Efficient Hybrid Encryption from ID-Based Encryption

Masayuki Abe[1], Yang Cui[2], Hideki Imai[3], and Eike Kiltz[4]

[1] Information Sharing Platform Laboratories
NTT
1-1 Hikarino-oka, Yokosuka-shi, 239-0847 Japan
abe.masayuki@lab.ntt.co.jp
[2] University of Tokyo
cuiyang@imailab.iis.u-tokyo.ac.jp
[3] Chuo University & RCIS, AIST, Japan
h-imai@aist.or.jp
[4] CWI Amsterdam
The Netherlands
kiltz@cwi.nl
http://kiltz.net

Abstract. This paper deals with generic transformations from ID-based key encapsulation mechanisms (IBKEM) to hybrid public-key encryption (PKE). The best generic transformation known until now is by Boneh and Katz and requires roughly 704-bit overhead in the ciphertext. We present two new such generic transformations that are applicable to *partitioned* IBKEMs. A partitioned IBKEM is an IBKEM that provides some extra structure. Such IBKEMs are quite natural and in fact nearly all known IBKEMs have this additional property. Our first transformation yields chosen-ciphertext secure PKE schemes from selective-ID secure partitioned IBKEMs with a 256-bit overhead in ciphertext size plus one extra exponentiation in encryption/decryption. As the central tool a Chameleon Hash function is used to map the identities. The second transformation transforms adaptive-ID secure partitioned IBKEMs into chosen-ciphertext secure PKE schemes with no additional overhead.

Applying our transformations to existing IBKEMs we propose a number of novel PKE schemes with different trade-offs. In some concrete instantiations the Chameleon Hash can be made "implicit" which results in improved efficiency by eliminating the additional exponentiation.

Since our transformations preserve the public verifiability property of the IBE schemes it is possible to extend our results to build threshold hybrid PKE schemes. We show an analogue generic transformation in the threshold setting and present a concrete scheme which results in the most efficient threshold PKE scheme in the standard model.

Keywords. Hybrid Encryption, Selective-ID, KEM, Threshold PKE, Chameleon Hash

1 Introduction

1.1 Background

Research on efficient and secure public-key encryption (PKE) has been central in cryptography. A vast number of papers are devoted to construct cryptosystems that achieves security against adaptive chosen ciphertext attacks, aka CCA-security [34, 38]. Many of the scheme are only secure in the random oracle model [3]. The random oracle model is a *heuristic*, and a proof of security in the random oracle model does not directly imply anything about the security of a system in the real world. In fact, it has been demonstrated that there exist cryptographic schemes which are secure in the random oracle model but which are inherently insecure when the random oracle is instantiated with any real hash function (see, e.g., [13]).

In this paper we focus on CCA-secure cryptosystems under standard cryptographic assumptions, without using random oracles. Early such constructions were given in [34, 38, 22], which are considered as theoretical due to the use of general technique of non-interactive zero-knowledge proofs. The first practical breakthrough was introduced by Cramer and Shoup [19], followed by its generalization [20].

Identity-based encryption (IBE) [40] extends the framework of public-key encryption such that one's public-key consists of an arbitrarily chosen string, possibly a string associated to one's real identity, on top of a set of system parameters shared by all users. The first instantiation given by Sakai et al. [39] and Boneh and Franklin [9] opened vista for further research on IBE itself and its numerous applications. While most early constructions of CCA-secure PKE followed the Naor-Yung paradigm [34], a completely different approach based on IBE schemes was presented by Canetti et al. [16]. The technique, aka CHK transform, is a *generic transformation* that converts any selective-ID CPA-secure IBE scheme into a CCA-secure PKE scheme by employing a one-time signature. Due to the inefficiency of one-time signatures, the transformation adds about 65k bits[5] of overhead to the original ciphertext in a typical setting. Later, Boneh and Katz gave an alternative transformation (BK transformation) with improved efficiency by essentially using a MAC [11]. However, the BK transformation still requires 704 bits of overhead. Since typical IBE schemes have ciphertext sizes of about 512 bits this transformation still doubles the ciphertext overhead.

Meanwhile more efficient but dedicated constructions of CCA-secure key-encapsulation mechanisms (KEMs) were developed in [12, 27–29]. Whereas a PKE scheme encrypts messages, in a KEM random session keys are encapsulated. According to the KEM/DEM composition theorem [20], these KEMs in combination with CCA-secure symmetric encryption schemes eventually yield CCA-secure hybrid PKE schemes. These dedicated KEM constructions are based on "identity-based techniques" and exploit certain algebraic structures of known IBE schemes, mostly the one from Boneh and Boyen [6]. Compared to the PKE schemes obtained by the generic transformations these dedicated PKE schemes are more efficient in ciphertext length.

Constructing hybrid PKEs in the threshold setting is yet another interesting issue. There are some dedicated constructions in the standard model based on the Cramer-Shoup encryption, which need interaction between the decryption servers [14, 2]. The paper [8] extends the CHK transformation to the threshold setting and presents a *non-interactive* threshold CCA-secure PKE based on a threshold variant of the Boneh-Boyen IBE. Unfortunately the more efficient BK transformation cannot be applied here since due to the MAC consistency of the resulting scheme can only be privately verified. Accordingly, the threshold scheme in [8] suffers from a large overhead of about 65k caused by the CHK transform.

Some efficient CCA-secure threshold KEMs are presented in [12] and [24]. Unfortunately, the KEM/DEM composition theorem *does not* work in the threshold setting [1] and it is not known how to generically convert a threshold KEM into a threshold PKE.

1.2 Our Contributions

New generic transformations. We address the issue of reducing the ciphertext overhead introduced in generic transformations from IBE to PKE. As our main contribution we present two new constructions that transform an identity-based key encapsulation mechanism (IBKEM) into a hybrid public-key encryption scheme. Our constructions are applicable to so called *partitioned* IBKEMs which will be explained shortly.

Our new transformations have the following properties.

- The first transformation transforms selective-ID secure partitioned IBKEMs into chosen-ciphertext secure PKE schemes. The additional overhead in the transformation is one application of a Chameleon hash [30] (also called trapdoor hash function) in encryption and decryption, respectively. For a typical implementation of a chameleon hash this results in a ciphertext expansion of 128-bits plus one extra multi-exponentiation in encryption/decryption.[6]
- The second transformation transforms (a strong variant of) adaptive-ID secure partitioned IBKEMs into chosen-ciphertext secure PKE schemes with no additional overhead.

We remark that in contrast to the CHK and BK transformations, our construction adds another computational assumption to the security of the resulting PKE scheme. However, efficient Chameleon hashes

[5] There are various ways to build one-time signatures with different tradeoffs in size and computational cost. Our estimated 65k overhead comes from the a hash-tree based one-time signature scheme [23]. Using a one-time signature based on Pedersen's commitment [36] one can reach to a ciphertext overhead of roughly 1k-6k with two extra exponentiations.

[6] One important detail is that, in contrast to the CHK and BK transformations, we do not have to include the public parameters of the Chameleon hash in the PKE ciphertext.

can be built an reletively weak assumptions such as discrete logarithms or factoring [30] which are implied by the security assumptions on which all known IBE schemes are based on.

Our transformations yield tag-KEMs [1], not PKE schemes. This is more general since every tag-KEM can again be transformed into a hybrid PKE scheme by pairing it with a passively secure symmetric encryption scheme. The advantage of a tag-KEM (rather than PKE) is its great flexibility, i.e. it completely decouples the key encapsulation from the asymmetric part. Furthermore, in contrast to a standard KEM [20], a tag-KEM only requires a passively secure symmetric encryption scheme for the hybrid tag-KEM/DEM construction. We refer to [1] for more details.

Partitioned IBKEMs. A small price we pay in our transformations is a restriction on the ingredient, i.e. that the underlying IBKEM is required to be *partitioned*. Roughly, an IBKEM is *partitioned*, if (i) the encapsulated session key does not depend on the identity, and (ii) the ciphertext can be split into two parts such that one part depends on the identity but the other one does not. We will argue that partitioned IBKEMs are a natural extension of IBKEMs — all known IBKEMs without random oracles [6, 41, 25] are in fact partitioned. There are only two known ID-KEMs that are not partitioned [39, 10]. However, both schemes are only secure in the random-oracle model, while we prefer to the generic solution in the standard model.

We think that our notion of partitioned IBKEMs may be also of independent theoretical interest. As we will discuss later, it provides more insight in the problem of construction efficient PKE schemes in the standard model.

New PKE schemes. We demonstrate the usefulness of our new transformations by providing example instantiations of PKE schemes based on known IBE schemes from Boneh-Boyen [6], Waters [41], and Gentry [25].

- For the selective-ID secure Boneh-Boyen IBE scheme we use our first transformation to obtain an efficient PKE scheme. We further demonstrate how to improve efficiency of our transformation by using an *implicit Chameleon Hash*. The implied PKE scheme saves one multi-exponentiation in encryption/decryption.
- For Waters' adaptive-ID secure IBE scheme we use our second (loss-free) transformation. The resulting PKE scheme resembles exactly the PKE scheme presented in [12]. Using our generic transformation we are able to explain its security in terms of the original IBE scheme.
- For Gentry's adaptive-ID secure IBE scheme we again use our second transformation. The implied PKE is a new construction of a hybrid PKE scheme with interesting properties.

New Generic Transformations in the Threshold Setting. Since our transformations preserve the public verifiability property of the IBE schemes it is possible to extend our results to the threshold setting where some servers shares the private key and collaborate to decrypt a ciphertext. We provide the required transformations and give a concrete example instantiation of a threshold PKE scheme which is more efficient than recently proposed solutions in the standard model [8].

2 Preliminaries

This section introduces notations and definitions that are essential to this paper. Due to page limitation we may use some standard primitives such as public-key encryption (PKE) and deta encapsulation mechanism (DEM) without rigorous definitions but using intuitive notations. Formal definitions apper in the full version of this paper.

2.1 Notations

If x is a string, then $|x|$ denotes its length, while if S is a set then $|S|$ denotes its size. If $k \in \mathbb{N}$ then 1^k denotes the string of k ones. If S is a set then $s \overset{\$}{\leftarrow} S$ denotes the operation of picking an element s of S uniformly at random. We write $\mathcal{A}(x, y, \ldots)$ to indicate that $\mathcal{A}$ is an algorithm with inputs $x, y, \ldots$ and by $z \overset{\$}{\leftarrow} \mathcal{A}(x, y, \ldots)$ we denote the operation of running $\mathcal{A}$ with inputs $(x, y, \ldots)$ and letting z be the output. We write $\mathcal{A}^{\mathcal{O}_1, \mathcal{O}_2, \cdots}(x, y, \ldots)$ to indicate that $\mathcal{A}$ is an algorithm with inputs $x, y, \ldots$ and access to oracles $\mathcal{O}_1, \mathcal{O}_2, \ldots$ and by $z \overset{\$}{\leftarrow} \mathcal{A}^{\mathcal{O}_1, \mathcal{O}_2, \cdots}(x, y, \ldots)$ we denote the operation of running $\mathcal{A}$ with inputs $(x, y, \ldots)$ and access to oracles $\mathcal{O}_1, \mathcal{O}_2, \ldots$, and letting z be the output.

2.2 Tag-KEM

A *key-encapsulation mechanism with tags* (tag-KEM) [1] $\mathcal{TKEM}$ consists of four polynomial-time algorithms.

$(pk, sk) \xleftarrow{\$} \mathsf{TKEM.Kg}(1^k)$; A probabilistic key-generation algorithm that produces a master key pair for given security parameter $k \in \mathbb{N}$. The public-key pk defines a key space $KeySp$ and a tag space $TagSp$.

$(K, \omega) \xleftarrow{\$} \mathsf{TKEM.Key}(pk)$; A probabilistic algorithm that outputs a key K and some state information ω.

$C \xleftarrow{\$} \mathsf{TKEM.Cipher}(t, \omega)$; A probabilistic algorithm that creates a ciphertext C for tag t and key K which is implicitly transmitted by state information ω.

$K \xleftarrow{\$} \mathsf{TKEM.Decaps}(sk, t, C)$; A probabilistic decryption algorithm that decapsulates ciphertext C with tag t to recover a session key K. It may also output a special symbol $\perp$ to present rejection.

Note that in contrast to the original definition [1] we allow decapsulation also to be a probabilistic algorithm.

For consistency, we require that for all $k \in \mathbb{N}$, all tags $t \in TagSp$, and all $(K, \omega) \xleftarrow{\$} \mathsf{TKEM.Key}(pk)$ and $C \xleftarrow{\$} \mathsf{TKEM.Cipher}(t, \omega)$, it must hold that $\Pr[\mathsf{TKEM.Decaps}(sk, t, C) = K] = 1$, where the probability is taken over the above randomized algorithms.

The security we require for tag-KEMs is IND-CCA security [1]. It is captured by defining the following experiment.

```
Experiment $\mathbf{Exp}^{tkem\text{-cca}}_{\mathcal{TKEM},\mathcal{A}}(k)$
    $(pk, sk) \xleftarrow{\$} \mathsf{TKEM.Kg}(1^k)$ ; $(K_1^*, \omega) \xleftarrow{\$} \mathsf{TKEM.Key}(pk)$ ; $K_0^* \xleftarrow{\$} KeySp$ ; $b \xleftarrow{\$} \{0,1\}$
    $(t^*, St_1) \xleftarrow{\$} \mathcal{A}_1^{\mathsf{TKEM.Decaps}(sk,\cdot,\cdot)}(pk, K_b^*)$
    $C^* \xleftarrow{\$} \mathsf{TKEM.Cipher}(\omega, t^*)$
    $b' \xleftarrow{\$} \mathcal{A}_2^{\mathsf{TKEM.Decaps}(sk,\cdot,\cdot)}(C^*, St_1)$
    If $b \neq b'$ then return 0 else return 1.
```

The adversary is restricted not to ask (t^*, C^*) to the decryption oracle $\mathsf{TKEM.Decaps}$.

We define the advantage of $\mathcal{A}$ in the chosen-ciphertext experiment as

$$\mathbf{Adv}^{tkem\text{-cca}}_{\mathcal{TKEM},\mathcal{A}}(k) \; = \; |\Pr[\mathbf{Exp}^{tkem\text{-cca}}_{\mathcal{TKEM},\mathcal{A}}(k) = 1] - 1/2| \; .$$

A tag-KEM $\mathcal{TKEM}$ is said to be indistinguishable against chosen-ciphertext attacks (IND-CCA secure in short) if the advantage function $\mathbf{Adv}^{tkem\text{-cca}}_{\mathcal{TKEM},\mathcal{A}}(k)$ is a negligible function in k for all polynomial-time adversaries $\mathcal{A}$.

2.3 Tag-KEM/DEM: From Tag-KEM to Hybrid PKE

We transform a tag-KEM $\mathcal{TKEM} = (\mathsf{TKEM.Kg}, \mathsf{TKEM.Key}, \mathsf{TKEM.Cipher}, \mathsf{TKEM.Decaps})$ and a DEM $\mathcal{DEM} = (\mathsf{DEM.Enc}, \mathsf{DEM.Dec})$ into a public-key encryption scheme $\mathcal{PKE} = (\mathsf{PKE.kg} = \mathsf{TKEM.Kg}, \mathsf{PKE.Enc}, \mathsf{PKE.Dec})$ as follows.

```
$\mathsf{PKE.Enc}(pk, M)$
    $(K, \omega) \xleftarrow{\$} \mathsf{TKEM.Key}(pk)$
    $\psi \leftarrow \mathsf{DEM.Enc}(K, M)$
    $C \xleftarrow{\$} \mathsf{TKEM.Cipher}(\psi, \omega)$
    Return $C||\psi$
```

```
$\mathsf{PKE.Dec}(sk, C||\psi)$
    $K \xleftarrow{\$} \mathsf{TKEM.Decaps}(sk, \psi)$
    $M \leftarrow \mathsf{DEM.Dec}(K, \psi)$
    Return $M$
```

The following was proven in [1].

Theorem 1. *Suppose $\mathcal{TKEM}$ is IND-CCA secure and $\mathcal{DEM}$ is IND-OT secure. Then the PKE scheme is IND-CCA secure.*

2.4 Chameleon Hash

A Chameleon Hash $\mathcal{CMH} = (\mathsf{CMH.Kg}, \mathsf{CMH.H}, \mathsf{CMH.Trap})$ is specified by the following algorithms [30].

$(pk_{ch}, sk_{ch}) \overset{\$}{\leftarrow} \mathsf{CMH.Kg}(1^k)$**;** A key-generation algorithm $\mathsf{CMH.Kg}$ that outputs a pair of hash and trapdoor keys pk_{ch}, sk_{ch}, respectively. The hash key defines the message space $\mathsf{Msg_{CMH}}$ and randomness space $\mathsf{Rand_{CMH}}$.

$h \leftarrow \mathsf{CMH.H}(pk_{ch}, m, s)$**;** A hash algorithm that takes the hash key pk_{ch}, message $m \in \mathsf{Msg_{CMH}}$, and randomness $s \in \mathsf{Rand_{CMH}}$ and outputs a hash value h.

$s \leftarrow \mathsf{CMH.Trap}(sk_{ch}, m', s', m)$**;** An algorithm that computes randomness s such that $\mathsf{CMH.H}(pk_{ch}, m', s') = \mathsf{CMH.H}(pk_{ch}, m, s)$ for given $m', m \in \mathsf{Msg_{CMH}}$ and $s' \in \mathsf{Rand_{CMH}}$ by using trapdoor-key sk_{ch}.

In the original security definition of a Chameleon hash, full collision resistance [21] was considered. However, in our case something weaker is sufficient, i.e. random prefix collision resistance [1], which seems close to the target collision resistance [33, 4] rather than full collision resistance. It is captured by defining the following experiment.

```
Experiment $\mathbf{Exp}^{cmh\text{-}\mathrm{rpc}}_{\mathcal{CMH},\mathcal{A}}(k)$
    $(pk_{ch}, sk_{ch}) \overset{\$}{\leftarrow} \mathsf{CMH.Kg}(1^k)$
    $(m, St) \overset{\$}{\leftarrow} \mathcal{A}_1(pk_{ch})$
    $s \overset{\$}{\leftarrow} \mathsf{Rand_{CMH}}$
    $(m', s') \overset{\$}{\leftarrow} \mathcal{A}_1(s, St)$
    If $\mathsf{CMH.H}(pk_{ch}, m', s') = \mathsf{CMH.H}(pk_{ch}, m, s)$ then return 1 else return 0.
```

We define the success probability of $\mathcal{A}$ in the above experiment as

$$\mathbf{Adv}^{cmh\text{-}\mathrm{rpc}}_{\mathcal{CMH},\mathcal{A}}(k) \;=\; \Pr[\mathbf{Exp}^{cmh\text{-}\mathrm{rpc}}_{\mathcal{CMH},\mathcal{A}}(k) = 1]\;.$$

A chameleon hash $\mathcal{CMH}$ is said to be random prefix collision resistant (RPC secure, in short) if $\mathbf{Adv}^{cmh\text{-}\mathrm{rpc}}_{\mathcal{CMH},\mathcal{A}}(k)$ is a negligible function in k for all polynomial-time adversaries $\mathcal{A}$.

3 Partitioned Identity-Based Key Encapsulation Mechanisms

3.1 Syntax

An *identity-based key-encapsulation mechanism* (IBKEM) scheme [40, 10] $\mathcal{IBKEM} = (\mathsf{IBKEM.Kg}, \mathsf{IBKEM.Extract}, \mathsf{IBKEM.Key}, \mathsf{IBKEM.Encaps}, \mathsf{IBKEM.Decaps})$ consists of four polynomial-time algorithms.

$(pk, sk) \overset{\$}{\leftarrow} \mathsf{IBKEM.Kg}(1^k)$**;** A probabilistic key-generation algorithm that produces a master key pair for given security parameter $k \in \mathbb{N}$. The public-key pk defines a key space $KeySp$ and an identity space $IDSp$.

$usk[id] \overset{\$}{\leftarrow} \mathsf{IBKEM.Extract}(sk, id)$**;** A probabilistic algorithm that outputs a user secret key $usk[id]$ for identity id.

$(C, K) \overset{\$}{\leftarrow} \mathsf{IBKEM.Cipher}(pk, id)$**;** A probabilistic algorithm that creates a key K and a ciphertext C for identity id.

$K \leftarrow \mathsf{IBKEM.Decaps}(usk[id], id, C)$**;** A deterministic decryption algorithm that decapsulates ciphertext C with identity id to recover a session key K. It may also outputs a special symbol $\perp$ to present rejection.

For consistency, we require that for all $k \in \mathbb{N}$, all identities id, and all $(C, K) \overset{\$}{\leftarrow} \mathsf{IBKEM.Encaps}(pk, id)$, we have $\Pr[\mathsf{IBKEM.Decaps}(\mathsf{IBKEM.Extract}(sk, id), id, C) = K] = 1$, where the probability is taken over the choice of $(pk, sk) \overset{\$}{\leftarrow} \mathsf{IBKEM.Kg}(1^k)$, and the coins of all the algorithms in the expression above.

For fixed public key pk and identity id we define the set of valid ciphertexts $CipherSp = CipherSp(pk, id)$ as all possible C that can be output from the probabilistic $(C, K) \overset{\$}{\leftarrow} \mathsf{IBKEM.Cipher}(pk, id)$.

Definition 2. *An IBKEM is said to be* partitioned *if (for all possible public/secret keys and identities) it satisfies the following three properties:*

- *(Independence property of the key.) The encapsulated key K does not depend on id.*
- *(Unique split property of the ciphertext.) The ciphertext C can be split into two parts $C = (c_1, c_2)$ such that the first part c_1 does not depend on id. Furthermore, the first part c_1 and identity id together uniquely determine the second part c_2.*
- *(Perfect \$-rejection property.) For every invalid $C = (\tilde{c}_1, \tilde{c}_2) \notin CipherSp(pk, id)$ we require that* $\mathsf{IBKEM.Decaps}(\mathsf{IBKEM.Extract}(sk, id), id, (\tilde{c}_1, \tilde{c}_2))$ *outputs a random $K \in KeySp$. (Since* $\mathsf{IBKEM.Decaps}$ *is deterministic, the randomness for K would come from* $\mathsf{IBKEM.Extract}(sk, id)$*.)*

Note that for the second condition it is important to require c_2 to be uniquely defined on c_1 and id, since otherwise a trivial splitting of the ciphertext as $(c_1, c_2) = (\varepsilon, c_1||c_2)$ is always possible, where ε denotes the empty string.

Though such strong properties are indeed provided by all existing schemes in the standard model, we will show that some requirements can be relaxed or generalized in the sequel.

The properties of a partitioned IBKEM imply that algorithm $\mathsf{IBKEM.Encaps}$ can be split into three parts, say $\mathsf{IBKEM.Key}$, $\mathsf{IBKEM.Cipher1}$, and $\mathsf{IBKEM.Cipher2}$ that create K, c_1, and c_2, respectively. Formally, we define these algorithms as a set of functions that shares the state information φ.

$(K, \varphi) \xleftarrow{\$} \mathsf{IBKEM.Key}(pk)$**;** A probabilistic algorithm that takes a master public-key, and outputs a session key K and state information φ.

$c_1 \leftarrow \mathsf{IBKEM.Cipher1}(\varphi)$**;** A deterministic algorithm that takes a state information φ generated by $\mathsf{IBKEM.Key}$, and outputs the first part of ciphertext c_1.

$c_2 \leftarrow \mathsf{IBKEM.Cipher2}(\varphi, id)$**;** A deterministic algorithm that takes a state information φ and an identity id, and computes the second part of ciphertext c_2.

The ciphertext is (c_1, c_2). The state information φ generated by $\mathsf{IBKEM.Key}$ would include the randomness used to generate K and the public key pk. Executing these three functions in sequence with consistent state information φ must yield the same output as $\mathsf{IBKEM.Encaps}(pk, id)$.

At this point it is worth noting that, to our best knowledge, all known IBKEMs [6, 41, 25] are in fact partitioned. We will present concrete instantiations of the resulting tag-KEMs in Section 6. The only known IBKEMs that are not partitioned are the Sakai et al. scheme [39] and the Boneh-Franklin scheme [10]. However, both schemes are out of our main purpose in this paper, since they are only secure in the random-oracle model.

3.2 Security against Selective-ID IND-CPA attacks.

The notion of indistinguishability against selective-identity and chosen-plaintext attacks, sID IND-CPA in short, is defined in the same way as given in [10] regardless of the partitioned structure. Formally, it is defined through the following game between a challenger and an adversary $\mathcal{A} = (\mathcal{A}_0, \mathcal{A}_1)$. Let $\mathcal{IBKEM} = (\mathsf{IBKEM.Kg}, \mathsf{IBKEM.Extract}, \mathsf{IBKEM.Key}, \mathsf{IBKEM.Cipher1}, \mathsf{IBKEM.Cipher2}, \mathsf{IBKEM.Decaps})$ be a partitioned IBKEM.

To an adversary $\mathcal{A}$ we associate the following experiment:

```
Experiment $\mathbf{Exp}^{ibkem\text{-sid-cpa}}_{\mathcal{IBKEM},\mathcal{A}}(k)$
  $(id^*, St_0) \xleftarrow{\$} \mathcal{A}_0(1^k)$
  $(pk, sk) \xleftarrow{\$} \mathsf{IBKEM.Kg}(1^k)$ ; $(K_1^*, \varphi) \xleftarrow{\$} \mathsf{IBKEM.Key}(pk)$ ; $K_0^* \xleftarrow{\$} KeySp$ ; $b \xleftarrow{\$} \{0,1\}$
  $c_1^* \leftarrow \mathsf{IBKEM.Cipher1}(\varphi)$ ; $c_2^* \leftarrow \mathsf{IBKEM.Cipher2}(\varphi, id^*)$
  $b' \xleftarrow{\$} \mathcal{A}_1^{\mathsf{IBKEM.Extract}(sk,\cdot)}(pk, K_b^*, c_1^*, c_2^*, St_0)$
  If $b \neq b'$ then return 0 else return 1
```

Adversary $\mathcal{A}$ is not allowed to query oracle $\mathsf{IBKEM.Extract}(sk, \cdot)$ for the target identity id^*.

It is stressed that though the above experiment is described by using the separated encryption functions of an 'partitioned' IBKEM, from the viewpoint of the adversary it is exactly the same experiment as in the standard selective-ID IBKEM security definition presented in [10]. Hence an adversary having some advantage in attacking the standard IBKEM security has exactly the same advantage in attacking the partitioned IBKEM security described in the above experiment.

We define the advantage of $\mathcal{A}$ in the experiment as

$$\mathbf{Adv}^{ibkem\text{-sid-cpa}}_{\mathcal{IBKEM},\mathcal{A}}(k) \ = \ |\Pr[\mathbf{Exp}^{ibkem\text{-sid-cpa}}_{\mathcal{IBKEM},\mathcal{A}}(k) = 1] - 1/2| \ .$$

A partitioned IBKEM $\mathcal{IBKEM}$ is said to be selective-identity IND-CPA if the advantage functions $\mathbf{Adv}^{ibkem\text{-sid-cpa}}_{\mathcal{IBKEM},\mathcal{A}}(k)$ is a negligible function in k for all polynomial-time adversaries $\mathcal{A}$.

4 Transformation using Chameleon Hash

We construct a tag-KEM $\mathcal{TKEM} = (\mathsf{TKEM.Kg}, \mathsf{TKEM.Key}, \mathsf{TKEM.Cipher}, \mathsf{TKEM.Decaps})$ from a partitioned selective-ID IBKEM scheme $\mathcal{IBKEM}$ and a chameleon hash scheme $\mathcal{CMH}$. For ease of notation, $\mathcal{CMH}$ is considered as global so that $\mathsf{CMH.H}$ is accessible from a function without explicitly taking its key from outside.

First of all, we set the key generation algorithm $\mathsf{TKEM.Kg} = \mathsf{IBKEM.Kg}$.[7] The rest of the algorithms are constructed as follows.

```
TKEM.Key($pk$)
    $(K, \varphi) \xleftarrow{\$} \mathsf{IBKEM.Key}(pk)$
    Return $(K, \varphi)$

TKEM.Cipher($pk, t, \varphi$)
    $c_1 \leftarrow \mathsf{IBKEM.Cipher1}(\varphi)$
    $s \xleftarrow{\$} \mathsf{Rand}_{\mathsf{CMH}}$ ; $id \leftarrow \mathsf{CMH.H}(t||c_1, s)$
    $c_2 \leftarrow \mathsf{IBKEM.Cipher2}(\varphi, id)$
    Return $c \leftarrow c_1||c_2||s$

TKEM.Decaps($sk, t, c$)
    $c_1||c_2||s \leftarrow c$
    $id \leftarrow \mathsf{CMH.H}(t||c_1, s)$
    $sk[id] \xleftarrow{\$} \mathsf{IBKEM.Extract}(sk, id)$
    $K \leftarrow \mathsf{IBKEM.Decaps}(sk[id], id, (c_1, c_2))$
    Return $K$
```

In this transformation we implicitly require that the Chameleon hash maps elements from correct domains. In a more general setting we can apply (target) collision resistant hash functions, see Section 6 for more details.

Theorem 3. *If the partitioned IBKEM is selective-ID IND-CPA secure and the chameleon hash is random prefix collision resistant, then the above TKEM is IND-CCA secure. In particular,*

$$\mathbf{Adv}^{tkem\text{-cca}}_{\mathcal{TKEM},\mathcal{A}}(k) \le \mathbf{Adv}^{ibkem\text{-sid-cpa}}_{\mathcal{IBKEM},\mathcal{B}}(k) + \mathbf{Adv}^{cmh\text{-rpc}}_{\mathcal{CMH},\mathcal{C}}(k).$$

Proof. (Sketch.) Assume there exists an adversary $\mathcal{A}$ against the IND-CCA security of the tag-KEM $\mathcal{TKEM}$. We show that then there exists either an adversary $\mathcal{B}$ against the selective-ID IND-CPA security of $\mathcal{IBKEM}$ or an adversary $\mathcal{C}$ against the random prefix collision resistance of $\mathcal{CMH}$.

We first describe adversary $\mathcal{B}$.

Setup. Given security parameter k, adversary $\mathcal{B}$ sets up a random instance (pk_{ch}, sk_{ch}) of the chameleon hash via $\mathsf{CMH.Kg}(1^k)$. It then generates the target identity id^* by $id^* \leftarrow \mathsf{CMH.H}(t', s')$ where t' and s' are chosen randomly from appropriate domains. It sends id^* to the challenger and receives a public-key pk and a challenge (K_b^*, c_1^*, c_2^*). Adversary $\mathcal{B}$ now runs $\mathcal{A}$ by giving pk and pk_{ch}.

Challenge Simulation. At some point, $\mathcal{A}$ outputs a target tag t^*. $\mathcal{B}$ then computes $s^* \leftarrow \mathsf{CMH.Trap}(sk_{ch}, t', s', t^*||c_1^*)$ and sends K_b^* and (c_1^*, c_2^*, s^*) to $\mathcal{A}$.

Decryption Oracle Simulation. If $\mathcal{A}$ makes a decryption query with a ciphertext (c_1, c_2, s) and a tag t, adversary $\mathcal{B}$ simulates decryption oracle $\mathsf{TKEM.Decaps}(sk, t, (c_1, c_2, s))$ in the following way. Let $id = \mathsf{CMH.H}(t||c_1, s)$.

Case 0: If $(c_1, c_2, s, t) = (c_1^*, c_2^*, s^*, t^*)$, return nothing.

Case 1: If $(c_1, s, t) \neq (c_1^*, s^*, t^*)$ and $id = id^*$, abort. (Denote this event by COL.)

[7] When $\mathcal{CMH}$ is local, $\mathsf{TKEM.Kg}$ also invokes $\mathsf{CMH.Kg}$ and include the hash keys to its public-key.

Case 2: If $(c_1, s, t) = (c_1^*, s^*, t^*)$ and $c_2 \neq c_2^*$, return a random K.
Case 3: If none of the above happens, send id to oracle $\mathsf{IBKEM.Extract}$ and receive $usk[id]$. Then compute $K \leftarrow \mathsf{IBKEM.Decaps}(usk[id], id, (c_1, c_2))$ and return K to $\mathcal{A}$.

Output. Finally, $\mathcal{A}$ returns a guess bit b'. Adversary $\mathcal{B}$ returns the same bit b' and terminates the game.

Challenge simulation is perfect since $id^* = \mathsf{CMH.H}(t', s') = \mathsf{CMH.H}(t^*, s^*)$. In the simulation of the decryption oracle, Case 0 and 3 are just as defined. In Case 2, the ciphertext (c_1^*, c_2) is clearly incorrect since $c_2^* \neq c_2$ is the only correct second part for (c_1^*, s^*, t^*). Consequently, due to the perfect \$-rejection property, the decryption oracle in the original experiment outputs a random K, just as done by $\mathcal{B}$. Accordingly, $\mathcal{B}$ perfectly simulates $\mathcal{A}$'s view in the IND-CCA experiment unless event COL happens.

If COL does not happen then $\mathcal{B}$ has the same advantage in winning the experiment as $\mathcal{A}$. On the other hand, we can build an adversary $\mathcal{C}$ against the random prefix collision resistance of the chameleon hash that wins with probability one if COL happens during $\mathcal{A}$'s simulation. This concludes the proof. □

Remark 4. Following Thoerem 1, one can combine the aobve tag-KEM and DEM to obtain an IND-CCA secure hybrid PKE.

Remark 5. There may exist some particular IBKEMs where the ciphertext C can not even be split into the two parts c_1 and c_2. One possible reason is that all elements in C depend on id. This is in particular the case when C only consists of one element. In that case one can add one "dummy element" c_1 to the ciphertext such that: (i) the security of the IBKEM is not influenced; (ii) the new split (c_1, c_2) now fulfills the "unique split property" as defined in Definition 2. The IBKEM from Sakai et al. [39] is such an example where this dummy element can easily be added without harming the security of the scheme, but only secure in the random oracle model.

From a theoretical point of view we find the need for element c_1 in the transformation quite interesting. Since a fixed id uniquely links c_1 with c_2 we observe that c_1 it can be viewed as a "witness" to prove correctness of the resulting tag-KEM ciphertext. Without witness c_1 such correctness cannot be verified. This may enables an adversary to modify c_2 in a certain way to mount a CCA attack.

If the element c_2 does not uniquely depend on c_1 and id, we instead require a limited sense of non-malleability. Namely, given correct (c_1, c_2, id), it is possible to create another correct (c_1, c_2', id) only with negligible probability even with access to the oracle $\mathsf{IBKEM.Extract}$. Such a property, however, would result in needing a dedicated proof, which is not preferable for generic construction we concern.

Remark 6. In the proof of Theorem 3, adversary $\mathcal{B}$ must simulate the behavior of the decryption oracle $\mathsf{IBKEM.Decaps}(\mathsf{IBKEM.Extract}(sk, id^*), id^*, (c_1^*, c_2))$ without knowing sk. The unique split and perfect \$-rejection properties from Definition 2 allow to do so.

Such a simulation is also possible if $\mathsf{IBKEM.Decaps}(\mathsf{IBKEM.Extract}(sk, id^*), id^*, (c_1^*, c_2))$ returns a special symbol, say $\perp$, for every incorrect c_2. In such a case, $\mathcal{B}$ simply outputs $\perp$ instead of random K. Hence such a property, which we call $\perp$-rejection property, is also acceptable instead of \$-rejection. Indeed, any kind of rejection is acceptable as long as the output distribution of $\mathsf{IBKEM.Decaps}(\mathsf{IBKEM.Extract}(sk, id^*), id^*, (c_1^*, c_2))$ is efficiently simulatable without knowing sk. One can also relax the "perfect" part by introducing a small error probability that additively affects to the reduction cost shown in Theorem 3 and 7.

5 Removing the Chameleon Hash

The heart of the transformation in Section 4 is the use of a chameleon hashing. Somewhat contradictory, this section shows several ways to eliminate the use of the chameleon hashing. The resulting transformations are of theoretical interest, or even have practical advantages.

5.1 Making a CMH-like functionality from CR

With closer look, one may notice that the scheme and the security proof in Section 4 did not use the full power of the CMH because the simulator only needs to show one preimage of the prefixed hash value to the adversary. The minimum property we really need is to compute s^* that maps given $t^*||c_1^*$ into id^* selected in advance. Theoretically, such a function can be made from one-way function plus a collision resistant hash function; First commit to the input string by applying Naor's bit commitment scheme [32] and then compress the commitment with the collision resistant hash function. The resulting function is collision resistant but not CMH in that one must choose an irregular hash key (but indistinguishable from correct ones) to make a commitment to be opened to an arbitrary value. But it is sufficient for our purpose since we need the property only in the security proof. Nonetheless, this is only of theoretical interest and out of the main scope of this paper since the resulting scheme suffers a huge overhead both in the public-key and ciphertext size.

5.2 Directly Replacing CMH with RPC

As mentioned above, the trapdoor information of CMH is used only in the security proof. The scheme therefore functions correctly even if chameleon hash function CMH is replaced by a collision resistant hash function CR. More precisely, we can use a random prefix collision resistant hash function RPC, which has more relaxed collision resistant property than that of CR as defined in Section 2.4, and replace $\mathsf{CMH.H}(t||c_1, s)$ in $\mathsf{TKEM.Cipher}$ and $\mathsf{TKEM.Decaps}$ by $\mathsf{RPC}(t||c_1||s)$.

With this modification the current security proof no longer works since the reduction algorithm needs to compute s^* that fulfills $id^* = \mathsf{RPC}(t^*||c_1^*||s^*)$ for id^*, t^*, and c_1^* selected in advance. We circumvent this technical difficulty by allowing the reduction algorithm to access to an oracle, which we call *hash partial preimage oracle*, that outputs such s^* for input (id^*, t^*, c_1^*) whenever it exists. For this idea to work, it is required that the underlying selective-ID IBKEM should remain IND-CPA even against the adversaries equipped with the partial preimage oracle. It would be reasonable when the underlying IBKEM and RPC can be considered as independent. Such a relaxation, which allows the adversary to access to a seemingly irrelevant oracle, has been used in the literature, e.g. [37, 31].

For example, let us take the concrete scheme shown in Section 6.1. The scheme uses the Boneh-Boyen's IBE in the above mentioned modified transformation. The Boneh-Boyen's IBE is selective-ID IND-CPA under the BDDH assumption. To cope with the hash partial preimage oracle, we need to strengthen the underlying assumption to the BDDH with the hash partial preimage oracle. That is, the necessary assumption for the resulting tag-KEM to be secure is that the BDDH assumption holds even for the adversary given access to the hash partial preimage oracle. Assuming that SHA-256 or AES-based hash functions is RPC would be reasonable since they are seemingly irrelevant to the number-theoretic structure of the BDDH problem. On the other hand, using RPC based on the discrete-logarithm assumption such as Pedersen hash would not be acceptable especially when the group is shared with the BDDH instance. Full description of the transformation with RPC and its security proof is given in Appendix B.

The modified scheme improves the efficiency. All known efficient construction of CMH limit the input length and need CR to compress the input string of arbitrary length. Needing single RPC in the modified scheme thus saves the computation for CMH. Also, the public-key has no additional elements than that of the underlying IBKEM. Note however that the length of the cipehrtext gets slightly longer because the randomness s would have to be chosen from larger domain as shown in the detailed security argument in Appendix B.

5.3 Based on Strongly Adaptive-ID IBKEM

In Section 3.2 we introduced the security notion of selective-ID IND-CPA for IBKEMs. For the stronger notion of security against "full-identity" IND-CPA attacks (IND-CPA security) changes the security experiment from Section 3.2 as follows. Instead of committing to the target identity id^* before seeing the public key the adversary now can adaptively select it. Namely, $(pk, sk) \overset{\$}{\leftarrow} \mathsf{IBKEM.Kg}(1^k)$ is done at the beginning of the experiment and pk is given to $\mathcal{A}_0$. No changes in anywhere else. We however do not use this notion in this paper and introduce rather stronger one as below to obtain a more efficient transformation.

Roughly, in this stronger case, the adversary is given session-key K_b^* and the first part of ciphertext c_1^* *before* selecting the target identity id^*. Note that K_b^* and c_1^* are independent of id^* in partitioned IBKEMs. Formally, the experiment is as follows.

```
**Experiment** $\mathbf{Exp}^{ibkem\text{-id-cpa}^*}_{\mathcal{IBKEM},\mathcal{A}}(k)$
    $(pk, sk) \xleftarrow{\$} \mathsf{IBKEM.Kg}(1^k)$ ; $(K_1^*, \varphi) \xleftarrow{\$} \mathsf{IBKEM.Key}(pk)$ ; $K_0^* \xleftarrow{\$} KeySp$ ; $b \xleftarrow{\$} \{0,1\}$
    $c_1^* \leftarrow \mathsf{IBKEM.Cipher1}(pk, \varphi)$
    $(id^*, St_1) \xleftarrow{\$} \mathcal{A}_1^{\mathsf{IBKEM.Extract}(sk,\cdot)}(pk, K_b^*, c_1^*)$
    $c_2^* \leftarrow \mathsf{IBKEM.Cipher2}(pk, \varphi, id^*)$
    $b' \xleftarrow{\$} \mathcal{A}_2^{\mathsf{IBKEM.Extract}(sk,\cdot)}(c_2^*, St_1)$
    If $b \neq b'$ then return 0 else return 1
```

Adversary $\mathcal{A}$ is not allowed to query oracle $\mathsf{IBKEM.Extract}(sk, \cdot)$ for the target identity id^*.

We define the advantage of $\mathcal{A}$ in the experiment as

$$\mathbf{Adv}^{ibkem\text{-id-cpa}^*}_{\mathcal{IBKEM},\mathcal{A}}(k) \;=\; |\Pr[\mathbf{Exp}^{ibkem\text{-id-cpa}^*}_{\mathcal{IBKEM},\mathcal{A}}(k) = 1] - 1/2| \;.$$

A partitioned IBKEM $\mathcal{IBKEM}$ is said to be strongly adaptive-ID IND-CPA (strongly IND-CPA) if the advantage functions $\mathbf{Adv}^{ibkem\text{-id-cpa}^*}_{\mathcal{IBKEM},\mathcal{A}}(k)$ is a negligible function in k for all polynomial-time adversaries $\mathcal{A}$.

The fact that the adversary can adaptively select the target identity id^* depending on K_b^* and c_1^* is crucial to our definition and also the reason why we call it *strong* adaptive-ID IND-CPA security. In fact, there may exist partitioned IBKEMs that are adaptive-ID IND-CPA in the standard sense of [10] (where id^* has to be provided before receiving challenge ciphertext/key) but not strongly adaptive-ID IND-CPA because of this information given in advance. However, these separating examples will hardly appear in practise since c_1 and K are both required to be independent of id. For concrete schemes (such as [41, 25]) it is an easy exercise to verify that the schemes actually provide this stronger type of security.

We remark that this difficulty does not arise in the case of selective-ID IND-CPA security since there the adversary has to commit to the target identity before even seeing the public key.

When a partitioned strongly adaptive-ID IBKEM is available, we can construct a tag-KEM without any overhead. Indeed, the construction is obtained simply by removing the chameleon hash from the construction in Section 4. For completeness, we show the construction below.

Set the key generation algorithm $\mathsf{TKEM.Kg} = \mathsf{IBKEM.Kg}$. The rest of the functions are constructed as follows.

```
$\mathsf{TKEM.Key}(pk)$
    $(K, \varphi) \xleftarrow{\$} \mathsf{IBKEM.Key}(pk)$
    Return $(K, \varphi)$

$\mathsf{TKEM.Cipher}(pk, t, \varphi)$
    $c_1 \leftarrow \mathsf{IBKEM.Cipher1}(\varphi)$
    $id \leftarrow t||c_1$
    $c_2 \leftarrow \mathsf{IBKEM.Cipher2}(\varphi, id)$
    Return $c \leftarrow c_1||c_2$

$\mathsf{TKEM.Decaps}(sk, t, c)$
    $c_1||c_2 \leftarrow c$
    $id \leftarrow t||c_1$
    $sk[id] \xleftarrow{\$} \mathsf{IBKEM.Extract}(sk, id)$
    $K \leftarrow \mathsf{IBKEM.Decaps}(sk[id], id, (c_1, c_2))$
    Return $K$
```

Theorem 7. *If the partitioned IBKEM is strongly adaptive-ID IND-CPA secure, then the above TKEM is IND-CCA secure. In particular,*

$$\mathbf{Adv}^{tkem\text{-cca}}_{\mathcal{TKEM},\mathcal{A}}(k) \leq \mathbf{Adv}^{ibkem\text{-id-cpa}^*}_{\mathcal{IBKEM},\mathcal{B}}(k).$$

The proof can be done by showing the construction of $\mathcal{B}$ from $\mathcal{A}$. Unlike the selective-ID case, $\mathcal{B}$ commits to the target identity after having c_1^* from the challenger and t^* from $\mathcal{A}$. Hence $\mathcal{B}$ can form $id^* = t^*||c_1^*$ without any problem. The rest of the simulation is the same as that for the proof of Theorem 3 with trivial modifications for removing the chameleon hash.

5.4 Other approaches

Similar to [12, 27] one could try to directly build a key-encapsulation mechanism [20] (KEM) out of a sID partioned IB-KEM using a target-collission resistant hash function TCR as follows. To encapsulate, first compute key and the first part of the ciphertext as $(K, \varphi) \overset{\$}{\leftarrow} \mathsf{IBKEM.Key}(pk)$ and $c_1 \leftarrow \mathsf{IBKEM.Cipher1}(\varphi)$. Then, the second part of the ciphertext is computed as $c_2 \leftarrow \mathsf{IBKEM.Cipher2}(\varphi, id)$, where $id = \mathsf{TCR}(c_1)$ is used to tie the two ciphertexts together. Whereas syntactically this is correct, without assuming further algebraic structure it seems hard to relate security of the KEM to the security of the IBKEM. This is since a simualator for the KEM security experiment interacting with the sID IBKEM challenger has to commit to a target identity before seeing the public key. But in the scheme the target identity depends on the first part of the target ciphertext and hence a stronger (and less natural) security requirement to the IBKEM scheme is needed for a general security reduction. In general, proving that the IBKEM satisfies such a stronger security property seems not easier than providing a direct proof for the transformed KEM.

5.5 Efficieny comparison

We make an efficiency comparison (with a security parameter of 128 bits) of the IBE to PKE transformations from [16, 11] with ours. The intries in Table 1 refer to the following transformations.

CHK+Pedersen: The CHK transformation instantiated with a one-time signature based on Pedersen's commitment. The one-time signature scheme works as follows. The public key consists of two group elements $u = g^a, v = g^b$, the two exponents a, b are the signing key. To sign a message m the signer computes $\sigma = (b - m)/a \in \mathbb{Z}_{|\mathbb{G}|}$ and the verifier checks if $g^m u^\sigma = v$. The overall ciphertext expansion introduced by the CHK transformation consists of the public key (two group elements) plus the signature. Depending on the instantiation of the group this is between 768 (elliptic curve groups) and $6k$ bits (prime-order subgroups of $\mathbb{Z}_q$). This transformation is secure under the DLOG assumption in the group.

CHK+hash-tree: The CHK transformation instantiated with a hash-tree based one-time signature, as recommended in [16]. The latter signature scheme has the advantage of having negliglible computational cost but has a ciphertext overhead of roughly 65k ($\approx 256^2$). This construction is secure under the assumption that the used hash functions are one-way and collission-resistant.

BK: The BK transformation instantiated with a computationally secure MAC (i.e., CBC-MAC) and a UOWHF-based commitment scheme [11].

Ours+DLOG: Our transformation from Section 4 instantiated with a DLOG-based Chameleon hash. The ciphertext overhead consists of one element in $\mathbb{Z}_{|\mathbb{G}|}$ which can be represented in 256 bits (independent of the group).

Ours+RPC: Our transformation from Section 5.2 based on a RPC secure hash function.

Ours+strong IBKEM: Our transformation from Section 5.3 applied to a strongly secure IBKEM.

Transformation	Input	Overhead			Additional	Publicly
		Ciphertext	Enc	Dec	assumption	verifiable?
CHK+hash-tree	sID IBE + one-time sig	65k	—	—	CR + OW	√
CHK+Pedersen	sID IBE + one-time sig	768-6k	2 exp	2 exp	DLOG	√
BK	sID IBE + MAC/COM	704	—	—	MAC + UOWHF	—
Ours+DLOG §4	sID part. IBKEM + CMH	256	1 exp	—	DLOG	√
Ours §5.2	sID part. IBKEM + RPC hash	128	1 exp	—	RPC	√
Ours §5.3	fID part. IBKEM	—	—	—	strong IBKEM	√

Table 1. Efficiency of the transformations for 128 bits security.

6 Concrete Tag-KEMs

In this section we demonstrate the usefulness of our transformation by giving possible instantiations of tag-KEMs based on known IBE schemes. Note that all our tag-KEMs imply IND-CCA secure PKE schemes using the generic transformation from Section 2.3.

Bilinear Groups. All pairing based schemes will be parameterized by a *pairing parameter generator.* This is a polynomial-time algorithm $\mathcal{G}$ that on input 1^k returns the description of an multiplicative cyclic group $\mathbb{G}$ of prime order p, where $2^k < p < 2^{k+1}$, the description of a multiplicative cyclic group $\mathbb{G}_T$ of the same order, and a non-degenerate bilinear pairing $\hat{e}$: $\mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$. See [10] for a description of the properties of such pairings. We use $\mathbb{G}^*$ to denote $\mathbb{G} \setminus \{1\}$, i.e. the set of all group elements except the neutral element. Throughout the paper we use $\mathbb{PG} = (\mathbb{G}, \mathbb{G}_T, p, \hat{e}, g)$ as shorthand for the description of bilinear groups, where g is a generator of $\mathbb{G}$.

The BDDH assumption in $\mathbb{PG}$ is captured by defining the bddh-advantage of an adversary $\mathcal{B}$ as

$$\mathbf{Adv}^{\text{bddh}}_{\mathbb{PG},\mathcal{B}}(k) = |\Pr[\mathcal{B}(g^x, g^y, g^z, \hat{e}(g,g)^{xyz}) = 1] - \Pr[\mathcal{B}(g^x, g^y, g^z, g^r) = 1]|$$

where $x, y, z, r \xleftarrow{\$} \mathbb{Z}_p^*$.

Chameleon Hash based on DLOG. To apply our generic transformation we first recall a simple construction from [30] of a discrete-logarithm based Chameleon hash. The key generator creates a random group element $h = g^z$. The public hash key pk_{ch} is h, the trapdoor key sk_{ch} is z. Message space and randomness are defined as $\mathbb{Z}_p$. To hash an element $m \in \mathbb{Z}_p$ using randomness $s \in \mathbb{Z}_p$, compute $g^m h^s \leftarrow \mathsf{CMH.H}(pk_{ch}, m, s)$. To compute a collision for (m', s', m) the trapdoor algorithm $\mathsf{CMH.Trap}(sk_{ch} = z, m', s', m)$ computes $s = (m - m')/z + s' \bmod p$. if $m \neq m'$ so is $s \neq s'$. We verify the correctness of the trapdoor algorithm by $\mathsf{CMH.H}(pk_{ch}, m', s') = g^{m'} h^{s'} = g^{m'+zs'} = g^{m+zs} = \mathsf{CMH.H}(pk_{ch}, m, s)$. This Chameleon hash is prefix collision resistant under the the assumption that computing discrete logs in $\mathbb{G}$ is computational infeasible (DLOG assumption). Note that the BDDH assumption in $\mathbb{PG}$ implies the DLOG assumption in $\mathbb{G}$.

This Chameleon hash maps elements from $\mathbb{Z}_p$ to $\mathbb{G}$. Unfortunately, in our application we additionally need to modify it to map elements from *TagSp* $\times\, \mathbb{G}$ to $\mathbb{Z}_p$. To this end we employ two (target collision-resistant) hash functions $\mathsf{CR} : 0, 1^* \times \mathbb{G} \to \mathbb{G}$ and $\mathsf{TCR} : \mathbb{G} \to \mathbb{Z}_p$. The modified Chameleon Hash function is defined as $\mathsf{CMH.H}_{\text{dl}}(h, t||c_1, s) = \mathsf{TCR}(g^{\mathsf{CR}(t||c_1)} h^s)$. If CR is a collision resistant hash function and TCR is a target collision resistant hash function then the modified Chameleon Hash is prefix collision resistant.

6.1 Based on Boneh-Boyen's IBE

We first recall the (first) IBE scheme from Boneh and Boyen [6] which we already present in its partitioned IBKEM form $\mathcal{BB} = (\mathsf{BB.Kg}, \mathsf{BB.Key}, \mathsf{BB.Cipher1}, \mathsf{BB.Cipher2}, \mathsf{BB.Extract}, \mathsf{BB.Decaps})$.

```
BB.Kg($1^k$)
    $x_0, x_1, y \xleftarrow{\$} \mathbb{Z}_p^*$
    $u_0 \leftarrow g^{x_0}$ ; $u_1 \leftarrow g^{x_1}$
    $v \leftarrow \hat{e}(g,g)^y$
    $pk \leftarrow (u_0, u_1, v)$
    $sk \leftarrow (x_0, x_1, y)$
    Return $(pk, sk)$

BB.Key($pk$)
    $r \xleftarrow{\$} \mathbb{Z}_p^*$ ; $K \leftarrow v^r \in \mathbb{G}_T$
    Return $(K, \varphi = (pk, r))$

BB.Cipher1($\varphi = (pk, r)$)
    Return $c_1 \leftarrow g^r$

BB.Cipher2($id, \varphi = (pk, r)$)
    Return $c_2 \leftarrow (u_0 u_1^{id})^r$

BB.Extract($sk, id$)
    $s \xleftarrow{\$} \mathbb{Z}_p^*$
    $d_1 \leftarrow g^s$ ; $d_2 \leftarrow g^y \cdot (u_0 u_1^{id})^s$
    Return $sk[id] = (d_1||d_2)$

BB.Decaps($sk[id], id, c_1||c_2$)
    Parse $sk[id] = (d_1, d_2)$
    Return $K \leftarrow \hat{e}(c_1, d_2)/\hat{e}(c_2, d_1)$
```

The above IBKEM scheme is known to be selective-ID IND-CPA secure [6] under the BDDH assumption. We quickly verify it has the perfect \$-rejection property. Fix *pk* and *id* and let $\tilde{C} = (\tilde{c_1}, \tilde{c_2}) \notin \mathit{CipherSp}$ be a ciphertext that was not generated by the encapsulation algorithms. That means that $\tilde{c_1} = g^{r_1}$ and $\tilde{c_2} = (u_0 u_1^{id})^{r_2}$ for some $r_1 \neq r_2$. Then, probabilistic $\mathsf{IBKEM.Decaps}(\mathsf{IBKEM.Extract}(sk, id]), \tilde{c})$ picks a random $s \in \mathbb{Z}_p$ and returns

$$K = \hat{e}(\tilde{c_1}, g^y \cdot (u_0 u_1^{id})^s)/\hat{e}(\tilde{c_2}, g^s) = \hat{e}(g^{r_1}, g^y) \cdot \hat{e}(g^{r_1}, (u_0 u_1^{id})^s)/\hat{e}((u_0 u_1^{id})^{r_2}, g^s) = v^{r_1} \cdot \hat{e}(g, u_0 u_1^{id})^{s(r_1 - r_2)}.$$

Since for every decapsulation a fresh random value $s \in \mathbb{Z}_p$ is chosen, K is indeed a random key, uniformly distributed, and independent of anything else. On the other hand, a consistent ciphertext $\tilde{C} \in \mathit{CipherSp}$ has $r := r_1 = r_2$ and yield the correct key $K = v^r$.

Using the transformation from Section 4 and the modified discrete-log based Chameleon Hash $\mathsf{CMH.H}_{\text{dl}}(h, \cdot, \cdot)$ we get the following tag-KEM. As it was already done in [15, 12, 27, 28], decapsulation can be further simplified by using *sk* to reject all inconsistent ciphertexts that are not contained in *CipherSp*.

```
$\mathsf{TKEM.Kg}(1^k)$
    $(pk, sk) \overset{\$}{\leftarrow} \mathsf{BB.Kg}(1^k)$
    $z \overset{\$}{\leftarrow} \mathbb{Z}_p^*$ ; $h \leftarrow g^z$
    $pk' \leftarrow (pk, h)$
    Return $(pk', sk)$

$\mathsf{TKEM.Key}(pk')$
    $r \overset{\$}{\leftarrow} \mathbb{Z}_p^*$ ; $K \leftarrow v^r \in \mathbb{G}_T$
    Return $(K, \varphi = (pk', r))$

$\mathsf{TKEM.Cipher}(t, \varphi = (pk', r))$
    $c_1 \leftarrow g^r$
    $s \overset{\$}{\leftarrow} \mathbb{Z}_p$ ; $id \leftarrow \mathsf{CMH.H}_{\mathrm{dl}}(h, t||c_1, s)$
    $c_2 \leftarrow (u_0 u_1^{id})^r$
    Return $C = c_1||c_2||s$

$\mathsf{TKEM.Decaps}(sk, t, c_1||c_2||s)$
    $id \leftarrow \mathsf{CMH.H}_{\mathrm{dl}}(h, t||c_1, s)$
    If $c_1^{x_0 + id \cdot x_1} \neq c_2$
        then return $\perp$
    Else Return $K \leftarrow \hat{e}(c_1, g^y)$
```

Security of the above tag-KEM can be reduced by Theorem 3 to the security of the underlying IBKEM and the Chameleon hash. Hence the tag-KEM is IND-CCA under the BDDH assumption.

There is also a second IBE scheme in the paper by Boneh and Boyen [6], i.e. a (more efficient) scheme based on the stronger q-BDDHI assumption. We remark that applying our transformation to this scheme leads to a tag-KEM that is almost identical to the above tag-KEM. We therefore forbear from giving more details about the resulting here. This is reminiscent to what what happens to the two Boneh-Boyen IBE schemes applied to the CHK transformation from [16], see [28] for more details.

The above scheme uses a Chameleon Hash which additionally employs two hash functions to map elements to the proper domains. We now show a non-generic improvement of the above scheme that avoids these problems. The idea is to use an *implicit Chameleon Hash* in the exponent defined as $id = \mathsf{ICMH.H}(t', s) = t' + x_2 s$. In the public key of the Chameleon hash we only include $u_1^{x_2}$, so $u_1^{\mathsf{ICMH.H}(t',s)}$ can be publicly evaluated by computing $u_1^{t'} \cdot u_2^s$. Hence the element c_2 of the IBKEM ciphertext can be computed as $c_2 = (u_0 u_1^{id})^r = (u_0 u_1^{t'} \cdot u_2^s)^r$. Breaking prefix collision resistance of this implicit Chameleon hash is as hard as breaking the DLOG assumption in $\mathbb{G}$. A similar technique was already used in [7] to obtain short signatures without random oracles. Again, we need a collision-resistant hash function $\mathsf{CR} : \mathit{TagSp} \times \mathbb{G}_1 \rightarrow \mathbb{Z}_p$.

```
$\mathsf{TKEM.Kg}(1^k)$
    $(pk, sk) \overset{\$}{\leftarrow} \mathsf{BB.Kg}(1^k)$
    $x_2 \overset{\$}{\leftarrow} \mathbb{Z}_p^*$ ; $u_2 \leftarrow u_1^{x_2}$
    $pk' \leftarrow (pk, u_2)$
    $sk' \leftarrow (sk, x_2)$
    Return $(pk', sk')$

$\mathsf{TKEM.Key}(pk')$
    $r \overset{\$}{\leftarrow} \mathbb{Z}_p^*$ ; $K \leftarrow v^r \in \mathbb{G}_T$
    Return $(K, \varphi = (pk', r))$

$\mathsf{TKEM.Cipher}(t, \varphi = (pk', r))$
    $s \overset{\$}{\leftarrow} \mathbb{Z}_p^*$ ; $c_1 \leftarrow g^r$
    $t' \leftarrow \mathsf{CR}(c_1||t)$ ; $c_2 \leftarrow \left(u_0 u_1^{\mathsf{ICMH.H}(t',s)}\right)^r$
    Return $C = c_1||c_2||s$

$\mathsf{TKEM.Decaps}(sk', t, c_1||c_2||s)$
    $t' \leftarrow \mathsf{CR}(c_1||t)$
    $id \leftarrow \mathsf{ICMH.H}(t', s)$
    If $c_1^{x_0 + id \cdot x_1} \neq c_2$
        then return $\perp$
    Else Return $K \leftarrow \hat{e}(c_1, g^y)$
```

We apply Theorem 3 (with obvious modifications to take into account the implicit Chameleon hash) to prove that the above tag-KEM is IND-CCA secure if the hash function CR is collision resistant and the BDDH assumption holds in $\mathbb{PG}$.

6.2 Based on Waters' IBE

Viewing Waters' (adaptive-identity secure) IBE scheme [41] as an IBKEM we note that in fact it is already a partitioned IBKEM.

Let $\mathsf{CR} : \mathit{TagSp} \times \mathbb{G} \rightarrow \{0,1\}^n$ be a collision resistant hash function, where $n \approx k$ is a security parameter of the scheme. The transformation from Section 5.3 gives the following tag-KEM.

```
$\mathsf{TKEM.Kg}(1^k)$
    $x_0, x_1, \ldots, x_n, y \overset{\$}{\leftarrow} \mathbb{Z}_p^*$
    $u_0 \leftarrow g^{x_0}$ ; $\ldots$ ; $u_n \leftarrow g^{x_n}$
    $z \leftarrow \hat{e}(g, g)^y$
    $pk \leftarrow (u_0, \ldots, u_n, z)$
    $sk \leftarrow (x_0, \ldots, x_n, g^y)$
    Return $(pk, sk)$

$\mathsf{TKEM.Key}(pk)$
    $r \overset{\$}{\leftarrow} \mathbb{Z}_p^*$ ; $K \leftarrow z^r$
    Return $(K, \varphi = (pk, r))$

$\mathsf{TKEM.Cipher}(t, \varphi = (pk, r))$
    $c_1 \leftarrow g^r$
    $id \leftarrow \mathsf{CR}(t||c_1)$ ; $c_2 \leftarrow (u_0 \prod_{i=1}^n u_i^{id_i})^r$
    Return $C = c_1||c_2$

$\mathsf{PKE.Dec}(sk, t, c_1||c_2)$
    $id \leftarrow \mathsf{CR}(t||c_1)$
    If $c_1^{x_0 + \sum_{i=1}^n id_i x_i} \neq c_2$
        then return $\perp$
    Else Return $K \leftarrow \hat{e}(c_1, g^y)$
```

Here id_i means the ith bit of $id \in \{0,1\}^n$. In fact, one can readily verify that the proof in [41] already shows that, under the BDDH assumption, Waters' IBKEM is strongly adaptive-ID IND-CPA. Using the tag-KEM/DEM framework from Section 2.3 the obtained PKE scheme is the same as the "direct PKE scheme based on IBE techniques" from [12]. Security of the PKE scheme can now be directly understood in terms of Theorem 7, i.e. by reducing it to the security of the hash function plus the strong security properties of Water's original IBE scheme. In contrast, [12] had to rely on a dedicated proof.

6.3 Based on Gentry's IBE

Viewing Gentry' (adaptive-identity secure) IBE scheme [25] as an IBKEM we note that it is also partitioned. The transformation from Section 5.3 gives the following tag-KEM. Again, we can greatly simplify decapsulation by directly accessing the secret key. Let $\mathsf{CR} : \mathit{TagSp} \times \mathbb{G}_T \rightarrow \mathbb{Z}_p$ be a collision resistant hash function.

```
TKEM.Kg($1^k$)
    $x_0, y \stackrel{\$}{\leftarrow} \mathbb{Z}_p^*$
    $u_0 \leftarrow g^{x_0}$ ; $v \leftarrow \hat{e}(g,g)^y$
    $pk \leftarrow (u_0, v)$
    $sk \leftarrow (x_0, y)$
    Return $(pk, sk)$

TKEM.Key($pk$)
    $r \stackrel{\$}{\leftarrow} \mathbb{Z}_p^*$ ; $K \leftarrow v^r$
    Return $(K, \varphi = (pk, r))$

TKEM.Cipher($t, \varphi = (pk, r)$)
    $c_1 \leftarrow \hat{e}(g,g)^r$ ; $id \leftarrow \mathsf{CR}(t||c_1)$
    $c_2 \leftarrow (u_0 \cdot g^{id})^r$
    Return $C = c_1||c_2$

TKEM.Decaps($sk, t, c_1||c_2$)
    $id \leftarrow \mathsf{CR}(t||c_1)$
    If $\hat{e}(c_2, g^{x_0+id}) \neq c_1$
        then return $\perp$
    Else Return $K \leftarrow c_1^y$
```

Again one can verify that the proof in [25] already shows that, under the q-ABDHE assumption, Gentry's IBKEM is strongly adaptive-ID IND-CPA. Using the tag-KEM/DEM framework from Section 2.3 security of the obtained PKE scheme can be reduced to the security of the hash function plus the security of Gentry's original IBE scheme.

6.4 Comparison

A quick comparison of all tag KEMs discussed in this section is given in Table 2. As security parameter we fixed $k = 128$ bits. For pairing groups that means that elements in $\mathbb{G}$ can be represented in ≈ 256 bits and elements in $\mathbb{G}_T$ can be represented in 3072 bits. We remark that using the concept of sequential multiplications all decapsulation algorithms can be optimized to use only one pairing plus one sequential exponentiation. The security reduction of Waters' IBE to the BDDH assumption is not tight, i.e. it introduces a $\log q$ bit of security, where q is the maximal number of decryption queries made by an adversary. Also note that the key-size of the Waters' IBE based tag-KEM from can be reduced by the factor of l by loosing any other l bits of security in the reduction [18].

Tag-KEM	Security	Ciphertext	Encryption	Decryption	Keysize
	Assumption	Overhead	#pairings + #[multi,regular]-exp		(pk/sk)
§6.1	BDDH	768	$0 + [1,2]$	$1 + [0,1]$	4/3
§6.1 (implicit CMH)	BDDH	768	$0 + [1,1]$	$1 + [0,1]$	4/4
§6.2	BDDH	512	$0 + [3,0]$	$1 + [0,1]$	$\approx 130/130$
§6.3	q-ABDHE	3200	$0 + [3,0]$	$1 + [0,1]$	2/2

Table 2. Efficiency comparison for our chosen-ciphertext secure tag KEMs. For efficiency we count the number of pairings + [multi exponentiations, regular exponentiations] used for encryption, decryption, and key generation. All "symmetric" operations (such as the hash function) are ignored. Ciphertext overhead represents the expected difference (in bits) between ciphertext and plaintext length for $k = 128$ bit security. For comparison we mention that relative timings for the various operations are as follows: bilinear pairing $\approx 3 - 5$ [35], multi(=sequential)-exponentiation ≈ 1.2 [5], and regular exponentiation $= 1$.

7 Extensions

7.1 Application to Threshold Cryptography

Observe that, the generic transformation given in Section 4 does not require any private-key operation. The chameleon hashing in the decryption process comes *before* the decryption of the underlying IBKEM. Accordingly, if the underlying partitioned IBKEM is publicly verifiable, so is the resulting tag-KEM. This feature is quite useful in the threshold setting where every decryption server needs to be convinced of the correctness of the ciphertext, preferable without interacting with other servers.

Based on the above observation, we extend our framework from Section 4 to construct efficient threshold PKE schemes from selective-ID partitioned threshold IBKEMs. We quickly sketch the ingredients of this transformation.

- A threshold version of our generic transformation that transforms threshold selective-ID partitioned IBKEMs into threshold tag-KEMs.
- An extension of the tag-KEM/DEM framework to the threshold setting. Every threshold tag-KEM can be combined with a passively secure DEM yielding efficient threshold PKE. This construction was already briefly mentioned in the original work of [1].
- A concrete example of a threshold selective-ID partitioned IBKEM that is secure on the BDDH assumption.

Putting all three ingredients together we get an efficient threshold PKE scheme which is IND-CCA secure under the BDDH assumption. Similar to the standard (non-threshold) case our proposal introduces a ciphertext overhead of 128 bits. A more detailed construction including formal definitions and proofs will be given in the full version of the paper.

An existing proposal of a threshold PKE scheme was giving in [8]. This construction makes use of the CHK-transformation [16] based on a one-time signature scheme and therefore has a ciphertext overhead of roughly $65k$ bits. We note that the more efficient BK-transformation [11] is not applicable in this setting since it does not provide public verifiability of the threshold ciphertexts.

7.2 CCA-secure (Hierarchical) Identity-Based Encryption

Similar to the case of the CHK and BK transformations [17, 11] our transformations from Section 4 generalize to the setting of (hierarchical) identity-based encryption (HIBE) [26]. In particular, any partitioned selective-ID secure k-level HIB-KEM can be transformed into an IND-CCA secure $\ell - 1$-level HIBE.

7.3 Tag-based KEMs

For the CHK and BK transformations [17, 11] from selective-ID secure IBE to CCA-secure PKE it was recently shown in [27] that a more general concept called *tag-based encryption* (TBE) is already sufficient for the transformations to work. TBE can be understood as IBE without the necessity to perform key derivation. In contrast to IBE there exists known instantiations of TBE schemes without pairings, for instance on the linear assumption [27]. We remark that our transformations can also be stated in terms of *partitioned tag-based KEMs* leading to more example instantiations than given in Section 6.

References

1. M. Abe, R. Gennaro, and K. Kurosawa. Tag-KEM/DEM: A new framework for hybrid encryption. IACR e-print 2005/027, 2005.
2. Masayuki Abe. Robust distributed multiplication without interaction. In Michael J. Wiener, editor, *Advances in Cryptology – CRYPTO'99*, volume 1666 of *Lecture Notes in Computer Science*, pages 130–147, Santa Barbara, CA, USA, August 15–19, 1999. Springer-Verlag, Berlin, Germany.
3. Mihir Bellare and Phillip Rogaway. Random oracles are practical: A paradigm for designing efficient protocols. In *ACM CCS 93: 1st Conference on Computer and Communications Security*, pages 62–73, Fairfax, Virginia, USA, November 3–5, 1993. ACM Press.

4. Mihir Bellare and Phillip Rogaway. Collision-resistant hashing: Towards making UOWHFs practical. In Burton S. Kaliski Jr., editor, *Advances in Cryptology – CRYPTO'97*, volume 1294 of *Lecture Notes in Computer Science*, pages 470–484, Santa Barbara, CA, USA, August 17–21, 1997. Springer-Verlag, Berlin, Germany.
5. D. J. Bernstein. Pippenger's exponentiation algorithm, 2001. `http://cr.yp.to/papers.html`.
6. Dan Boneh and Xavier Boyen. Efficient selective-ID secure identity based encryption without random oracles. In Christian Cachin and Jan Camenisch, editors, *Advances in Cryptology – EUROCRYPT 2004*, volume 3027 of *Lecture Notes in Computer Science*, pages 223–238, Interlaken, Switzerland, May 2–6, 2004. Springer-Verlag, Berlin, Germany.
7. Dan Boneh and Xavier Boyen. Short signatures without random oracles. In Christian Cachin and Jan Camenisch, editors, *Advances in Cryptology – EUROCRYPT 2004*, volume 3027 of *Lecture Notes in Computer Science*, pages 56–73, Interlaken, Switzerland, May 2–6, 2004. Springer-Verlag, Berlin, Germany.
8. Dan Boneh, Xavier Boyen, and Shai Halevi. Chosen ciphertext secure public key threshold encryption without random oracles. In David Pointcheval, editor, *Topics in Cryptology – CT-RSA 2006*, volume 3860 of *Lecture Notes in Computer Science*, pages 226–243, San Jose, CA, USA, February 13–17, 2006. Springer-Verlag, Berlin, Germany.
9. Dan Boneh and Matthew K. Franklin. Identity-based encryption from the Weil pairing. In Joe Kilian, editor, *Advances in Cryptology – CRYPTO 2001*, volume 2139 of *Lecture Notes in Computer Science*, pages 213–229, Santa Barbara, CA, USA, August 19–23, 2001. Springer-Verlag, Berlin, Germany.
10. Dan Boneh and Matthew K. Franklin. Identity based encryption from the Weil pairing. *SIAM Journal on Computing*, 32(3):586–615, 2003.
11. Dan Boneh and Jonathan Katz. Improved efficiency for CCA-secure cryptosystems built using identity-based encryption. In Alfred Menezes, editor, *Topics in Cryptology – CT-RSA 2005*, volume 3376 of *Lecture Notes in Computer Science*, pages 87–103, San Francisco, CA, USA, February 14–18, 2005. Springer-Verlag, Berlin, Germany.
12. Xavier Boyen, Qixiang Mei, and Brent Waters. Direct chosen ciphertext security from identity-based techniques. In *ACM CCS 05: 12th Conference on Computer and Communications Security*, pages 320–329, Alexandria, Virginia, USA, November 7–11, 2005. ACM Press.
13. Ran Canetti, Oded Goldreich, and Shai Halevi. The random oracle methodology, revisited. In *30th Annual ACM Symposium on Theory of Computing*, pages 209–218, Dallas, Texas, USA, May 23–26, 1998. ACM Press.
14. Ran Canetti and Shafi Goldwasser. An efficient threshold public key cryptosystem secure against adaptive chosen ciphertext attack. In Jacques Stern, editor, *Advances in Cryptology – EUROCRYPT'99*, volume 1592 of *Lecture Notes in Computer Science*, pages 90–106, Prague, Czech Republic, May 2–6, 1999. Springer-Verlag, Berlin, Germany.
15. Ran Canetti, Shai Halevi, and Jonathan Katz. A forward-secure public-key encryption scheme. In Eli Biham, editor, *Advances in Cryptology – EUROCRYPT 2003*, volume 2656 of *Lecture Notes in Computer Science*, pages 255–271, Warsaw, Poland, May 4–8, 2003. Springer-Verlag, Berlin, Germany.
16. Ran Canetti, Shai Halevi, and Jonathan Katz. Chosen-ciphertext security from identity-based encryption. In Christian Cachin and Jan Camenisch, editors, *Advances in Cryptology – EUROCRYPT 2004*, volume 3027 of *Lecture Notes in Computer Science*, pages 207–222, Interlaken, Switzerland, May 2–6, 2004. Springer-Verlag, Berlin, Germany.
17. Ran Canetti, Shai Halevi, and Jonathan Katz. Adaptively-secure, non-interactive public-key encryption. In Joe kilian, editor, *TCC 2005: 2nd Theory of Cryptography Conference*, volume 3378 of *Lecture Notes in Computer Science*, pages 150–168, Cambridge, MA, USA, February 10–12, 2005. Springer-Verlag, Berlin, Germany.
18. Sanjit Chatterjee and Palash Sarkar. Trading time for space: Towards an efficient ibe scheme with short(er) public parameters in the standard model. Proceedings of ICISC 2005, 2005.
19. Ronald Cramer and Victor Shoup. A practical public key cryptosystem provably secure against adaptive chosen ciphertext attack. In Hugo Krawczyk, editor, *Advances in Cryptology – CRYPTO'98*, volume 1462 of *Lecture Notes in Computer Science*, pages 13–25, Santa Barbara, CA, USA, August 23–27, 1998. Springer-Verlag, Berlin, Germany.
20. 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.
21. Ivan Damgård. Collision free hash functions and public key signature schemes. In David Chaum and Wyn L. Price, editors, *Advances in Cryptology – EUROCRYPT'87*, volume 304 of *Lecture Notes in Computer Science*, pages 203–216, Amsterdam, The Netherlands, April 13–15, 1988. Springer-Verlag, Berlin, Germany.
22. Danny Dolev, Cynthia Dwork, and Moni Naor. Nonmalleable cryptography. *SIAM Journal on Computing*, 30(2):391–437, 2000.
23. Shimon Even, Oded Goldreich, and Silvio Micali. On-line/off-line digital signatures. *Journal of Cryptology*, 9(1):35–67, 1996.

24. David Galindo and Eike Kiltz. Threshold chosen-ciphertext secure identity-based key encapsulation without random oracles. In *SCN 2006*, volume 4116, pages 173–185. Springer-Verlag, 2006.
25. Craig Gentry. Practical identity-based encryption without random oracles. In Serge Vaudenay, editor, *Advances in Cryptology – EUROCRYPT 2006*, volume 4004 of *Lecture Notes in Computer Science*, pages 445–464, St. Petersburg, Russia, May 28 – June 1, 2006. Springer-Verlag, Berlin, Germany.
26. Craig Gentry and Alice Silverberg. Hierarchical ID-based cryptography. In Yuliang Zheng, editor, *Advances in Cryptology – ASIACRYPT 2002*, volume 2501 of *Lecture Notes in Computer Science*, pages 548–566, Queenstown, New Zealand, December 1–5, 2002. Springer-Verlag, Berlin, Germany.
27. Eike Kiltz. Chosen-ciphertext security from tag-based encryption. In Shai Halevi and Tal Rabin, editors, *TCC 2006: 3rd Theory of Cryptography Conference*, volume 3876 of *Lecture Notes in Computer Science*, pages 581–600, New York, NY, USA, March 4–7, 2006. Springer-Verlag, Berlin, Germany.
28. Eike Kiltz. On the limitations of the spread of an IBE-to-PKE transformation. In Moti Yung, Yevgeniy Dodis, Aggelos Kiayias, and Tal Malkin, editors, *PKC 2006: 9th International Conference on Theory and Practice of Public Key Cryptography*, volume 3958 of *Lecture Notes in Computer Science*, pages 274–289, New York, NY, USA, April 24–26, 2006. Springer-Verlag, Berlin, Germany.
29. Eike Kiltz and David Galindo. Direct chosen-ciphertext secure identity-based key encapsulation without random oracles. In *ACISP 2006*, volume 4058, pages 336–347. Springer-Verlag, 2006.
30. Hugo Krawczyk and Tal Rabin. Chameleon signatures. In *ISOC Network and Distributed System Security Symposium – NDSS 2000*, San Diego, California, USA, February 2–4, 2000. The Internet Society.
31. T. Malkin, R. Moriarty, and N. Yakovenko. Generalized environmental security from number theoretic assumptions. In *TCC'06*, pages 343–359, 2006.
32. M. Naor. Bit commitment using pseudo-randomness. *Journal of Cryptology*, 4(2):151–158, 1991.
33. Moni Naor and Moti Yung. Universal one-way hash functions and their cryptographic applications. In *21st Annual ACM Symposium on Theory of Computing*, pages 33–43, Seattle, Washington, USA, May 15–17, 1989. ACM Press.
34. Moni Naor and Moti Yung. Public-key cryptosystems provably secure against chosen ciphertext attacks. In *22nd Annual ACM Symposium on Theory of Computing*, Baltimore, Maryland, USA, May 14–16, 1990. ACM Press.
35. D. Page, N.P. Smart, and F. Vercauteren. A comparison of MNT curves and supersingular curves. Cryptology ePrint Archive, Report 2004/165, 2004. `http://eprint.iacr.org/`.
36. T. P. Pedersen. Non-interactive and information-theoretic secure verifiable secret sharing. In J. Feigenbaum, editor, *Advances in Cryptology — CRYPTO '91*, volume 576 of *LNCS*, pages 129–140. Springer-Verlag, 1992.
37. M. Prabhakaran and A. Sahai. New notions of security: Achieving universal composability without trusted setup. In *STOC'04*, pages 242–251. ACM, 2004.
38. Charles Rackoff and Daniel R. Simon. Non-interactive zero-knowledge proof of knowledge and chosen ciphertext attack. In Joan Feigenbaum, editor, *Advances in Cryptology – CRYPTO'91*, volume 576 of *Lecture Notes in Computer Science*, pages 433–444, Santa Barbara, CA, USA, August 11–15, 1992. Springer-Verlag, Berlin, Germany.
39. Ryuichi Sakai, Kiyoshi Ohgishi, and Masao Kasahara. Cryptosystems based on pairing. In *SCIS 2000*, Okinawa, Japan, January 2000.
40. Adi Shamir. Identity-based cryptosystems and signature schemes. In G. R. Blakley and David Chaum, editors, *Advances in Cryptology – CRYPTO'84*, volume 196 of *Lecture Notes in Computer Science*, pages 47–53, Santa Barbara, CA, USA, August 19–23, 1985. Springer-Verlag, Berlin, Germany.
41. Brent R. Waters. Efficient identity-based encryption without random oracles. In Ronald Cramer, editor, *Advances in Cryptology – EUROCRYPT 2005*, volume 3494 of *Lecture Notes in Computer Science*, pages 114–127, Aarhus, Denmark, May 22–26, 2005. Springer-Verlag, Berlin, Germany.

A Supplemental Definitions

A.1 Public-Key Encryption

A public-key encryption (PKE) scheme consists of three algorithms $\mathcal{PKE} = (\mathsf{PKE.kg}, \mathsf{PKE.Enc}, \mathsf{PKE.Dec})$.

$(pk, sk) \overset{\$}{\leftarrow} \mathsf{PKE.kg}(1^k)$; A probabilistic key-generation algorithm that produces a master key pair for given security parameter $k \in \mathbb{N}$. The public-key pk defines a message space $MsgSp$.

$C \overset{\$}{\leftarrow} \mathsf{PKE.Enc}(pk, M)$; A probabilistic algorithm that outputs a ciphertext C for a message $M \in MsgSp$.

$M \overset{\$}{\leftarrow} \mathsf{PKE.Dec}(sk, C)$; A probabilistic decryption algorithm that decrypts ciphertext C to recover a message M. It may also output a special symbol $\perp$ to present rejection.

For consistency, we require that for all $k \in \mathbb{N}$, all messages $M \in \mathit{MsgSp}$, it must hold that $\Pr[\mathsf{PKE.Dec}(sk, \mathsf{PKE.Enc}(pk, M))$ $M] = 1$, where the probability is taken over the above randomized algorithms.

The security we require for PKE is IND-CCA security [38]. It is captured by defining the following experiment.

```
Experiment $\mathbf{Exp}^{pke\text{-cca}}_{\mathcal{PKE},\mathcal{A}}(k)$
  $(pk, sk) \xleftarrow{\$} \mathsf{PKE.kg}(1^k)$
  $(M_0, M_1, St_1) \xleftarrow{\$} \mathcal{A}_1^{\mathsf{PKE.Dec}(sk,\cdot)}(pk)$
  $b \xleftarrow{\$} \{0,1\}$ ; $C^* \xleftarrow{\$} \mathsf{PKE.Enc}(pk, M_b)$
  $b' \xleftarrow{\$} \mathcal{A}_2^{\mathsf{PKE.Dec}(sk,\cdot)}(C^*, St_1)$
  If $b \neq b'$ then return 0 else return 1.
```

The adversary $\mathcal{A}_2$ is restricted not to ask C^* to the decryption oracle $\mathsf{PKE.Dec}$ and the two messages M_0 and M_1 have to be of equals length.

We define the advantage of $\mathcal{A}$ in the chosen-ciphertext experiment as

$$\mathbf{Adv}^{pke\text{-cca}}_{\mathcal{PKE},\mathcal{A}}(k) \;=\; |\Pr[\mathbf{Exp}^{pke\text{-cca}}_{\mathcal{PKE},\mathcal{A}}(k) = 1] - 1/2| \;.$$

A PKE scheme $\mathcal{PKE}$ is said to be indistinguishable against chosen-ciphertext attacks (IND-CCA secure in short) if the advantage function $\mathbf{Adv}^{pke\text{-cca}}_{\mathcal{PKE},\mathcal{A}}(k)$ is a negligible function in k for all polynomial-time adversaries $\mathcal{A}$.

A.2 Data Encapsulation Mechanism

A data encapsulation mechanism (DEM, in short) $\mathcal{DEM} = (\mathsf{DEM.Enc}, \mathsf{DEM.Dec})$ with key-space $\{0,1\}^k$ is specified by its encryption algorithm $\mathsf{DEM.Enc}$ and decryption algorithm $\mathsf{DEM.Dec}$. The DEM needs to be indistinguishable against one-time attacks captured by defining the ind-ot-advantage of an adversary $\mathcal{B}$ as

$$\mathbf{Adv}^{dem\text{-ind-ot}}_{\mathcal{DEM},\mathcal{B}}(k) = |\Pr[b = b' \;:\; K_S \xleftarrow{\$} \{0,1\}^k \;;\; b \xleftarrow{\$} \{0,1\} \;;\; b' \xleftarrow{\$} \mathcal{B}^{\mathsf{LOR}(K_s,\cdot,\cdot,b)}(1^k)] - 1/2|$$

Above, $\mathsf{LOR}(K_S, M_0, M_1, b)$ returns $\psi \xleftarrow{\$} \mathsf{DEM.Enc}(K_S, M_b)$. $\mathcal{B}$ is allowed only one query to this left-or-right encryption oracle, consisting of a pair of equal-length messages.

B Transformation using RPC

This section fleshes out the modified transformation mentioned in Section 5.2 and its security proof. The modification applies only to $\mathsf{TKEM.Cipher}$ and $\mathsf{TKEM.Decaps}$ as shown below. All other functions remain as described in Section 5.2.

```
$\mathsf{TKEM.Cipher}(pk, t, \varphi)$
    $c_1 \leftarrow \mathsf{IBKEM.Cipher1}(\varphi)$
    $s \xleftarrow{\$} \mathsf{Rand_{RPC}}$ ; $id \leftarrow \mathsf{RPC}(s||t||c_1)$
    $c_2 \leftarrow \mathsf{IBKEM.Cipher2}(\varphi, id)$
    Return $c \leftarrow c_1||c_2||s$
```

```
$\mathsf{TKEM.Decaps}(sk, t, c)$
    $c_1||c_2||s \leftarrow c$
    $id \leftarrow \mathsf{RPC}(s||t||c_1)$
    $sk[id] \xleftarrow{\$} \mathsf{IBKEM.Extract}(sk, id)$
    $K \leftarrow \mathsf{IBKEM.Decaps}(sk[id], id, (c_1, c_2))$
    Return $K$
```

Let $\mathsf{RPC} : \mathsf{Rand_{RPC}} \times \{0,1\}^* \to \mathit{IDSp}$ be a random prefix collision resistant hash function. For $id \in \mathit{IDSp}$ and $x \in \{0,1\}^*$, define $S_x(id)$ by $S_x(id) = \{s \in \mathsf{Rand_{RPC}} \mid id = \mathsf{RPC}(s||x)\}$. We assume that RPC has a property that for every x the distribution of id is statistically close to uniform when s is taken uniformly. For the sake of this property $|\mathsf{Rand_{RPC}}| \geq |\mathit{IDSp}|$ is needed. Let $\mathcal{O}_{\text{p-preimg}}$ be a hash partial preimage oracle for RPC that takes as input $(id, x) \in \mathit{IDSp} \times \{0,1\}^*$ and outputs $s^* \xleftarrow{\$} S_x(id)$. It outputs $\perp$ if $S_x(id)$ is empty. From the above statistical property, $\perp$ is returned only with negligible probability when id is chosen randomly.

Theorem 8. *If the partitioned IBKEM is selective-ID IND-CPA secure with regard to all adversaries that have oracle access to* $\mathcal{O}_{\text{p-preimg}}$ *and* RPC *is random prefix collision resistant, then the above TKEM is IND-CCA secure.*

Proof. The outline of the proof is the same as that of Theorem 3. The following describes the difference only. We first show an adversary $\mathcal{B}$ against the selective-ID IND-CPA security of $\mathcal{IBKEM}$ as follows.

Setup. Given security parameter k, adversary $\mathcal{B}$ chooses RPC and selects the target identity $id^* \overset{\$}{\leftarrow} IDSp$. It sends id^* to the challenger and receives a public-key pk and a challenge (K_b^*, c_1^*, c_2^*). Adversary $\mathcal{B}$ now runs $\mathcal{A}$ by giving pk and pk_{ch}.

Challenge Simulation. At some point, $\mathcal{A}$ outputs a target tag t^*. $\mathcal{B}$ then sends $(id^*, t^*||c_1^*)$ to $\mathcal{O}_{\text{p-preimg}}$. If $\perp$ is returned, abort. Otherwise, $s^* \in S_{t^*||c_1^*}(id^*)$ is obtained. $\mathcal{B}$ sends K_b^* and (c_1^*, c_2^*, s^*) to $\mathcal{A}$.

"Decryption Oracle Simulation" and "Output" are unchanged.

Challenge simulation will be successful with overwhelming probability by the statistical property of RPC. Furthermore, the distribution of selected id^* and s^* are statistically close to the real execution where s^* is selected first. The rest of analysis is unchanged and hence $\mathcal{B}$ simulates $\mathcal{A}$'s view in the IND-CCA experiment statistically close to the real execution unless event COL happens.

If COL does not happen then $\mathcal{B}$ has the same advantage in winning the experiment as $\mathcal{A}$ except for the negligible simulation errors. If COL happens during $\mathcal{A}$'s simulation with noticeable probability, we use $\mathcal{A}$ to build an adversary $\mathcal{C}$ against the random prefix collision resistance property of RPC. Adversary $\mathcal{C}$ will attack RPC by correctly computing the IBKEM part with the secret-key generated by itself and perfectly simulates $\mathcal{A}$'s view in the IND-CCA experiment till $\mathcal{A}$ outputs a collision. □