New Generic Constructions of Public Key Encryption from Identity Based Encryption

Palash Sarkar and Sanjit Chatterjee

Applied Statistics Unit
Indian Statistical Institute
203, B.T. Road, Kolkata
India 700108.
e-mail:{palash,sanjit_t}@isical.ac.in

Abstract. Canetti-Halevi-Katz (CHK) introduced the idea of generically converting a CPA-secure identity based encryption (IBE) protocol into a CCA-secure public key encryption (PKE) protocol. The efficiency of the CHK construction was improved by Boneh-Katz (BK) by using a MAC algorithm to replace one-time signatures. In this paper, we present two new constructions for obtaining a hybrid PKE from an IBE. The first construction is obtained by redrawing the abstraction boundary of the BK construction to obtain an explicit hybrid protocol. This allows us to use a (possibly single-pass) authenticated encryption (AE) protocol to instantiate the symmetric component. In the second protocol, we show that the MAC component in the BK construction can be removed. Instead, the message is authenticated differently through the process of deriving the identity of the underlying identity based key encapsulation mechanism (IBKEM). The second protocol requires the identity based component to be secure in the full model as opposed to the first protocol which is required to be secure only in the selective-ID model. As an additional contribution, we also show how to use AE in the direct construction of hybrid PKE due to Kurosawa-Desmedt (KD).
Keywords: identity based encryption, public key encryption.

1 Introduction

Public key encryption protocols are of central importance in modern cryptography. A major ongoing theme of research is to obtain and to understand efficient methods for building new PKE protocols. Given the importance of PKE in modern cryptography and the amount of time (three decades) for which the concept has been around, a surprisingly few actual constructions are known. This is perhaps due to the inherent difficulty of constructing such schemes.

An important early PKE which has a proof of security is the RSA-OAEP [4, 11]. However, the proof of this construction requires the random oracle assumption. The only known direct construction of PKE without using the random oracle heuristic is the Cramer-Shoup (CS) [10] construction and its variants. From an application point of view, it is important to construct hybrid PKE protocols, i.e., protocols, which use the public key component to encapsulate a secret key and the main "bulk" message is encrypted using a symmetric encryption algorithm using the secret key. Kurosawa-Desmedt [14] described a more efficient variant of hybrid CS protocol.

One of the most important topics of current research is that of identity based encryption (IBE) [18, 5]. This is public key encryption scheme where the public key can be any string. Perhaps surprisingly, it was shown by Canetti-Halevi-Katz [8] and later by Boneh-Katz [6] that a weakly secure IBE can be used to *generically* build a PKE which is secure in the strongest sense. Later work by Boyen-Mei-Waters [7] have shown how to improve the efficiency of such constructions for particular IBE schemes. The constructions of BMW are non-generic.

Our Contributions: In this paper, we revisit the problem of generic construction of PKE from IBE. As mentioned earlier, the first such construction is the CHK construction, which was based on the use of one-time signatures. This makes the conversion quite inefficient. Efficiency improvements were made by the BK construction which used a message authentication code (MAC) algorithm to replace the one-time signature. The BK construction is based on an abstraction of a special kind of commitment scheme which can be instantiated by a universal one-way hash function (UOWHF) and a family of pairwise independent hash functions.

We "unfold" and rewrite the BK construction, i.e., we do away with the abstraction of the special commitment scheme and write the entire protocol in terms of the constituent primitives. This allows us to better understand how and why the protocol works. Based on this understanding we suggest two new protocols, $\boldsymbol{\mathcal{P}}_1$ and $\boldsymbol{\mathcal{P}}_2$, for generically converting an IBE to a PKE. Both the protocols are hybrid constructions.

In the first protocol, $\boldsymbol{\mathcal{P}}_1$, we make explicit use of an identity based key encapsulation mechanism (IBKEM). This enables a clearer identification of the roles of the symmetric components in the protocol. We find that two of these – encryption and authentication – can be combined using a single known primitive called authenticated encryption (AE). In the process, we realized that it is also easy to use AE in the hybrid KD protocol and we have described the KD protocol using AE. We note that the use of AE in the context of stateful hybrid PKE has been earlier mentioned in [3].

One advantage of using AE is that there are known single pass AE protocols [17, 13, 12, 9], i.e., both encryption and authentication can be performed in a single-pass over the message. This leads to efficiency improvement in the symmetric component by a factor of two. Further, the ability for performing single-pass AE allows for on-line encryption of the message, which can be useful when secure buffer size is limited.

Returning to our first protocol, we also do away with the pairwise independent hash family. In the BK protocol, this was used to obtain the secret key for the MAC algorithm. Instead we use a computationally secure KDF to derive the secret key for the AE algorithm from the secret key produced by the IBKEM protocol. A computationally secure KDF will be more efficient than a pairwise independent hash family.

In the BK protocol, the identity to be used with the IBKEM protocol is derived by applying a UOWHF from a random string x. In our first protocol we retain this feature. But we realized that if M is a message it is possible to apply a hash function H_s to the string $M \circ x$ to derive the identity y. Applied in such a manner, the string y could also potentially play the role of a tag, by authenticating the message and the random string x. For this, we need to relax the requirement on H_s from a UOWHF to a partial UOWHF (but not quite a collision resistant hash function).

In any case, using y as a tag means that we can do away with the explicit MAC generation algorithm of the BK protocol (or the authentication property of our first protocol). One advantage of doing away with the MAC algorithm is that the ciphertext overhead reduces which could be useful for some applications.

Another consequence is that we need the IBKEM to satisfy the notion of security in the full model where the challenge identity can be chosen adaptively based on earlier key extraction queries and their responses. On the other hand, the CHK, BK as well as protocol $\boldsymbol{\mathcal{P}}_1$ require only selective-ID security, where the adversary for the IBKEM protocol has to commit to the challenge identity even before the protocol is set-up. The selective-ID model is weaker than the full model. Using the full model allows us to remove one component from the resulting PKE protocol. This aspect has perhaps not been noticed earlier.

From a more practical point of view, the net effect of our constructions is to improve the efficiency of the BK construction – in one case by improving the efficiency of the symmetric components by a factor of two and in the other case by reducing ciphertext expansion. In our opinion, equally important is the fact that our constructions lead to a better understanding of the interplay of different components and their security requirements.

Another aspect is that of the security proof. The proof given in [6] is quite complex and even though we can understand parts of it, the entire proof is very difficult to grasp. On the other hand, we provide game sequence style proofs for our protocols. Needless to say, some of the ideas in the proofs are borrowed from the Boneh-Katz paper though we have put them together according to our understanding. One central point highlighted in [6] is an apparent circularity in the construction which complicates the proof. A circularity is also identified by [2] in their PKE protocol and a technique of deferred analysis in the game sequence style of proofs is used to tackle the situation. In our proof, we have also used this deferred analysis to overcome the apparent circularity in our construction.

Recent Related Work. A recent work [1] also revisits the problem of constructing a PKE from an IBE. The approach in [1] is different from ours and we briefly describe the differences below.

Two constructions are presented in [1]. The first construction considers a partitioned IBKEM, a notion which is also introduced in [1]. Informally, a partitioned IBKEM is one where the encapsulation procedure can be divided into two parts – the first part produces c_1 and does not depend on the identity. The second part depends on the identity. Given a partitioned IBKEM and a chameleon hash function, [1] shows how to construct a tag-KEM, which is a variant of KEM introduced in [2]. Though conceptually interesting, the use of a chameleon hash function leads to an additional cost of one multi-exponentiation per encapsulation and decapsulation.

The second construction requires a partitioned IBKEM which is secure in the full model (i.e., where the adversary can adaptively choose the challenge identity) with an additional security requirement. During the challenge stage, the adversary is given the first part c_1 of the partitioned IBKEM as well as the session key K_b before it produces the challenge identity. Given such an IBKEM, [1] shows how to obtain a tag-KEM where there is no overhead in ciphertext expansion over and above the underlying IBKEM.

Both the constructions in [1] are based on a different approach to the problem. As such, it is difficult to compare the constructions to the ones we obtain. We feel that the constructions in [1] as well as the constructions in this paper are interesting and of independent interest.

2 Components

We need several components to build a PKE protocol. Most of these are standard and due to lack of space we describe them along with their security requirement in Appendix A. In our second PKE protocol, we require an assumption on a hash function which is perhaps not well known. So, we describe the hash function assumptions here.

2.1 Hash Functions

We will require hash functions with certain security properties in our constructions. These are defined below.

Universal One-Way Hash Functions (UOWHF). A keyed family $H_{s \in \mathcal{S}}$ (for all s, H_s has the same domain and range) of functions is said to be a UOWHF if the following task is computationally infeasible for an adversary: The adversary outputs an x from the domain, is given a random key s and then has to output $x' \neq x$ such that $H_s(x) = H_s(x')$. We say that H_s is an (ϵ, t)-secure UOWHF if the maximum advantage of any adversary running in time t in winning the above game is ϵ.

Partial UOWHF (PUOWHF). In this case, the security requirement is the following. The adversary commits to a string x and is then given a randomly chosen s and has to find M, M', x' such that $M \circ x \neq M' \circ x'$ but $H_s(M \circ x) = H_s(M' \circ x')$.

In a UOWHF, the adversary has to commit one input before learning the hash function for which it has to find a collision. On the other extreme, in a collision resistant hash function (CRHF), the adversary gets to know the hash function first and then has to find the colliding inputs. In a PUOWHF, the adversary commits to a portion of one input before knowing the hash function for which it has to find a collision. Thus, the security requirement of a PUOWHF is somewhere between that of a UOWHF and a CRHF. This notion was introduced by Mironov [15] and to the best of our knowledge has not been earlier used for building any other cryptographic protocol.

We will be using the above security notions in our proofs. However, we would like to point out that the security proofs can be obtained with much weaker assumptions on the hash functions. These will be described in the full version of the paper.

3 New Constructions

The Boneh-Katz construction is described in terms of a special commitment scheme which is then instantiated with a UOWHF H_s and a hash function h from a pairwise independent hash family. In Figure 1, we describe the "unfolded" BK construction, i.e., the BK construction written directly in terms of H_s and h. We feel that this helps to obtain a clearer understanding of how the identity is obtained and bound to the ciphertext.

3.1 Protocol $\mathcal{P}_1$

Our first protocol (Figure 2) is obtained from the unfolded BK protocol by redrawing the abstraction boundary. The BK protocol can use hybrid encryption, in which case the underlying IBE protocol has to be a hybrid protocol. We separate out this IBE protocol into an IBKEM protocol along with a symmetric encryption protocol.

By making this explicit separation, we can consider the requirements on the symmetric components more carefully. These requirements are twofold – to perform encryption and also to authenticate. Both these requirements can be combined into a single primitive – that of authenticated encryption. The ability to isolate this primitive is important, since an AE protocol can potentially be almost two times more efficient [17, 13, 12, 9] than separate applications of encryption and authentication algorithms.

The other aspect of this separation is that we can now generate the key-IV pair for the AE protocol by applying a computationally secure KDF on the secret key returned by the encapsulation algorithm of the IBKEM protocol. This allows us to do away with the family of pairwise independent hash functions.

Theorem 1. *The PKE protocol described in Figure 2 is secure against adaptive chosen ciphertext attacks.*

Fig. 1. The "unfolded" Boneh-Katz construction.

$\underline{\mathsf{PKE.SetUp}}$ 1. Run $\mathsf{IBE.SetUp}$ to obtain (PP,msk). 2. Randomly choose h from a family of pairwise independent hash functions. 3. Randomly choose H_s from a UOWHF. **Public Key:** PP, h, H_s. **Secret Key:** msk.	
$\underline{\mathsf{PKE.Encrypt}}$ input: message M. 1. Generate random x. 2. $\mathrm{com} = H_s(x)$, $\mathrm{dec} = x$. 3. $C = \mathsf{IBE.Encrypt}(\mathrm{PP}, \mathrm{com}, M \circ \mathrm{dec})$. 4. $r = h(x)$. 5. $\mathrm{tag} = \mathsf{MAC.Gen}_r(C)$. 6. Output $(\mathrm{com}, C, \mathrm{tag})$.	$\underline{\mathsf{PKE.Decrypt}}$ input: $(\mathrm{com}, C, \mathrm{tag})$. 1. $\mathrm{sk} = \mathsf{IBE.KeyGen}(\mathrm{PP}, \mathrm{com}, \mathrm{msk})$. 2. $M \circ \mathrm{dec} = \mathsf{IBE.Decrypt}(\mathrm{PP}, \mathrm{com}, \mathrm{sk}, C)$. (This may abort and return $\perp$.) 3. $r = h(\mathrm{dec})$. 4. If $H_s(x) \neq \mathrm{com}$ return $\perp$. 5. If $\mathsf{MAC.Verify}_r(C, \mathrm{tag})$ is false, then return $\perp$. 6. Return M.

Fig. 2. From IBKEM to PKE: Protocol $\boldsymbol{\mathcal{P}}_1$

$\underline{\mathsf{PKE.SetUp}}$ 1. Run $\mathsf{IBKEM.SetUp}$ to obtain (PP,msk). 2. Randomly choose H_s from a UOWHF. **Public Key:** PP, H_s. **Secret Key:** msk.	
$\underline{\mathsf{PKE.Encrypt}}$ input: message M. 1. Generate random x. 2. $y = H_s(x)$. 3. $(K, k) = \mathsf{IBKEM.Encap}(\mathrm{PP}, y)$. 4. $(\mathrm{IV}, l) = \mathsf{KDF}(k)$. 5. $(C, \mathrm{tag}) = \mathsf{AE.Encrypt}_l(\mathrm{IV}, M \circ x)$. 6. Output (y, K, C, tag).	$\underline{\mathsf{PKE.Decrypt}}$ input: (y, K, C, tag). 1. $\mathrm{sk} = \mathsf{IBKEM.KeyGen}(\mathrm{PP}, y, \mathrm{msk})$. 2. $k = \mathsf{IBKEM.Decap}(\mathrm{PP}, y, \mathrm{sk}, K)$. (This may abort and return $\perp$.) 3. $(\mathrm{IV}, l) = \mathsf{KDF}(k)$. 4. $M \circ x = \mathsf{AE.Decrypt}_l(\mathrm{IV}, C, \mathrm{tag})$ (This may abort and return $\perp$.) 5. If $H_s(x) \neq y$ return $\perp$. 6. Output M.

Fig. 3. From IBKEM to PKE: Protocol $\boldsymbol{\mathcal{P}}_2$

$\underline{\mathsf{PKE.SetUp}}$ 1. Run $\mathsf{IBKEM.SetUp}$ to obtain (PP,msk). 2. Randomly choose H_s from a UOWHF. **Public Key:** PP, H_s. **Secret Key:** msk.	
$\underline{\mathsf{PKE.Encrypt}}$ input: message M. 1. Generate random x. 2. $y = H_s(M \circ x)$. 3. $(K, k) = \mathsf{IBKEM.Encap}(\mathrm{PP}, y)$. 4. $(\mathrm{IV}, l) = \mathsf{KDF}(k)$. 5. $(C, \mathrm{tag}) = \mathsf{Sym.Encrypt}_l(\mathrm{IV}, M \circ x)$. 6. Output (y, K, C).	$\underline{\mathsf{PKE.Decrypt}}$ input: (y, K, C). 1. $\mathrm{sk} = \mathsf{IBKEM.KeyGen}(\mathrm{PP}, y, \mathrm{msk})$. 2. $k = \mathsf{IBKEM.Decap}(\mathrm{PP}, y, \mathrm{sk}, K)$. (This may abort and return $\perp$.) 3. $(\mathrm{IV}, l) = \mathsf{KDF}(k)$. 4. $M \circ x = \mathsf{Sym.Decrypt}_l(\mathrm{IV}, C)$. (This may abort and return $\perp$.) 5. If $H_s(M \circ x) \neq y$ return $\perp$. 6. Output M.

Proof: The proof is using a game sequence.

Game 0: This is the usual adversarial game for PKE. In this game, the PKE is set-up according to the protocol and the adversary $\mathcal{A}$ is given the PP. Decryption queries are answered using the secret key. Suppose that M_β is encrypted in the challenge stage, where β is a random bit. Also, let $(y^*, K^*, C^*, \text{tag}^*)$ be the challenge ciphertext. All quantities related to the challenge ciphertext will be denoted by a super-script * (such as x^*). Since x^*, y^*, K^* does not depend on the adversary's input, we will assume that these are chosen during the set-up stage. By the rules of the game, for any adversarial query (y, K, C, tag), we must have $(y, K, C, \text{tag}) \neq (y^*, K^*, C^*, \text{tag}^*)$. In the following, we will be assuming this condition without mention.

At the end of the game, the adversary outputs the guess β'. Let X_0 be the event $\beta = \beta'$. In the following, we will be defining further games and X_i will be the event that $\beta = \beta'$ in Game i. We assume that the adversary playing Game 0 maximizes the advantage among all adversaries with similar resources. Thus, we have

$$\epsilon_{pke} = \left| \mathsf{Prob}[X_0] - \frac{1}{2} \right|.$$

Game 1: In this game, we make the following modification. If the adversary ever makes a decryption query (y, K, C, tag) with $H_s(x) = y = y^* = H_s(x^*)$ and $x \neq x^*$, then this query is rejected. Note that since we are assuming that the simulator knows the secret key, it can decrypt (C, tag) to obtain x.

Let E_1 be the above event. If E_1 does not occur, then Games 0 and 1 are the same. If E_1 occurs, then the adversary has effectively broken the UOWHF property of H_s. To see this, consider the following adversary $\mathcal{B}$ which attacks the UOWHF property of H_s. $\mathcal{B}$ first chooses a random x^* and then is given the key s for H_s. Since s and x^* are independent random quantities, the order in which they are chosen does not matter. $\mathcal{B}$ now sets up the PKE, provides PP to $\mathcal{A}$ and answers decryption queries using msk. If at some point of time, $\mathcal{A}$ makes a query for which the corresponding x is not equal to x^* but $H_s(x) = H_s(x^*)$, then $\mathcal{B}$ aborts and outputs x, x^*. If no such query is made by $\mathcal{A}$, at the end of the game $\mathcal{B}$ reports failure. This shows that if E_1 occurs, then it is possible to break the UOWHF property of H_s. Hence, we have

$$|\mathsf{Prob}[X_0] - \mathsf{Prob}[X_1]| \leq \mathsf{Prob}[E_1] \leq \epsilon_{uowhf}.$$

Game 2: Let E_2 be the event that the adversary makes a query (y, K, C, tag) such that $y = y^*$, $x = x^*$ and $K = K^*$. All such queries are rejected. If E_2 does not occur, then Games 1 and 2 are the same and we have

$$|\mathsf{Prob}[X_1] - \mathsf{Prob}[X_2]| \leq \mathsf{Prob}[E_2].$$

Game 3: Let E_3 be the event that the adversary makes a query (y, K, C, tag) such that $y = y^*$, $x = x^*$ and $K \neq K^*$. All such queries are rejected. If E_3 does not occur, then Games 2 and 3 are the same and we have

$$|\mathsf{Prob}[X_2] - \mathsf{Prob}[X_3]| \leq \mathsf{Prob}[E_3].$$

At this point we will not upper bound $\mathsf{Prob}[E_2]$ or $\mathsf{Prob}[E_3]$. Instead, we will use the technique of "deferred analysis" from [2] and upper bound these two quantities later. Deferred analysis was used in [2] to tackle an apparent circularity in argument. It turns out that it is also applicable in the present context where there is a circular dependence among x^* and k^*.

Game 4: Note that by this game, we have introduced conditions such that any query with $y = y^*$ is rejected. Now we are in a position to attack the CPA-security of the IBKEM protocol. We make the following modification. During the challenge stage, the IBKEM protocol returns (K^*, k^*). We replace k^* by a random value from the same domain.

Claim: $|\mathsf{Prob}[X_3] - \mathsf{Prob}[X_4]| \leq \epsilon_{ibkem}$.

Proof: *(Of claim.)* From the adversary $\mathcal{A}$ attacking the CCA-security of the PKE protocol we can build an adversary $\mathcal{B}$ attacking the CPA-security of the IBKEM protocol in the following manner.

$\mathcal{B}$ invokes the set-up of the IBKEM protocol. Note that the IBKEM protocol is required to be secure in the selective-ID model. For this, the adversary (in this case $\mathcal{B}$) has to commit to the challenge identity before the protocol is set-up. To fulfill this requirement $\mathcal{B}$ first chooses a UOWHF H_s and a random x^* and computes $y^* = H_s(x^*)$. $\mathcal{B}$ commits to this y^* as the challenge identity. The simulator for the IBKEM protocol provides $\mathcal{B}$ with the PP and $\mathcal{B}$ passes these on to $\mathcal{A}$ along with the UOWHF H_s.

$\mathcal{A}$ now makes decryption queries of the type (y, K, C, tag). We can assume that none of the y's is equal to y^*. Hence, for any such query $\mathcal{B}$ can invoke the key extraction oracle of the IBKEM on y and obtain the corresponding key sk. Using sk, $\mathcal{B}$ can decrypt the query and return a proper answer to $\mathcal{A}$. Thus, all decryption queries are properly answered.

In the challenge stage, $\mathcal{B}$ is given two equal length messages M_0 and M_1 by $\mathcal{A}$. $\mathcal{B}$ invokes the challenge stage for the simulator of the IBKEM protocol and receives (K^*, k^*) corresponding to the already committed challenge identity y^*. $\mathcal{B}$ now chooses a random bit β and completes the rest of the computation and gives the challenge ciphertext $(y^*, K^*, C^*, \text{tag}^*)$. Finally $\mathcal{B}$ outputs whatever $\mathcal{A}$ outputs.

If the k^* provided to $\mathcal{B}$ is proper, then $\mathcal{A}$'s view is that of Game 3. On the other hand, if k^* is random, then $\mathcal{A}$'s view is that of Game 4. This proves the claim. □

Game 5: In this game, we replace (IV^*, l^*) obtained by applying KDF with a random pair (IV^*, l^*) from the respective domains. Thus,

$$|\mathsf{Prob}[X_4] - \mathsf{Prob}[X_5]| \leq \epsilon_{kdf}.$$

Note that IV^* and l^* are now random. The only use of these quantities is to encrypt M_β. If the adversary is able to guess β correctly, then it has broken the (one-time) CPA-security of the AE protocol. Thus,

$$\left|\mathsf{Prob}[X_5] - \frac{1}{2}\right| \leq \epsilon_{enc}.$$

We now take a more detailed look at Game 5 and bound the probability of the events E_2 and E_3.

Case of E_2: In this case, the adversary submits a query with $y = y^*$, $x = x^*$ and $K = K^*$. The condition $K = K^*$ implies that $k = k^*$ and $(\text{IV}, l) = (\text{IV}^*, l^*)$. On the other hand, since $(y, K, C, \text{tag}) \neq (y^*, K^*, C^*, \text{tag}^*)$ and $(y, K) = (y^*, K^*)$ we must have $(C, \text{tag}) \neq (C^*, \text{tag}^*)$. If $\mathsf{AE.Decrypt}_l(\text{IV}, C, \text{tag})$ does not return $\perp$, then the adversary has been able to produce a valid (C, tag) pair after seeing one encryption (C^*, tag^*) under a random $(\text{IV}^*, l^*) = (\text{IV}, l)$. This breaks the authorization security of the AE protocol. Hence,

$$\mathsf{Prob}[E_2] \leq \epsilon_{auth}.$$

Case of E_3: In this case, the adversary submits a query with $y = y^*$, $x = x^*$ and $K \neq K^*$. Let us informally consider what is happening in this case. First note that the adversary is given x^* only as part of C^* which is the encryption of $M_\beta \circ x^*$ using a random (IV^*, l^*). Second, since $K \neq K^*$, the pair (IV, l) will not be equal to (IV^*, l^*). Now $x = x^*$ means that the adversary after obtaining an encryption of x^* under a random (IV^*, l^*) is able to create a new encryption of x^* under a different (IV, l). Intuitively, the adversary should not be able to do this until it has broken the semantic security of the AE protocol and obtained x^* from C^*. Next we formalize this argument.

An adversary $\mathcal{B}$ attacking the CPA-security of AE is built in the following manner. It first sets up the IBKEM protocol and chooses the UOWHF H_s in the usual manner. $\mathcal{B}$ retains the master secret key and provides the PP and H_s to $\mathcal{A}$, the adversary for attacking the PKE. Then $\mathcal{B}$ executes Game 5. All decryption queries can be answered by $\mathcal{B}$.

Note that decryption queries of the form $y = y^*$ and $x = x^*$ (with or without $K = K^*$) can be made by $\mathcal{A}$ only in the second phase. This is because $\mathcal{A}$ gets to see the encryption of x^* only as part of the challenge ciphertext. Previous to this it has no information on x^* and hence in the first phase the probability that $x = x^*$ is $1/2^{|x^*|}$ which is negligible and is ignored.

During the challenge phase, $\mathcal{A}$ outputs two equal length messages M_0 and M_1. $\mathcal{B}$ chooses a bit β randomly and submits $M'_0 = M_\beta \circ x^*$ and $M'_1 = M_\beta \circ 0^{|x^*|}$ to the encryption oracle of the AE protocol which is instantiated by a random and unknown (to $\mathcal{B}$) (IV^*, l^*) pair. In return it receives (C^*, tag^*) which is an encryption of M'_γ where γ is a random bit. $\mathcal{B}$ passes (C^*, tag^*) to $\mathcal{A}$ along with y^* and K^*.

In the second phase, if $\mathcal{A}$ makes a query with $y = y^*$, $x = x^*$ and $K \neq K^*$ (which is event E_2), then $\mathcal{B}$ outputs 0 and aborts the game. If no such query is made by $\mathcal{A}$, then at the end of the game $\mathcal{B}$ outputs a random bit. Let γ' be the output of $\mathcal{B}$. Then

$$\begin{aligned}
\mathsf{Prob}[\gamma = \gamma'] &= \mathsf{Prob}[(\gamma = \gamma') \wedge (E_2 \vee \overline{E_2})] \\
&= \mathsf{Prob}[(\gamma = \gamma')|E_2]\mathsf{Prob}[E_2] + \mathsf{Prob}[(\gamma = \gamma')|\overline{E_2}](1 - \mathsf{Prob}[E_2]) \\
&= \mathsf{Prob}[E_2] + \frac{1}{2}(1 - \mathsf{Prob}[E_2]) \\
&= \frac{1}{2} + \frac{1}{2}\mathsf{Prob}[E_2].
\end{aligned}$$

Thus,

$$\mathsf{Prob}[E_2] = 2\left(\mathsf{Prob}[\gamma = \gamma'] - \frac{1}{2}\right) \leq 2\epsilon_{enc}.$$

Putting all these together, we get

$$\begin{aligned}
\epsilon_{pke} &= \left|\mathsf{Prob}[X_0] - \frac{1}{2}\right| \\
&\leq \epsilon_{uowhf} + \epsilon_{ibkem} + \epsilon_{kdf} + \epsilon_{auth} + 3\epsilon_{enc}.
\end{aligned}$$

This completes the proof. □

3.2 Protocol $\mathcal{P}_2$

The description of the second protocol for converting an IBE to PKE is given in Figure 3. In the BK protocol as well as protocol $\mathcal{P}_1$, the "identity" is generated by applying H_s to a random string x. In the second protocol, we derive the "identity" y by applying H_s to $M \circ x$, where M is the message. The y can then serve a dual role of authenticating the message M as well as being the

identity for the IBKEM protocol. Since y serves the purpose of a tag, we can do away with the explicit MAC algorithm.

Allowing the identity y to depend on the message M has a different consequence. If we consider the challenge identity y^*, then this is defined to be $y^* = H_s(M_\beta \circ x^*)$, where M_β is one of M_0 or M_1 provided by the adversary during the challenge stage of the PKE adversarial game. Since y^* now depends on M_β it is not possible to compute y^* before the IBKEM protocol is set-up. Consequently, we cannot use the selective-ID model of security for the IBKEM. Instead, we have to use the full model security, where the challenge identity can be chosen adaptively by the adversary. Viewed differently, use of the full model allows us to remove the MAC algorithm which is required if we use the selective-ID model. We consider this to be an interesting feature which has perhaps not been noticed earlier.

Theorem 2. *The protocol described in Figure 3 is secure against adaptive chosen ciphertext attacks.*

Proof: The proof is again using a game sequence. The sequence of games is similar to the sequence given in the proof of Theorem 1. Several modifications are required, which we describe below.

Game 0: This game is similar to Game 0 played by an adversary $\mathcal{A}$ in the proof of Theorem 1.

Game 1: This game is obtained by modifying Game 0 in the following manner. If the adversary ever submits a decryption query (y, K, C), such that $H_s(M \circ x) = y = y^* = H_s(M_\beta \circ x^*)$ and $M \circ x \neq M_\beta \circ x^*$, then the query is rejected. This means that the adversary has broken the partial UOWHF property of H_s. To see this consider the following adversary $\mathcal{B}$ attacking the partial UOWHF property of H_s.

$\mathcal{B}$ chooses a random x^* (which does not depend on $\mathcal{A}$'s queries) and is then given a random key s for the function H_s. Since x^* and s are independent random quantities, the order in which they are chosen does not matter. $\mathcal{B}$ then sets up the PKE, gives PP to $\mathcal{A}$ and keeps msk to itself. It answers decryption queries using msk. During the challenge stage, $\mathcal{A}$ provides M_0 and M_1 and $\mathcal{B}$ generates β randomly and gives $\mathcal{A}$ a proper encryption of $M_\beta \circ x^*$. At this point, y^* is set to be equal to $H_s(M_\beta \circ x^*)$. If during the game, the adversary outputs M, x such that $M \circ x \neq M_\beta \circ x^*$ but $H_s(M \circ x) = H_s(M_\beta \circ x^*)$, then $\mathcal{B}$ outputs M_β, M, x. If the adversary does not make any such query, then at the end of the game $\mathcal{B}$ reports failure. Clearly, if the adversary makes such a query, $\mathcal{B}$ is able to break the partial-UOWHF property of H_s. Hence, we have

$$|\mathsf{Prob}[X_0] - \mathsf{Prob}[X_1]| \leq \epsilon_{puowhf}.$$

Game 2: This game is obtained by modifying Game 1 in the following manner. Let E be the event that the adversary submits a query with $y = y^*$, $M \circ x = M_\beta \circ x^*$ and $K \neq K^*$. If E occurs, then the query is rejected. If E does not occur, then the Games 1 and 2 are same and hence,

$$|\mathsf{Prob}[X_1] - \mathsf{Prob}[X_2]| \leq \mathsf{Prob}[E].$$

Up to this point, decryption queries with $y = y^*$ are rejected if either (a) $M \circ x \neq M_\beta \circ x^*$ or (b) $M \circ x = M_\beta \circ x^*$ and $K \neq K^*$. We now argue that if $y = y^*$ and $M \circ x = M_\beta \circ x^*$, then K cannot be equal to K^*. If indeed $K = K^*$, then $k = k^*$ which implies $(\mathrm{IV}, l) = (\mathrm{IV}^*, l^*)$. This along with $M \circ x = M_\beta \circ x^*$ implies $C = C^*$. But then $(y, K, C) = (y^*, K^*, C^*)$ which means that the challenge ciphertext is queried to the decryption oracle. This is not allowed. Hence, we must have $K \neq K^*$.

Recall that in the proof of Theorem 1, the condition $K = K^*$ was possible. Tackling this case led to an attack on the authorization property of the underlying AE protocol. In the present protocol, since the authorization is taken care of by H_s itself, the case $K = K^*$ is naturally excluded.

The important aspect of Game 2 is that in this game all queries with $y = y^*$ are rejected.

Game 3: In this game we change k^* to be a random quantity from the appropriate domain. This change is the same as the change from Game 3 to Game 4 in the proof of Theorem 1.

As before, from $\mathcal{A}$ we can construct an adversary $\mathcal{B}$ attacking the IBKEM protocol. The difference is that in this case we do not require $\mathcal{B}$ to commit to the challenge identity for the IBKEM before the protocol is set-up. During the challenge stage, $\mathcal{A}$ submits two equal length messages M_0 and M_1. A random bit β is chosen and y^* is set to be equal to $H_s(M_\beta \circ x^*)$. The challenge stage for IBKEM is invoked with y^* as the challenge identity and $\mathcal{B}$ receives in return (K^*, k^*). The rest of the game proceeds in a standard manner. If k^* is proper, then $\mathcal{B}$ is playing Game 2 and if k^* is random, then $\mathcal{B}$ is playing Game 3. Hence, we have

$$|\mathsf{Prob}[X_2] - \mathsf{Prob}[X_3]| \leq \epsilon_{ibkem}.$$

Game 4: This tackles the security of the KDF protocol in the standard manner (i.e., choosing random values for (IV^*, l^*)) giving us

$$|\mathsf{Prob}[X_3] - \mathsf{Prob}[X_4]| \leq \epsilon_{kdf}.$$

In this game, the adversary is given an encryption of $M_\beta \circ x^*$ under a random (IV^*, l^*), which is not used for any other purpose. Hence, if the adversary wins this game then it also breaks the (one-time) semantic security of the symmetric encryption scheme. This shows

$$\left|\mathsf{Prob}[X_4] - \frac{1}{2}\right| \leq \epsilon_{enc}.$$

Now we turn to bound $\mathsf{Prob}[E]$. Exactly as in the case of event E_3 in the proof of Theorem 1, it can be shown that

$$\mathsf{Prob}[E] \leq 2\epsilon_{enc}.$$

Combining the above inequalities, we obtain

$$\epsilon_{pke} \leq \epsilon_{puowhf} + \epsilon_{ibkem} + \epsilon_{kdf} + 3\epsilon_{enc}.$$

This completes the proof. □

4 Using Authenticated Encryption in PKE

Kurosawa-Desmedt [14] presented a variant of the Cramer-Shoup protocol for hybrid PKE which reduces the number of exponentiations by one. In this section, we briefly show how to use an AE protocol in the KD construction. The potential advantage of doing this is that one can then use a single pass AE protocol [17, 13, 12, 9] which can improve the efficiency of the symmetric components by a factor of two. The algorithm is shown in Figure 4. Note that the public key part of the algorithm is the same as that of the KD protocol. A sketch of the security proof is given in Section B.

Fig. 4. Hybrid PKE using an AE protocol.

PKE.SetUp 1. g_1, g_2 are random generators of G. 2. Choose x_1, x_2, y_1, y_2 randomly from $\mathbb{Z}_q$. 3. Set $c = g_1^{x_1} g_2^{x_2}$ and $d = g_1^{y_1} g_2^{y_2}$. **Public Key:** g_1, g_2, c, d. **Secret Key:** x_1, x_2, y_1, y_2.	
PKE.Encrypt input: message M. 1. Generate random r in $\mathbb{Z}_q$. 2. $u_1 = g_1^r$, $u_2 = g_2^r$. 3. $\alpha = H_s(u_1, u_2)$. 4. $v = c^r d^{r\alpha}$. 5. $(\text{IV}, k) = \mathsf{KDF}(v)$. 6. $(C, \text{tag}) = \mathsf{AE.Encrypt}_k(\text{IV}, M)$. 7. Output $(u_1, u_2, C, \text{tag})$.	PKE.Decrypt input: $(u_1, u_2, C, \text{tag})$. 1. $\alpha = H_s(u_1, u_2)$. 2. $v = u_1^{x_1+\alpha y_1} u_2^{x_2+\alpha y_2}$. 3. $(\text{IV}, k) = \mathsf{KDF}(v)$. 4. $M = \mathsf{AE.Decrypt}_k(\text{IV}, C, \text{tag})$. (This may abort and return $\perp$.) 5. Output M.

4.1 Discussion

The recent work [3] introduces the interesting notion of stateful public key encryption. A variant of the KD algorithm is used to instantiate this primitive. The important point about this instantiation is that the hybrid encryption requires only one exponentiation. A consequence is that there is no randomness in the public key part. The entire randomness comes from the symmetric encryption algorithm. The security requirement on the symmetric component is strictly stronger than CCA-security and it has been mentioned in [3] that this security notion can be obtained using a encrypt-then-MAC construction. The possibility of using a (single-pass) AE protocol in the construction of stateful public key encryption has also been mentioned in [3].

We consider the difference in the possible use of an AE in a stateful PKE as opposed to a usual (stateless) PKE. To be used with a stateful PKE, the AE must be a probabilistic algorithm. Usual constructions of AE algorithms are deterministic. A conceivable way of converting a usual AE algorithm into one suitable for use with a stateful PKE is to choose the IV randomly. The security proofs for usual AE assumes that IVs are nonces, i.e., they will not be repeated. Choosing the IVs randomly adds an additive quadratic security degradation to the security bound. This, however, is not really significant. The main issue is that the randomly chosen IV has to be transmitted to the decryption end resulting in an overhead in ciphertext expansion which has been mentioned in [3].

In contrast, for usual hybrid PKE protocols, the randomness comes from the public key portion. This allows the symmetric component to be deterministic. In our description of the KD protocol using AE, we have derived the IV (and the secret key) of the AE protocol from the session key using a KDF. Consequently, the IV does not need to be transmitted to the decryption end.

To summarize, we would first like to mention that the basic idea of using AE in hybrid PKE was already observed in [3]. This was done in the context of stateful PKE. Using AE in usual stateless KD can be done more directly as has been described earlier. Since KD and AE are important protocols in their respective categories, we feel that the explicit description of the KD+AE protocol given in this paper can be useful to actual practitioners.

5 Conclusion

In this paper we have presented two new constructions for generically transforming a weakly secure IBE into a strongly secure PKE. Compared to the previous work due to Boneh-Katz this leads to efficiency improvements. We have also pointed out the application of AE in the hybrid PKE due to Kurosawa-Desmedt.

References

1. Masayuki Abe, Yang Cui, Hideki Imai, and Eike Kiltz. Efficient hybrid encryption from id-based encryption. Cryptology ePrint Archive, Report 2007/023, 2007. http://eprint.iacr.org/.
2. 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 Ronald Cramer, editor, *EUROCRYPT*, volume 3494 of *Lecture Notes in Computer Science*, pages 128–146. Springer, 2005.
3. Mihir Bellare, Tadayoshi Kohno, and Victor Shoup. Stateful public-key cryptosystems: how to encrypt with one 160-bit exponentiation. In Ari Juels, Rebecca N. Wright, and Sabrina De Capitani di Vimercati, editors, *ACM Conference on Computer and Communications Security*, pages 380–389. ACM, 2006.
4. Mihir Bellare and Phillip Rogaway. Random Oracles are Practical: A Paradigm for Designing Efficient Protocols. In *ACM Conference on Computer and Communications Security*, pages 62–73, 1993.
5. Dan Boneh and Matthew K. Franklin. Identity-Based Encryption from the Weil Pairing. *SIAM J. Comput.*, 32(3):586–615, 2003. Earlier version appeared in the proceedings of CRYPTO 2001.
6. Dan Boneh and Jonathan Katz. Improved Efficiency for CCA-Secure Cryptosystems Built Using Identity-Based Encryption. In Alfred Menezes, editor, *CT-RSA*, volume 3376 of *Lecture Notes in Computer Science*, pages 87–103. Springer, 2005.
7. Xavier Boyen, Qixiang Mei, and Brent Waters. Direct Chosen Ciphertext Security from Identity-Based Techniques. In Vijay Atluri, Catherine Meadows, and Ari Juels, editors, *ACM Conference on Computer and Communications Security*, pages 320–329. ACM, 2005.
8. Ran Canetti, Shai Halevi, and Jonathan Katz. Chosen-Ciphertext Security from Identity-Based Encryption. In Christian Cachin and Jan Camenisch, editors, *EUROCRYPT*, volume 3027 of *Lecture Notes in Computer Science*, pages 207–222. Springer, 2004.
9. Debrup Chakraborty and Palash Sarkar. A General Construction of Tweakable Block Ciphers and Different Modes of Operations. In Helger Lipmaa, Moti Yung, and Dongdai Lin, editors, *Inscrypt*, volume 4318 of *Lecture Notes in Computer Science*, pages 88–102. Springer, 2006.
10. Ronald Cramer and Victor Shoup. A Practical Public Key Cryptosystem Provably Secure Against Adaptive Chosen Ciphertext Attack. In Hugo Krawczyk, editor, *CRYPTO*, volume 1462 of *Lecture Notes in Computer Science*, pages 13–25. Springer, 1998.
11. Eiichiro Fujisaki, Tatsuaki Okamoto, David Pointcheval, and Jacques Stern. RSA-OAEP Is Secure under the RSA Assumption. In Joe Kilian, editor, *CRYPTO*, volume 2139 of *Lecture Notes in Computer Science*, pages 260–274. Springer, 2001.
12. Virgil D. Gligor and Pompiliu Donescu. Fast encryption and authentication: XCBC encryption and XECB authentication modes. In Mitsuru Matsui, editor, *FSE*, volume 2355 of *Lecture Notes in Computer Science*, pages 92–108. Springer, 2001.
13. Charanjit S. Jutla. Encryption Modes with Almost Free Message Integrity. In Pfitzmann [16], pages 529–544.
14. Kaoru Kurosawa and Yvo Desmedt. A New Paradigm of Hybrid Encryption Scheme. In Matthew K. Franklin, editor, *CRYPTO*, volume 3152 of *Lecture Notes in Computer Science*, pages 426–442. Springer, 2004.
15. Ilya Mironov. Hash Functions: From Merkle-Damgård to Shoup. In Pfitzmann [16], pages 166–181.
16. Birgit Pfitzmann, editor. *Advances in Cryptology - EUROCRYPT 2001, International Conference on the Theory and Application of Cryptographic Techniques, Innsbruck, Austria, May 6-10, 2001, Proceeding*, volume 2045 of *Lecture Notes in Computer Science*. Springer, 2001.
17. Phillip Rogaway. Efficient Instantiations of Tweakable Blockciphers and Refinements to Modes OCB and PMAC. In Pil Joong Lee, editor, *ASIACRYPT*, volume 3329 of *Lecture Notes in Computer Science*, pages 16–31. Springer, 2004.
18. Adi Shamir. Identity-Based Cryptosystems and Signature Schemes. In G. R. Blakley and David Chaum, editors, *CRYPTO*, volume 196 of *Lecture Notes in Computer Science*, pages 47–53. Springer, 1984.

Appendix

A Components

The protocols described in this paper are built from several kinds of cryptographic primitives. In this section, we briefly describe these as well as public key encryption and identity based encryption.

A.1 Public Key Encryption

This consists of three probabilistic algorithms SetUp, Encrypt and Decrypt. The runtime of these algorithms is upper bounded by some polynomial function of a security parameter.

Algorithm SetUp returns the public key (or public parameter) and the secret key of the system. Algorithm Encrypt takes as input a message and the public key and returns a ciphertext whereas algorithm Decrypt takes as input a ciphertext, the public parameters and the secret key and returns either the corresponding message or $\perp$ (indicating that the ciphertext is not valid).

Security of a PKE against adaptive chosen ciphertext attack (CCA-security) is defined through the following game. An instance of the PKE protocol is setup and the adversary is given the public key. The adversary provides ciphertexts and is given either the corresponding messages or $\perp$ as the case maybe. This is considered to be the adversary's interaction with a decryption oracle instantiated by the secret key corresponding to the public key given to the adversary. At some stage, the adversary outputs two equal length messages M_0 and M_1. A bit β is chosen at random and the adversary is given an encryption C^* of M_β. The adversary continues its interaction with the decryption oracle subject to the restriction that it cannot query the decryption oracle on C^*. At the end the adversary outputs a bit β'. The adversary's advantage in breaking the PKE is defined to be

$$\left|\mathsf{Prob}[\beta = \beta'] - \frac{1}{2}\right|.$$

We say that a PKE is (ϵ, t)-secure if the maximum advantage of any adversary running in time t in the above game is ϵ.

A.2 Identity Based Encryption

An IBE consists of four probabilistic algorithms – SetUp, KeyGen, Encrypt and Decrypt. The SetUp algorithm generates the public parameters PP for the PKG and the master secret key msk. KeyGen takes as input PP, msk and an identity y and produces as output a secret key d_y corresponding to this identity. Algorithm Encrypt takes as input PP, an identity and a message M and produces a ciphertext as output. Decrypt takes as input a ciphertext, PP, identity y and a secret key d_y corresponding to y and produces as output either the corresponding message or $\perp$ (indicating invalid ciphertext).

An identity based key encapsulation mechanism (IBKEM) is similar to the above. Instead of Encrypt and Decrypt an IBKEM has an encapsulation algorithm Encap and a decapsulation algorithm Decap. The Encap algorithm takes as input PP, an identity y and returns (K, k), where k is a secret key and K is an encapsulation of the secret key. The Decap algorithm takes as input PP, identity y, the secret key d_y corresponding to y and an encapsulation K and returns as output either the corresponding secret k or $\perp$ (indicating invalid encapsulation).

The security models for IBE and IBKEM are similar with small differences. Here we describe the security model for IBKEM, since we will base our constructions on IBKEM rather than KEM. Further, we describe security against adaptive chosen plaintext attacks (CPA-security) since this is what we will need. The challenge identity can itself be either chosen adaptively (full model) or has to be specified before the protocol is set-up (selective-ID model).

An adversary $\mathcal{A}$ attacking an IBKEM protocol in the selective-ID model outputs an identity y^* even before the protocol is setup. Based on y^*, the simulator sets up the protocol and provides the PP to the adversary. While doing this the simulator has to ensure that the PP is distributed randomly as specified in the protocol.

$\mathcal{A}$ is given access to a key extraction oracle instantiated by the msk corresponding to the PP. $\mathcal{A}$ can ask for the secret key of any identity $y \neq y^*$. At some stage, $\mathcal{A}$ decides to move to the challenge stage. In this stage, a pair (K^*, k_0) is generated for the identity y^*, where k_0 is a secret key and K^* is an encapsulation of k_0. Also, a random quantity k_1 is chosen from the domain of k_0. A random bit β is chosen and $\mathcal{A}$ is given (K^*, k_β). $\mathcal{A}$ continues making key extraction queries subject to the previous restriction. At some point, $\mathcal{A}$ outputs a bit β'. The advantage of $\mathcal{A}$ is given by

$$\left| \mathsf{Prob}[\beta = \beta'] - \frac{1}{2} \right|.$$

We say that an IBKEM is (ϵ, t) CPA-secure in the selective-ID model if the maximum advantage of any adversary running in time at most t in the above game is ϵ.

In the full model, $\mathcal{A}$ does not have to commit to the challenge identity before protocol set-up. Instead, $\mathcal{A}$ can choose the challenge identity based on its key extraction queries and their responses. There is a natural restriction that $\mathcal{A}$ should not query the key extraction oracle on the challenge identity.

A.3 Authenticated Encryption

An AE protocol consists of two deterministic algorithms – $\mathsf{Encrypt}$ and $\mathsf{Decrypt}$. Both of these use a common secret key k. The $\mathsf{Encrypt}_k$ algorithm takes as input a nonce IV and a message M and returns (C, tag), where C is the ciphertext corresponding to M (and is usually of the same length as M). The $\mathsf{Decrypt}_k$ algorithm takes as input IV and a pair (C, tag) and returns either the message M or $\perp$ (indicating invalid ciphertext).

An AE algorithm possesses two security properties – privacy and authenticity. For privacy, the adversarial game is the following. The adversary $\mathcal{A}$ is given access to an oracle which is either the encryption oracle instantiated with a random key k or is an oracle which simply returns random strings of length equal to its input. After interacting with the oracle the adversary ultimately outputs a bit. The advantage of $\mathcal{A}$ is defined to be

$$|\mathsf{Prob}[\mathcal{A} = 1|\text{real oracle}] - \mathsf{Prob}[\mathcal{A} = 1|\text{random oracle}]|.$$

In the above game, the adversary is assumed to be nonce-respecting, in that it does not repeat a nonce. The requirement that IV is a nonce can be replaced the requirement that IV is chosen randomly. This leads to an additive quadratic degradation in the advantage.

The security notion defined above is that of pseudorandom permutation. This implies the notion of CPA-security. In particular, it implies the following notion of (one-time) CPA-security. The adversary submits two equal length messages M_0 and M_1. A random (IV^*, k^*) pair is chosen and a random bit β is chosen. The adversary is given (C^*, tag^*) which is the encryption of M_β using IV^* and k^*. The adversary then outputs β' and its advantage is

$$\left| \mathsf{Prob}[\beta = \beta'] - \frac{1}{2} \right|.$$

We say that an AE protocol satisfies (ϵ, t) one-time CPA-security if the maximum advantage of any adversary running in time t in the above game is ϵ.

The authenticity property of an AE protocol is defined through the following game. A nonce respecting adversary $\mathcal{A}$ is given access to an encryption oracle instantiated by a secret key k. It submits messages to the oracle and receives as output ciphertext-tag pairs. Finally, it outputs a "new" ciphertext-tag pair and a nonce, which can be equal to a previous nonce. The advantage of $\mathcal{A}$ in this game is the probability that the forgery is valid, i.e., it will be accepted as a valid ciphertext.

As before, we can replace the requirement that IV be a nonce by the requirement that IV is random without significant loss of security. By an (ϵ, t)-secure authentication of an AE protocol we mean that the maximum advantage of any adversary running in time t in the above game is ϵ.

Symmetric Encryption. We will also use a symmetric encryption scheme which consists of two algorithms – an $\mathsf{Encrypt}$ and a $\mathsf{Decrypt}$ algorithm both of which use a common secret key k. The $\mathsf{Encrypt}_k$ algorithm takes as input a random IV and a message M and returns as output a ciphertext C which is of the same length as M. The $\mathsf{Decrypt}_k$ algorithm takes as input a random IV and a ciphertext C and returns as output a message M. We require such a symmetric encryption scheme to satisfy the notion of (one-time) CPA-security described above.

A.4 Key Derivation Function

A KDF takes an input x and produces an output y. A KDF may be a keyed function, in which case the key will be public. The security requirement on KDF is the following. Given a random x as input, it should be computationally infeasible to distinguish the output of the KDF from a random string. By an (ϵ, t)-secure KDF we mean that the maximum advantage of an adversary running in time t in breaking the security of the KDF is at most ϵ.

B A Sketch of the Proof of Security for the Protocol in Figure 4

The proof is almost the same as the proof given in [2]. The changes are due to the way a message is authenticated. This is done using a MAC algorithm in [14, 2], whereas here we use an AE algorithm. We describe the idea behind the required modifications.

Let $(u_1^*, u_2^*, C^*, \mathrm{tag}^*)$ be the challenge ciphertext. Once the DDH problem and the KDF is taken care of we can assume k^* to be random. Also, using a "plug-and-pray" technique, let i be such that the i-th query is $(u_1^{(i)}, u_2^{(i)}, C^{(i)}, \mathrm{tag}^{(i)})$ with $\log_{g_1}(u_1) \neq \log_{g_2}(u_2)$. Then using simple linear algebra (and after taking care of KDF) it can be shown that $k^{(i)}$ is random. See [2] for details. There are two places where MAC is used in the game sequence in [2].

If the adversary ever submits a decryption query $(u_1, u_2, C, \mathrm{tag})$, with $(u_1, u_2) = (u_1^*, u_2^*)$ and the MAC algorithm verifies the message, the adversary has effectively broken MAC after seeing one digest using a random key k^*. In the context of AE, if the adversary can provide a valid (C, tag) for the key k^* after seeing only (C^*, tag^*) obtained using k^*, then the adversary has effectively broken the authorization aspect of the AE protocol.

In case of the i-th query $(u_1^{(i)}, u_2^{(i)}, C^{(i)}, \mathrm{tag}^{(i)})$, the key $k^{(i)}$ is random. Hence, in this case, the adversary is able to break the authentication aspect of the AE protocol by providing a forgery $(C^{(i)}, \mathrm{tag}^{(i)})$ on the unknown secret key $k^{(i)}$ without seeing any previous ciphertext-tag pair obtained using $k^{(i)}$. This is more difficult for the adversary and hence his advantage in doing this can be lower.

The other aspect is the symmetric encryption component. In [14, 2], it is assumed that the symmetric encryption is (one-time) CPA-secure. Since, we assume that the AE protocol is a pseudo-random permutation this implies the required (one-time) CPA-security.

The entire game sequence from [2] can be rewritten in a routine manner using the modifications described above. □