomness extraction function, we have:

$$\epsilon \geq |2\Pr[\mathcal{D} \text{ correct}] - 1| = |\Pr[\sigma_4] - \Pr[\sigma_5]| \tag{37}$$

$$\tau_4 = |2\Pr[\sigma_4] - 1| \tag{38}$$

$$\leq |2\Pr[\sigma_4] - 2\Pr[\sigma_5]| + |2\Pr[\sigma_5] - 1| \tag{39}$$

$$\leq 2\epsilon + \tau_5 \tag{40}$$

Analysis of Game 6 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 5 or 6. $\mathcal{D}^{'}$ 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 5, 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 $\frac{1}{2}\left(\Pr[\sigma_5]+1-\Pr[\sigma_6]\right)$. By the security of the randomness expansion function we have:

$$\epsilon' \geq \left|2\Pr[\mathcal{D}^{'} \text{ correct}]-1\right| = |\Pr[\sigma_5]-\Pr[\sigma_6]| \tag{41}$$

$$\begin{aligned}
\tau_5 &= |2\Pr[\sigma_5]-1| && (42)\\
&\leq |2\Pr[\sigma_5]-2\Pr[\sigma_6]| + |2\Pr[\sigma_6]-1| && (43)\\
&\leq 2\epsilon' + \tau_6 && (44)
\end{aligned}$$

Combining Results In Game 6, let us denote the key returned in the test session query with $R_1 \oplus \mathsf{Expd}_{K''_{T^*}}(s')\left[\oplus\mathsf{Expd}_{K''_{TT^*}}(s')\right]$ 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 $b=1$. Therefore, we wish to show that $\mathcal{B}$ is unable to find any information about R_1 (apart from what it might gain in the response to the test session query).

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, unless there is more than one such session, in which case none of the sessions would match, and a reveal key query would be allowed. However, Game 2 causes $\mathcal{B}$ to output a random bit when there is more than one such session. Therefore, if $\mathcal{B}$ is to have any advantage guessing b, there will only be one session owned by T^* with intended partner T that received C^* as input and has the same session identifier as the test session, and there will not be any session at T that outputs C^* for T^*, apart from the test session. Hence, $\mathcal{B}$ can gain no information about R_1, and so $\mathcal{B}$ can gain no information about b in Game 6.

Therefore, we claim that $\tau_6 = 0$ since it is impossible for $\mathcal{B}$ to have any advantage in Game 6, because the answer to the test session query is always a random value chosen independently of all other values in $\mathcal{B}$'s view. Hence, we may combine the results in equations (32), (36), (40) and (44) to conclude:

$$\tau_0 \leq \mathbf{Adv}^{\mathrm{cr}}_{\mathsf{CR},\mathcal{A}}(k) + \frac{n_{\mathrm{orac}}^2}{p} + 2n_{\mathrm{orac}}\left(2\mathbf{Adv}^{\mathrm{ib-kem-cca}}_{\mathcal{E},\mathcal{A}}(k) + \epsilon + \epsilon'\right) \tag{45}$$

C.3 Proof of Theorem 2:

The proof can be considered in two parts; the first part proves the security of Protocol 2 when the test session owner is not corrupted. This part of the proof is essentially identical to the proof of Theorem 1. The second part proves the security of Protocol 2 when the test session owner is corrupted (in this case, we require the test session to have a matching session by the time $\mathcal{B}$ finishes).

The second part of the proof assumes that the test session has a matching session by the time the adversary, $\mathcal{B}$, outputs its guess of the bit b. It allows any party to be corrupted. It considers the following 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 $\mathbb{K}$ is returned.

Game 1. This game is the same as the previous one, except that two parties, T and T^* are chosen at random, and if T is not the owner of the test session or T^* is not the test session's intended partner, the protocol halts and $\mathcal{B}$ fails and outputs a random bit. 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 $j^{*\text{th}}$ sessions respectively. if the j^{th} session at T is not the test session or if the output of the $j^{*\text{th}}$ 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 set to a random value from $\mathbb{K}$, the keyspace of the KEM. The test session's matching session has its key is set to the same random value (i.e. a random test session key is always returned, no matter what the value of b).

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 $\mathcal{B}$.

When the j^{th} and the $j^{*\text{th}}$ sessions are activated, $\mathcal{A}$ uses its inputs f^a and f^b instead of the values X_T and X_{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 its third input (which is either f^c or f^{ab}) in place of X_{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.

The probability that $\mathcal{A}$ will not have to abort the protocol as described in Game 1 is $\frac{1}{n_{\text{orac}}^2}$. Furthermore, in Game 1, $\mathcal{B}$ is able to gain no advantage since a random key is always returned in answer to the test session. Hence, by (17) and using a similar logic to that shown in (41) to (44) we have that:

$$\sigma_0 \leq 2n_{\text{orac}}^2 \epsilon'' \tag{46}$$

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:

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

and so we can combine (45) and (46) to find:

$$\begin{aligned}
\mathbf{Adv}^{\mathrm{sk}}_{\mathcal{B}}(k) \le \max\Bigg(&\mathbf{Adv}^{\mathrm{cr}}_{\mathsf{CR},\mathcal{A}}(k) + \frac{n^2_{\mathrm{orac}}}{p} \\
&+ 2n_{\mathrm{orac}}\left(2\mathbf{Adv}^{\mathrm{ib-kem-cca}}_{\mathcal{E},\mathcal{A}}(k) + \epsilon + \epsilon'\right), 2n^2_{\mathrm{orac}}\epsilon''\Bigg)
\end{aligned}$$

C.4 On Key Construction and Proof Technicalities

When a particular IB-KEM is used in conjunction with the protocol, it may be tempting to try to achieve forward security 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 $\mathcal{A}$ against the IB-KEM would no longer be able to generate the proper test session key when $\mathcal{B}$ 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.

If we try setting $K''_A = \mathsf{Exct}_\kappa(K'_A||K'_B)$, and changing K''_B similarly, then the test session key returned to $\mathcal{B}$ is $\mathsf{Expd}_{\mathsf{Exct}_\kappa(K'_T||K'_B)}(s')$ where $\mathcal{B}$ may know K'_B. However, if $\mathcal{B}$ uses the test session output as input to other sessions, $\mathcal{B}$ may learn values K'_i, $\mathsf{Expd}_{\mathsf{Exct}_\kappa(K'_T||K'_i)}(s'_i)$ for many i (since we allow $\mathcal{B}$ to corrupt the owner of the test session). However, 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.