RKA Security beyond the Linear Barrier: IBE, Encryption and Signatures

Mihir Bellare[1] Kenneth G. Paterson[2] Susan Thomson[3]

August 2012

Abstract

We provide a framework enabling the construction of IBE schemes that are secure under related-key attacks (RKAs). Specific instantiations of the framework yield RKA-secure IBE schemes for sets of related key derivation functions that are non-linear, thus overcoming a current barrier in RKA security. In particular, we obtain IBE schemes that are RKA secure for sets consisting of all affine functions and all polynomial functions of bounded degree. Based on this we obtain the first constructions of RKA-secure schemes for the same sets for the following primitives: CCA-secure public-key encryption, CCA-secure symmetric encryption and Signatures. All our results are in the standard model and hold under reasonable hardness assumptions.

[1] Department of Computer Science & Engineering, University of California San Diego, 9500 Gilman Drive, La Jolla, California 92093, USA. Email: mihir@eng.ucsd.edu. URL: cseweb.ucsd.edu/~mihir/. Supported in part by NSF grants CNS-0904380, CCF-0915675 and CNS-1116800.

[2] Information Security Group, Royal Holloway, University of London, Egham Hill, Egham, Surrey TW20 0EX, UK. Email: kenny.paterson@rhul.ac.uk. URL: www.isg.rhul.ac.uk/~kp. Supported by EPSRC Leadership Fellowship EP/H005455/1.

[3] Information Security Group, Royal Holloway, University of London, Egham Hill, Egham, Surrey TW20 0EX, UK. Email: s.thomson@rhul.ac.uk. URL: www.sthomson.co.uk. Supported by EPSRC Leadership Fellowship EP/H005455/1.

Contents

1 Introduction

Related-key attacks (RKAs) were first conceived as tools for the cryptanalysis of blockciphers [24, 8]. However, the ability of attackers to modify keys stored in memory via tampering [13, 9] raises concerns that RKAs can actually be mounted in practice. The key could be an IBE master key, a signing key of a certificate authority, or a decryption key, making RKA security important for a wide variety of primitives.

Provably achieving security against RKAs, however, has proven extremely challenging. This paper aims to advance the theory with new feasibility results showing achievability of security under richer classes of attacks than previously known across a variety of primitives.

Contributions in brief. The primitive we target in this paper is IBE. RKA security for this primitive was defined by Bellare, Cash, and Miller [4]. As per the founding theoretical treatment of RKAs by Bellare and Kohno [5], the definition is parameterized by the class Φ of functions that the adversary is allowed to apply to the target key. (With no restrictions, security is unachievable.) For future reference we define a few relevant classes of functions over the space $\mathcal{S}$ of master keys. The set $\Phi^c = \{\phi_c\}_{c\in\mathcal{S}}$ with $\phi_c(s) = c$ is the set of constant functions. If $\mathcal{S}$ is a group under an operation $*$ then $\Phi^{\rm lin} = \{\phi_a\}_{a\in\mathcal{S}}$ with $\phi_a(s) = a * s$ is the class of linear functions. (Here $*$ could be multiplication or addition.) If $\mathcal{S}$ is a field we let $\Phi^{\rm aff} = \{\phi_{a,b}\}_{a,b\in\mathcal{S}}$ with $\phi_{a,b}(s) = as + b$ be the class of affine functions and $\Phi^{{\rm poly}(d)} = \{\phi_q\}_{q\in\mathcal{S}_d[x]}$ with $\phi_q(s) = q(s)$ the class of polynomial functions, where q ranges over the set $\mathcal{S}_d[x]$ of polynomials over $\mathcal{S}$ of degree at most d. RKA security increases and is a more ambitious target as we move from $\Phi^{\rm lin}$ to $\Phi^{\rm aff}$ to $\Phi^{{\rm poly}(d)}$.

The choice of IBE as a primitive is not arbitrary. First, IBE is seeing a lot of deployment, and compromise of the master secret key would cause widespread damage, so we are well motivated to protect it against side-channel attacks. Second, IBE was shown in [4] to be an enabling primitive in the RKA domain: achieving RKA-secure IBE for any class Φ immediately yields Φ-RKA-secure CCA-PKE (CCA-secure public-key encryption) and Sig (signature) schemes. These results were obtained by noting that the CHK [12] IBE-to-CCA-PKE transform and the Naor IBE-to-Sig transform both preserve RKA security. Thus, results for IBE would immediately have wide impact.

We begin by presenting attacks showing that existing IBE schemes such as those of Boneh-Franklin [14] and Waters [27] are not RKA secure, even for $\Phi^{\rm lin}$. This means we must seek new designs.

We present a framework for constructing RKA-secure IBE schemes. It is an adaptation of the framework of Bellare and Cash [3] that builds RKA-secure PRFs based on key-malleable PRFs and fingerprinting. Our framework has two corresponding components. First, we require a starting IBE scheme that has a key-malleability property relative to our target class Φ of related-key deriving functions. Second, we require the IBE scheme to support what we call collision-resistant identity renaming. We provide a simple and efficient way to transform any IBE scheme with these properties into one that is Φ-RKA secure.

To exploit the framework, we must find key-malleable IBE schemes. Somewhat paradoxically, we show that the very attack strategies that broke the RKA security of existing IBE schemes can be used to show that these schemes are Φ-key-malleable, not just for $\Phi = \Phi^{\rm lin}$ but even for $\Phi = \Phi^{\rm aff}$. We additionally show that these schemes support efficient collision-resistant identity renaming. As a consequence we obtain $\Phi^{\rm aff}$-RKA-secure IBE schemes based on the same assumptions used to prove standard IBE security of the base IBE schemes.

From the practical perspective, the attraction of these results is that our schemes modify the known ones in a very small and local way limited only to the way identities are hashed. They thus not only preserve the efficiency of the base schemes, but implementing them would require minimal and modular software changes, so that non-trivial RKA security may be added without much increase in cost. From the theoretical perspective, the step of importance here is to be able to achieve RKA security for non-linear functions, and this without extra computational assumptions. As we will see below, linear RKAs,

Primitive	Linear	Affine	Polynomial
IBE	[4]+[3]	✓	✓
Sig	[4]+[3]	✓	✓
CCA-PKE	[28], [4]+[3]	✓	✓
CPA-SE	[2], [4]+[3]	[22]	[22]
CCA-SE	[4]+[3]	✓	✓*
PRF	[3]	–	–

Figure 1: Rows are indexed by primitives. Columns are indexed by the class Φ of related-key derivation functions, $\Phi^{\text{lin}}, \Phi^{\text{aff}}$ and $\Phi^{\text{poly}(d)}$ respectively. Entries indicate work achieving Φ-RKA security for the primitive in question. Checkmarks indicate results from this paper that bring many primitives all the way to security under polynomial RKAs in one step. The table only considers achieving the strong, adaptive notions of security from [4]; non-adaptively secure signature schemes for non-linear RKAs were provided in [22]. Note that symmetric key primitives cannot be RKA secure against constant RKD functions, so affine and polynomial RKA security for the last three rows is with respect to the RKD sets $\Phi^{\text{aff}} \setminus \Phi^c$ and $\Phi^{\text{poly}(d)} \setminus \Phi^c$. The "*" in the CCA-SE row is because our CCA-SE construction is insecure against RKD functions where the linear coefficient is zero, so does not achieve RKA security against the full set $\Phi^{\text{poly}(d)} \setminus \Phi^c$.

meaning Φ^{lin}-RKA security, has so far been a barrier for most primitives.

However, we can go further, providing a $\Phi^{\text{poly}(d)}$-RKA-secure IBE scheme. Our scheme is an extension of Waters' scheme [27]. The proof is under a q-type hardness assumption that we show holds in the generic group model. The significance of this result is to show that for IBE we can go well beyond linear RKAs, something not known for PRFs.

As indicated above, we immediately get Φ-RKA-secure CCA-PKE and Sig schemes for any class Φ for which we obtained Φ-RKA-secure IBE schemes, and under the same assumptions. When the base IBE scheme has a further malleability property, the CCA-PKE scheme so obtained can be converted into a Φ-RKA-secure CCA-SE (CCA-secure symmetric encryption) scheme. This yields the first RKA secure schemes for the primitives Sig, CCA-PKE, and CCA-SE for non-linear RKAs, meaning beyond Φ^{lin}.

Background and context. The theoretical foundations of RKA security were laid by Bellare and Kohno [5], who treated the case of PRFs and PRPs. Research then expanded to consider other primitives [21, 2, 22, 4]. In particular, Bellare, Cash and Miller [4] provide a comprehensive treatment including strong definitions for many primitives and ways to transfer Φ-RKA security from one primitive to another.

RKA-security is finding applications beyond providing protection against tampering-based sidechannel attacks [20], including instantiating random oracles in higher-level protocols and improving efficiency [2, 1].

With regard to achieving security, early efforts were able to find PRFs with proven RKA security only for limited Φ or under very strong assumptions. Eventually, using new techniques, Bellare and Cash [3] were able to present DDH-based PRFs secure against linear RKAs ($\Phi = \Phi^{\text{lin}}$). But it is not clear how to take their techniques further to handle larger RKA sets Φ.

Figure 1 summarizes the broad position. Primitives for which efforts have now been made to achieve RKA security include CPA-SE (CPA secure symmetric encryption), CCA-SE (CCA secure symmetric encryption), CCA-PKE (CCA secure public-key encryption[1]) Sig (Signatures), and IBE (CPA secure

[1] RKAs are interesting for symmetric encryption already in the CPA case because encryption depends on the secret key, but for public-key encryption they are only interesting for the CCA case because encryption does not depend on the

identity-based encryption). Schemes proven secure under a variety of assumptions have been provided. But the salient fact that stands out is that prior to our work, results were all for linear RKAs with the one exception of CPA-SE where a scheme secure against polynomial (and thus affine) RKAs was provided by [22].

In more detail, Bellare, Cash and Miller [4] show how to transfer RKA security from PRF to any other primitive, assuming an existing standard-secure instance of the primitive. Combining this with [3] yields DDH-based schemes secure against linear RKAs for all the primitives, indicated by a "[4]+[3]" table entry. Applebaum, Harnik and Ishai [2] present LPN and LWE-based CPA-SE schemes secure against linear RKAs. Wee [28] presents CCA-PKE secure schemes for linear RKAs. Goyal, O'Neill and Rao [22] gave a CPA-SE scheme secure against polynomial RKAs. (We note that their result statement should be amended to exclude constant RKD functions, for no symmetric primitive can be secure under these.) Wee [28] (based on a communication of Wichs) remarks that AMD codes [18] may be used to achieve RKA security for CCA-PKE, a method that extends to other primitives including IBE (but not PRF), but with current constructions of these codes [18], the results continue to be restricted to linear RKAs. We note that we are interested in the stronger, adaptive versions of the definitions as given in [4], but non-adaptively secure signature schemes for non-linear RKAs were provided in [22].

In summary, a basic theoretical question that emerges is how to go beyond linear RKAs. A concrete target here is to bring other primitives to parity with CPA-SE by achieving security for affine and polynomial RKAs. Ideally, we would like approaches that are general, meaning each primitive does not have to be treated separately. As discussed above, we are able to reach these goals with IBE as a starting point.

A CLOSER LOOK. Informally, key-malleability means that user-level private keys obtained by running the IBE scheme's key derivation algorithm $\mathcal{K}$ using a modified master secret key $\phi(s)$ (where $\phi \in \Phi$ and $s \in \mathcal{S}$, the space of master secret keys) can alternatively be computed by running $\mathcal{K}$ using the original master secret key s, followed by a suitable transformation. A collision-resistant identity renaming transform maps identities from the to-be-constructed RKA-secure IBE scheme back into identities in the starting IBE scheme in such a way as to "separate" the sets of identities coming from different values of $\phi(s)$. By modifying the starting IBE scheme to use renamed identities instead of the original ones, we obtain a means to handle otherwise difficult key extraction queries in the RKA setting.

To show that the framework is applicable to the Boneh-Franklin [14] and Waters [27] IBE schemes with $\Phi = \Phi^{\text{aff}}$ (the space of master keys here is $\mathbb{Z}_p$), we exploit specific algebraic properties of the starting IBE schemes. In the Waters case, we obtain an efficient, Φ^{aff}-RKA-secure IBE scheme in the standard model, under the Decisional Bilinear Diffie-Hellman (DBDH) assumption. In the Boneh-Franklin case, we obtain an efficient, Φ^{aff}-RKA-secure IBE scheme under the Bilinear Diffie-Hellman (BDH) assumption with more compact public keys at the expense of working in the Random Oracle Model. Going further, we exhibit a simple modification of the Waters scheme which allows us to handle related key attacks for $\Phi^{\text{poly}(d)}$, this being the set of polynomial functions of bounded degree d. This requires the inclusion of an extra $2d - 2$ elements in the master public key, and a modified, q-type hardness assumption. We show that this assumption holds in the generic group model.

Applying the results of [4] to these IBE schemes, we obtain the first constructions of RKA-secure CCA-PKE and signature schemes for Φ^{aff} and $\Phi^{\text{poly}(d)}$. Again, our schemes are efficient and our results hold in the standard model under reasonable hardness assumptions. The CCA-PKE schemes, being derived via the CHK transform [12], just involve the addition of a one-time signature and verification key to the IBE ciphertexts and so incur little additional overhead for RKA security. As an auxiliary result that improves on the corresponding result of [4], we show that the more efficient MAC-based transform of [15, 12] can be used in place of the CHK transform. The signature schemes arise from the Naor trick, wherein identities are mapped to messages, IBE user private keys are used as signatures, and a trial encryption and decryption on a random plaintext are used to verify the correctness of a

secret key.

signature. This generic construction can often be improved by tweaking the verification procedure, and the same is true here: for example, for the Waters-based signature scheme, we can base security on the CDH assumption instead of DBDH, and can achieve more efficient verification. We stress that our signature schemes are provably unforgeable in a fully adaptive related-key setting, in contrast to the recently proposed signatures in [22].

Note that RKA-secure PRFs for sets Φ^{aff} and $\Phi^{\text{poly}(d)}$ cannot exist, since these sets contain constant functions, and we know that no PRF can be RKA-secure in this case [5]. Thus we are able to show stronger results for IBE, CCA-PKE and Sig than are possible for PRF. Also, although Bellare, Cash and Miller [4] showed that Φ-RKA security for PRF implies Φ-RKA security for Sig and CCA-PKE, the observation just made means we cannot use this result to get Φ^{aff} or $\Phi^{\text{poly}(d)}$ RKA-secure IBE, CCA-PKE or Sig schemes. This provides further motivation for starting from RKA-secure IBE as we do, rather than from RKA-secure PRF.

Finally we note that even for linear RKAs where IBE schemes were known via [4]+[3], our schemes are significantly more efficient.

Further contributions. As a combination of the results of [4] and [26], we provide definitions for RKA security in the joint security setting, where the same key pair is used for both signature and encryption functions, and show that a Φ-RKA-secure IBE scheme can be used to build a Φ-RKA and jointly secure combined signature and encryption scheme. This construction can be instantiated using any of our specific IBE schemes, by which we obtain the first concrete jointly secure combined signature and encryption schemes for the RKA setting.

We also show how to adapt the KEM-DEM (or hybrid encryption) paradigm to the RKA setting, and describe a highly efficient, Φ^{aff}-RKA-secure CCA-KEM that is inspired by our IBE framework and is based on the scheme of Boyen, Mei and Waters [17]. Our CCA-KEM's security rests on the hardness of the DBDH problem for asymmetric pairings $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$; its ciphertexts consist of 2 group elements (one in $\mathbb{G}_1$ and one in $\mathbb{G}_2$), public keys are 3 group elements (two in $\mathbb{G}_2$ and one in $\mathbb{G}_T$), encryption is pairing-free, and the decryption cost is dominated by 3 pairing operations.

The final contribution is an extension of our framework that lets us build an RKA-secure CCA-SE scheme from any IBE scheme satisfying an additional master public key malleability property. Such an IBE scheme, when subjected to our transformation, meets a notion of strong Φ-RKA security [4] where the challenge encryption is also subject to RKA. Applying the CHK transform gives a strong Φ-RKA-secure CCA-PKE scheme which can be converted into a Φ-RKA-secure CCA-SE scheme in the natural way.

Paper organization. Section 2 contains preliminaries, Section 3 describes some IBE schemes and RKA attacks on them, while Section 4 presents our framework for constructing RKA-secure IBE schemes. Section 5 applies the framework to specific schemes, and sketches the CCA-PKE and signature schemes that result from applying the techniques of [4]. Section 7 contains our efficient RKA-secure CCA-KEM. In Section 8, we show how to extend our results to the setting of strong RKA security and how to construct RKA-secure CCA-SE. Appendix B explains how to extend our results to the joint security setting.

2 Preliminaries

Notation. For sets X, Y let $\mathsf{Fun}(X, Y)$ be the set of all functions mapping X to Y. If S is a set then $|S|$ denotes its size and $s \leftarrow\!\!\$\, S$ the operation of picking a random element of S and denoting it by s. Unless otherwise indicated, an algorithm may be randomized. An adversary is an algorithm. By $y \leftarrow\!\!\$\, A(x_1, x_2, \ldots)$ we denote the operation of running A on inputs $x_1, x_2, \ldots$ and letting y denote the outcome. We denote by $[A(x_1, x_2, \ldots, x_n)]$ the set of all possible outputs of A on inputs $x_1, x_2, \ldots, x_n$.

Games. Some of our definitions and proofs are expressed through code-based games [7]. Recall that

such a game consists of an INITIALIZE procedure, procedures to respond to adversary oracle queries, and a FINALIZE procedure. A game G is executed with an adversary A as follows. First, INITIALIZE executes and its output is the input to A. Then A executes, its oracle queries being answered by the corresponding procedures of G. When A terminates, its output becomes the input to the FINALIZE procedure. The output of the latter is called the output of the game. We let G^A denote the event that this game output takes value true. The running time of an adversary, by convention, is the worst case time for the execution of the adversary with any of the games defining its security, so that the time of the called game procedures is included.

RKD FUNCTIONS AND CLASSES. We say that ϕ is a related-key deriving (RKD) function over a set $\mathcal{S}$ if $\phi \in \mathsf{Fun}(\mathcal{S}, \mathcal{S})$. We say that Φ is a class of RKD functions over $\mathcal{S}$ if $\Phi \subseteq \mathsf{Fun}(\mathcal{S}, \mathcal{S})$ and $\mathsf{id} \in \Phi$ where id is the identity function on $\mathcal{S}$. In our constructs, $\mathcal{S}$ will have an algebraic structure, such as being a group, ring or field. In the last case, for $a, b \in \mathcal{S}$ we define $\phi_b^+, \phi_a^*, \phi_{a,b}^{\mathrm{aff}} \in \mathsf{Fun}(\mathcal{S}, \mathcal{S})$ via $\phi_b^+(s) = s + b$, $\phi_a^*(s) = as$, and $\phi_{a,b}^{\mathrm{aff}}(s) = as + b$ for all $s \in \mathcal{S}$. For a polynomial q over field $\mathcal{S}$, we define $\phi_q^{\mathrm{poly}}(s) = q(s)$ for all $s \in \mathcal{S}$. We let $\Phi^+ = \{\, \phi_b^+ \,:\, b \in \mathcal{S} \,\}$ be the class of additive RKD functions, $\Phi^* = \{\, \phi_a^* \,:\, a \in \mathcal{S} \,\}$ be the class of multiplicative RKD functions, $\Phi^{\mathrm{aff}} = \{\, \phi_{a,b}^{\mathrm{aff}} \,:\, a, b \in \mathcal{S} \,\}$ the class of affine RKD functions, and for any fixed positive integer d, we let $\Phi^{\mathrm{poly}(d)} = \{\, \phi_q^{\mathrm{poly}} \,:\, \deg q \leq d \,\}$ be the set of polynomial RKD functions of bounded degree d.

If $\phi \neq \phi'$ are distinct functions in a class Φ there is of course by definition an s such that $\phi(s) \neq \phi'(s)$, but there could also be keys s on which $\phi(s) = \phi'(s)$. We say that a class Φ is claw-free if the latter does not happen, meaning for all distinct $\phi \neq \phi'$ in Φ we have $\phi(s) \neq \phi'(s)$ for *all* $s \in \mathcal{S}$. With the exception of [22], all previous constructions of Φ-RKA-secure primitives with proofs of security have been for claw-free classes [5, 25, 21, 3, 4, 28]. In particular, key fingerprints are defined in [3] in such a way that their assumption of a Φ-key fingerprint automatically implies that Φ is claw-free.

IBE SYNTAX. We specify an IBE scheme $\mathcal{IBE} = (\mathcal{S}, \mathcal{P}, \mathcal{K}, \mathcal{E}, \mathcal{D})$ by first specifying a non-empty set $\mathcal{S}$ called the *master-key space* from which the master secret key s is drawn at random. The master public key $\pi \leftarrow \mathcal{P}(s)$ is then produced by applying to s a deterministic master public key generation algorithm $\mathcal{P}$. A decryption key for an identity u is produced via $dk_u \leftarrow\!\!\$\, \mathcal{K}(s, u)$. A ciphertext C encrypting a message M for u is generated via $C \leftarrow\!\!\$\, \mathcal{E}(\pi, u, M)$. A ciphertext C is deterministically decrypted via $M \leftarrow \mathcal{D}(dk, C)$. Correctness requires that $\mathcal{D}(\mathcal{K}(s, u), \mathcal{E}(\pi, u, M)) = M$ with probability one for all $M \in \mathsf{MSp}$ and all $u \in \mathsf{USp}$ where $\mathsf{MSp}, \mathsf{USp}$ are, respectively, the message and identity spaces associated to $\mathcal{IBE}$.

The usual IBE syntax specifies a single parameter generation algorithm that produces s, π together, and although there is of course a space from which the master secret key is drawn, it is not explicitly named. But RKD functions will have domain the space of master keys of the IBE scheme, which is why it is convenient in our context to make it explicit in the syntax. Saying the master public key is a deterministic function of the master secret key is not strictly necessary for us, but it helps make some things a little simpler and is true in all known schemes, so we assume it.

We make an important distinction between parameters and the master public key, namely that the former may not depend on s while the latter might. Parameters will be groups, group generators, pairings and the like. They will be fixed and available to all algorithms without being named as explicit inputs.

RKA-SECURE IBE. We define Φ-RKA security of IBE schemes following [4]. Game IBE of Figure 2 is associated to $\mathcal{IBE} = (\mathcal{S}, \mathcal{P}, \mathcal{K}, \mathcal{E}, \mathcal{D})$ and a class Φ of RKD functions over $\mathcal{S}$. An adversary is allowed only one query to LR. Let $\mathbf{Adv}^{\mathrm{ibe-rka}}_{\mathcal{IBE},\Phi}(A)$ equal $2\Pr[\mathrm{IBE}^A] - 1$. A feature of the definition we draw attention to is that the key derivation oracle KD refuses to act only when the identity it is given matches the challenge one *and* the derived key equals the real one. This not only creates a strong security requirement but one that is challenging to achieve because a simulator, not knowing s, cannot check whether or not the IBE adversary succeeded. This difficulty is easily resolved if Φ is claw-free

proc INITIALIZE // IBE
$s \leftarrow\!\!\$\ \mathcal{S}; \pi \leftarrow \mathcal{P}(s)$
$b \leftarrow\!\!\$\ \{0,1\}$
$u^* \leftarrow \perp\ ;\ I \leftarrow \emptyset$
Return π

proc FINALIZE(b') // IBE
Return $(b = b')$

proc KD(ϕ, u) // IBE
$s' \leftarrow \phi(s)$
If $(s' = s)$ $I \leftarrow I \cup \{u\}$
If $(u^* \in I)$ Return $\perp$
Return $dk \leftarrow\!\!\$\ \mathcal{K}(s', u)$

proc LR(u, M_0, M_1) // IBE
If $(|M_0| \neq |M_1|)$ Return $\perp$
$u^* \leftarrow u$
If $(u^* \in I)$ Return $\perp$
Return $C \leftarrow\!\!\$\ \mathcal{E}(\pi, u^*, M_b)$

Figure 2: Game IBE defining Φ-RKA-security of IBE scheme $\mathcal{IBE} = (\mathcal{S}, \mathcal{P}, \mathcal{K}, \mathcal{E}, \mathcal{D})$.

Algorithm $\mathcal{P}(s)$:
$\pi \leftarrow g^s$
Return π

Algorithm $\mathcal{K}(s, u)$:
$dk \leftarrow H_1(u)^s$
Return dk

Algorithm $\mathcal{E}(\pi, u, M)$:
$t \leftarrow\!\!\$\ \mathbb{Z}_p$
$C_1 \leftarrow g^t$
$C_2 \leftarrow H_2(e(\pi, H_1(u))^t) \oplus M$
Return (C_1, C_2)

Algorithm $\mathcal{D}(dk, C)$:
$M \leftarrow C_2 \oplus H_2(e(dk, C_1))$
Return M

Algorithm $\mathcal{P}(s)$:
$\pi \leftarrow g^s$
Return π

Algorithm $\mathcal{K}(s, u)$:
$r \leftarrow\!\!\$\ \mathbb{Z}_p$
$dk_1 \leftarrow g_1^s \cdot H(u)^r$
$dk_2 \leftarrow g^r$
Return (dk_1, dk_2)

Algorithm $\mathcal{E}(\pi, u, M)$:
$t \leftarrow\!\!\$\ \mathbb{Z}_p$
$C_1 \leftarrow g^t$
$C_2 \leftarrow H(u)^t$
$C_3 \leftarrow e(\pi, g_1)^t \cdot M$
Return (C_1, C_2, C_3)

Algorithm $\mathcal{D}(dk, C)$:
$M \leftarrow C_3 \cdot \frac{e(dk_2, C_2)}{e(dk_1, C_1)}$
Return M

Figure 3: Boneh-Franklin IBE scheme on the left, Waters IBE scheme on the right.

but not otherwise. We consider this particular RKA security definition as, in addition to its strength, it is the level of RKA security required of an IBE scheme so that application of the CHK and Naor transforms results in RKA-secure CCA-PKE and signature schemes.

3 Existing IBE schemes and RKA attacks on them

The algorithms of the Boneh-Franklin BasicIdent IBE scheme [14] are given in Figure 3. The parameters of the scheme are groups $\mathbb{G}_1, \mathbb{G}_T$ of prime order p, a symmetric pairing $e : \mathbb{G}_1 \times \mathbb{G}_1 \to \mathbb{G}_T$, a generator g of $\mathbb{G}_1$ and hash functions $H_1 : \{0,1\}^* \to \mathbb{G}_1$, $H_2 : \mathbb{G}_T \to \{0,1\}^n$ which are modeled as random oracles in the security analysis. Formally, these are output by a pairing parameter generator on input 1^k. This scheme is IND-CPA secure in the usual model for IBE security, under the Bilinear Diffie-Hellman (BDH) assumption.

The algorithms of the Waters IBE scheme [27] are also given in Figure 3. The parameters of the scheme are groups $\mathbb{G}_1, \mathbb{G}_T$ of prime order p, a symmetric pairing $e : \mathbb{G}_1 \times \mathbb{G}_1 \to \mathbb{G}_T$, generators g, g_1 of $\mathbb{G}_1$ and group elements $h_0, \ldots, h_n \in \mathbb{G}_1$ specifying the hash function $H(u) = h_0 \prod_{i \in u} h_i$. The Waters IBE scheme is also IND-CPA secure in the usual model for IBE security, under the DBDH assumption.

The Waters IBE scheme is not RKA secure if Φ includes a function $\phi_a^*(s) = as$. A call to the key derivation oracle with any such ϕ yields a user secret key $(dk_1, dk_2) = (g_1^{as} \cdot H(u)^r, g^r)$. Raising this to a^{-1} gives $(dk_1', dk_2') = (g_1^s \cdot H(u)^{ra^{-1}}, g^{ra^{-1}})$, so that (dk_1', dk_2') is a user secret key for identity u under the original master secret key with randomness $r' = ra^{-1}$. An RKA adversary can thus obtain the user secret key for any identity of his choosing and hence break the RKA security of the Waters scheme. A similar attack applies to the Boneh-Franklin scheme.

4 Framework for deriving RKA-secure IBE schemes

In the previous section we saw that the Boneh-Franklin and Waters schemes are not RKA secure. Here we will show how to modify these and other schemes to be RKA secure by taking advantage, in part, of the very algebra that leads to the attacks. We describe a general framework for creating RKA-secure IBE schemes and then apply it obtain several such schemes.

We target a very particular type of framework, one that allows us to reduce RKA security of a modified IBE scheme directly to the normal IBE security of a base IBE scheme. This will allow us to exploit known results on IBE in a blackbox way and avoid re-entering the often complex security proofs of the base IBE schemes.

KEY-MALLEABILITY. We say that an IBE scheme $\mathit{IBE} = (\mathcal{S}, \mathcal{P}, \mathcal{K}, \mathcal{E}, \mathcal{D})$ is Φ-key-malleable if there is an algorithm T, called the key simulator, which, given π, an identity u, a decryption key $dk' \leftarrow\!\!\$\, \mathcal{K}(s, u)$ for u under s and an RKD function $\phi \in \Phi$, outputs a decryption key dk for u under master secret key $\phi(s)$ that is distributed identically to the output of $\mathcal{K}(\phi(s), u)$. The formalization takes a little more care for in talking about two objects being identically distributed one needs to be precise about relative to what other known information this is true. A simple and rigorous definition here can be made using games. We ask that

$$\Pr[\mathrm{KMReal}^{M}_{\mathit{IBE},\Phi}] \;=\; \Pr[\mathrm{KMSim}^{M}_{\mathit{IBE},\Phi,T}]$$

for all (not necessarily computationally bounded) adversaries M, where the games are as follows. The INITIALIZE procedure of both picks s at random from $\mathcal{S}$ and returns $\pi \leftarrow \mathcal{P}(s)$ to the adversary. In game $\mathrm{KMReal}_{\mathit{IBE},\Phi}$, oracle $\mathrm{KD}(\phi, u)$ returns $dk \leftarrow\!\!\$\, \mathcal{K}(\phi(s), u)$ but in game $\mathrm{KMSim}_{\mathit{IBE},\Phi,T}$ it lets $dk' \leftarrow\!\!\$\, \mathcal{K}(s, u)$ and returns $T(\pi, u, dk', \phi)$. There are no other oracles, and $\textsc{Finalize}(b')$ returns $(b' = 1)$.

USING KM. Intuitively, key-malleability allows us to simulate a Φ-RKA adversary via a normal adversary and would thus seem to be enough to prove Φ-RKA security of IBE based on its normal security. Let us see how this argument goes and then see the catches that motivate a transformation of the scheme via collision-resistant identity renaming. Letting $\overline{A}$ be an adversary attacking the Φ-RKA security of IBE, we aim to build an adversary A such that

$$\mathbf{Adv}^{\mathrm{ibe-rka}}_{\mathit{IBE},\Phi}(\overline{A}) \le \mathbf{Adv}^{\mathrm{ibe}}_{\mathit{IBE}}(A)\ . \tag{1}$$

On input π, adversary A runs $\overline{A}(\pi)$. When the latter makes a $\mathrm{KD}(\phi, u)$ query, A lets $dk \leftarrow \mathrm{KD}(\mathsf{id}, u)$, where KD is A's own key derivation oracle. It then lets $\overline{dk} \leftarrow T(\pi, u, dk, \phi)$ and returns $\overline{dk}$ to $\overline{A}$. Key-malleability tells us that $\overline{dk}$ is distributed identically to an output of $\mathrm{KD}(\phi, u)$, so the response provided by A is perfectly correct. When $\overline{A}$ makes a $\mathrm{LR}(u, M_0, M_1)$ query, A lets $C \leftarrow \mathrm{LR}(u, M_0, M_1)$ and returns C to A. Finally when $\overline{A}$ halts with output a bit b', adversary A does the same.

The simulation seems perfect, so we appear to have established Equation (1). What's the catch? The problem is avoiding *challenge key derivation*. Suppose $\overline{A}$ made a $\mathrm{KD}(\phi, u)$ query for a ϕ such that $\phi(s) \neq s$; then made a $\mathrm{LR}(u, M_0, M_1)$ query; and finally, given C, correctly computed b. It would win its game, because the condition $\phi(s) \neq s$ means that identity u may legitimately be used both in a key derivation query and in the challenge LR query. But our constructed adversary A, in the simulation, would make query $\mathrm{KD}(\mathsf{id}, u)$ to answer $\overline{A}$'s $\mathrm{KD}(\phi, u)$ query, and then make query $\mathrm{LR}(u, M_0, M_1)$. A would thus have queried the challenge identity u to the key-extraction oracle and would not win.

This issue is dealt with by transforming the base scheme via what we call identity renaming, so that Φ-RKA security of the transformed scheme can be proved based on the Φ-key-malleability of the base scheme.

IDENTITY RENAMING. Renaming is a way to map identities in the new scheme back to identities of the given, base scheme. Let us now say how renaming works more precisely and then define the modified scheme.

Let $\mathit{IBE} = (\mathcal{S}, \mathcal{P}, \mathcal{K}, \mathcal{E}, \mathcal{D})$ denote the given, base IBE scheme, and let USp be its identity space. A renaming scheme is a pair (SI, PI) of functions where SI: $\mathcal{S} \times \overline{\mathsf{USp}} \to \mathsf{USp}$ and PI: $[\mathcal{P}(\mathcal{S})] \times \overline{\mathsf{USp}} \times \Phi \to \mathsf{USp}$ where $\overline{\mathsf{USp}}$, implicitly specified by the renaming scheme, will be the identity space of the new scheme we will soon define. The first function SI, called the secret renaming function, uses the master secret key, while its counterpart public renaming function PI uses the master public key. We require that $\mathrm{SI}(\phi(s), \overline{u}) = \mathrm{PI}(\pi, \overline{u}, \phi)$ for all $s \in \mathcal{S}$, all $\pi \in [\mathcal{P}(s)]$, all $\overline{u} \in \overline{\mathsf{USp}}$ and all $\phi \in \Phi$. This *compatibility condition* says that the two functions arrive, in different ways, at the same outcome.

The transform. The above is all we need to specify our Identity Renaming Transform **IRT** that maps a base IBE scheme $\mathit{IBE} = (\mathcal{S}, \mathcal{P}, \mathcal{K}, \mathcal{E}, \mathcal{D})$ to a new IBE scheme $\overline{\mathit{IBE}} = (\mathcal{S}, \mathcal{P}, \overline{\mathcal{K}}, \overline{\mathcal{E}}, \mathcal{D})$. As the notation indicates, the master key space, master public key generation algorithm and decryption algorithm are unchanged. The other algorithms are defined by

$$\overline{\mathcal{K}}(s, \overline{u}) = \mathcal{K}(s, \mathrm{SI}(s, \overline{u})) \qquad \text{and} \qquad \overline{\mathcal{E}}(\pi, \overline{u}, M) = \mathcal{E}(\pi, \mathrm{PI}(\pi, \overline{u}, \mathsf{id}), M) \ .$$

We clarify that algorithms of the new IBE scheme do not, and cannot, have as input the RKD functions ϕ used by the attacker. We are defining an IBE scheme, and algorithm inputs must follow the syntax of IBE schemes. When the new encryption algorithm invokes PI, it sets ϕ to the identity function id. (Looking ahead, the simulation will call the renaming functions with ϕ emanating from the adversary attacking the new IBE scheme.) The key derivation algorithm has s but not π (recall we cannot give it π because otherwise it becomes subject to the RKA) and thus uses the secret renaming function. On the other hand the encryption algorithm has π but obviously not s and thus uses the public renaming function. This explains why we need two, compatible renaming functions. The new scheme has the same message space as the old one. Its identity space is inherited from the renaming scheme, being the space $\overline{\mathsf{USp}}$ from which the renaming functions draw their identity inputs.

The above compatibility requirement implies that $\mathrm{SI}(s, \overline{u}) = \mathrm{PI}(\pi, \overline{u}, \mathsf{id})$. From this it follows that $\overline{\mathit{IBE}}$ preserves the correctness of IBE. We now go on to specifying properties of the base IBE scheme and the renaming functions that suffice to prove Φ-RKA security of the new scheme.

A trivial renaming scheme is obtained by setting $\mathrm{SI}(s, \overline{u}) = \overline{u} = \mathrm{PI}(\pi, \overline{u}, \phi)$. This satisfies the compatibility condition. However, the transformed IBE scheme $\overline{\mathit{IBE}}$ ends up identical to the base IBE and thus this trivial renaming cannot aid in getting security. We now turn to putting a non-trivial condition on the renaming scheme that we will show suffices.

Collision-resistance. The renaming scheme (SI, PI) will be required to have a collision-resistance property. In its simplest and strongest form the requirement is that

$$(\phi(s), \overline{u}_1) \neq (s, \overline{u}_2) \qquad \Rightarrow \qquad \mathrm{SI}(\phi(s), \overline{u}_1) \neq \mathrm{SI}(s, \overline{u}_2)$$

for all $s \in \mathcal{S}$, all $\overline{u}_1, \overline{u}_2 \in \overline{\mathsf{USp}}$ and all $\phi \in \Phi$. This *statistical collision-resistance* will be enough to prove that $\overline{\mathit{IBE}}$ is Φ-RKA secure if IBE is Φ-key-malleable (cf. Theorem 4.1). We will now see how this goes. Then we will instantiate these ideas to get concrete Φ-RKA-secure schemes for many interesting classes Φ including Φ^{aff} and $\Phi^{\mathrm{poly}(d)}$.

Theorem 4.1 *Let $\mathit{IBE} = (\mathcal{S}, \mathcal{P}, \mathcal{K}, \mathcal{E}, \mathcal{D})$ be a Φ-key-malleable IBE scheme with key simulator T. Let $\overline{\mathit{IBE}} = (\mathcal{S}, \mathcal{P}, \overline{\mathcal{K}}, \overline{\mathcal{E}}, \mathcal{D})$ be obtained from IBE and renaming scheme (SI, PI) via the transform* **IRT** *described above. Assume the renaming scheme is statistically collision-resistant. Let $\overline{A}$ be a Φ-RKA adversary against $\overline{\mathit{IBE}}$ that makes q key derivation queries. Then there is an adversary A making q key derivation queries such that*

$$\mathbf{Adv}^{\mathrm{ibe-rka}}_{\overline{\mathit{IBE}}, \Phi}(\overline{A}) \leq \mathbf{Adv}^{\mathrm{ibe}}_{\mathit{IBE}}(A) \ . \tag{2}$$

Furthermore, the running time of A is that of $\overline{A}$ plus the time for q executions of T and $q+1$ executions of PI.

Proof of Theorem 4.1: Consider the games of Figure 4. Game G_0 is written to be equivalent to

proc INITIALIZE $/\!/ \mathrm{G}_0$
000 $s \leftarrow\!\!\$\ \mathcal{S}$; $\pi \leftarrow \mathcal{P}(s)$
001 $b \leftarrow\!\!\$\ \{0,1\}$; $\overline{u}^* \leftarrow \bot$
002 $\overline{I} \leftarrow \emptyset$
003 Return π

proc INITIALIZE $/\!/ \mathrm{G}_1, \mathrm{G}_2, \mathrm{G}_3$
100 $s \leftarrow\!\!\$\ \mathcal{S}$; $\pi \leftarrow \mathcal{P}(s)$
101 $b \leftarrow\!\!\$\ \{0,1\}$; $u^* \leftarrow \bot$
102 $I \leftarrow \emptyset$
103 Return π

proc KD$(\phi, \overline{u})$ $/\!/ \mathrm{G}_0$
010 $s' \leftarrow \phi(s)$
011 If $(s' = s)$ $\overline{I} \leftarrow \overline{I} \cup \{\overline{u}\}$
012 If $(\overline{u}^* \in \overline{I})$ Return $\bot$
013 $u \leftarrow \mathrm{SI}(s', \overline{u})$
014 Return $\overline{dk} \leftarrow\!\!\$\ \mathcal{K}(s', u)$

proc KD$(\phi, \overline{u})$ $/\!/ \mathrm{G}_1$
110 $s' \leftarrow \phi(s)$
111 $u \leftarrow \mathrm{SI}(s', \overline{u})$
112 $I \leftarrow I \cup \{u\}$
113 If $(u^* \in I)$ Return $\bot$
114 Return $\overline{dk} \leftarrow\!\!\$\ \mathcal{K}(s', u)$

proc KD$(\phi, \overline{u})$ $/\!/ \mathrm{G}_2$
210 $u \leftarrow \mathrm{PI}(\pi, \overline{u}, \phi)$
211 $I \leftarrow I \cup \{u\}$
212 If $(u^* \in I)$ Return $\bot$
213 Return $\overline{dk} \leftarrow\!\!\$\ \mathcal{K}(\phi(s), u)$

proc KD$(\phi, \overline{u})$ $/\!/ \mathrm{G}_3$
310 $u \leftarrow \mathrm{PI}(\pi, \overline{u}, \phi)$
311 $I \leftarrow I \cup \{u\}$
312 If $(u^* \in I)$ Return $\bot$
313 $dk \leftarrow\!\!\$\ \mathcal{K}(s, u)$
314 Return $\overline{dk} \leftarrow T(\pi, u, dk, \phi)$

proc LR$(\overline{u}, M_0, M_1)$ $/\!/ \mathrm{G}_0$
020 If $(|M_0| \neq |M_1|)$ Return $\bot$
021 $\overline{u}^* \leftarrow \overline{u}$
022 If $(\overline{u}^* \in \overline{I})$ Return $\bot$
023 $u^* \leftarrow \mathrm{SI}(s, \overline{u}^*)$
024 Return $C \leftarrow\!\!\$\ \mathcal{E}(\pi, u^*, M_b)$

proc LR$(\overline{u}, M_0, M_1)$ $/\!/ \mathrm{G}_1$
120 If $(|M_0| \neq |M_1|)$ Return $\bot$
121 $u^* \leftarrow \mathrm{SI}(s, \overline{u})$
122 If $(u^* \in I)$ Return $\bot$
123 Return $C \leftarrow\!\!\$\ \mathcal{E}(\pi, u^*, M_b)$

proc LR$(\overline{u}, M_0, M_1)$ $/\!/ \mathrm{G}_2, \mathrm{G}_3$
220 If $(|M_0| \neq |M_1|)$ Return $\bot$
221 $u^* \leftarrow \mathrm{PI}(\pi, \overline{u}, \mathsf{id})$
222 If $(u^* \in I)$ Return $\bot$
223 Return $C \leftarrow\!\!\$\ \mathcal{E}(\pi, u^*, M_b)$

proc FINALIZE(b') $/\!/$ All
030 Return $(b = b')$

Figure 4: Games for proof of Theorem 4.1.

game $\mathrm{IBE}_{\overline{\mathcal{IBE}}}$, so that

$$\mathbf{Adv}^{\mathrm{ibe-rka}}_{\overline{\mathcal{IBE}},\Phi}(\overline{A}) \;=\; 2\Pr[\mathrm{G}_0^{\overline{A}}] - 1\ . \tag{3}$$

In answering a KD$(\phi, \overline{u})$ query, G_0 must use the key-generation algorithm $\overline{\mathcal{K}}$ of the new scheme $\overline{\mathcal{IBE}}$ but with master secret key $s' = \phi(s)$. From the definition of $\overline{\mathcal{K}}$, it follows that not only is the key-generation at line 014 done under s', but also the identity renaming at line 013. LR, correspondingly, should use $\overline{\mathcal{E}}$, and thus the public renaming function PI. The compatibility property however allows us at line 023 to use SI instead. This will be useful in exploiting statistical collision-resistance in the next step, after which we will revert back to PI.

The adversary A we aim to construct will not know s. A central difficulty in the simulation is thus lines 011, 012 of G_0 where the response provided to $\overline{A}$ depends on the result of a test involving s, a test that A cannot perform. Before we can design A we must get rid of this test. Statistical collision-resistance is what will allow us to do so. KD of game G_1 moves the identity renaming up before the list of queried identities is updated to line 111 and then, at line 112, adds the transformed identity to the list. LR is likewise modified so its test now involves the transformed (rather than original) identities. We claim this makes no difference, meaning

$$\Pr[\mathrm{G}_0^{\overline{A}}] \;=\; \Pr[\mathrm{G}_1^{\overline{A}}]\ . \tag{4}$$

Indeed, statistical collision-resistance tell us that $(s', \overline{u}) = (s, \overline{u}^*)$ iff $\mathrm{SI}(s', \overline{u}) = \mathrm{SI}(s, \overline{u}^*)$. This means that lines 011, 012 and lines 112, 113 are equivalent.

Compatibility is invoked to use PI in place of SI in both KD and in LR in G_2, so that

$$\Pr[\mathrm{G}_1^{\overline{A}}] \;=\; \Pr[\mathrm{G}_2^{\overline{A}}]\ . \tag{5}$$

Rather than use s' for key generation as at 213, G_3 uses s at 313 and then applies the key simulator T.

We claim the key-malleability implies

$$\Pr[\mathrm{G}_2^{\overline{A}}] \;=\; \Pr[\mathrm{G}_3^{\overline{A}}]\;. \tag{6}$$

To justify this we show that there is an adversary M such that

$$\Pr[\mathrm{KMReal}^{M}_{\mathcal{IBE},\Phi}] \;=\; \Pr[\mathrm{G}_2^{\overline{A}}] \qquad \text{and} \qquad \Pr[\mathrm{KMSim}^{M}_{\mathcal{IBE},\Phi,T}] \;=\; \Pr[\mathrm{G}_3^{\overline{A}}]\;.$$

Adversary M, on input π, begins with the initializations $u^* \leftarrow \bot$; $I \leftarrow \emptyset$; $b \leftarrow\!\!\$\ \{0,1\}$ and then runs $\overline{A}$ on input π. When $\overline{A}$ makes a $\mathrm{KD}(\phi, \overline{u})$ query, M does the following:

$$u \leftarrow \mathrm{PI}(\pi, \overline{u}, \phi)\ ;\ I \leftarrow I \cup \{u\}\ ;\ \text{If } (u^* \in I) \text{ Return } \bot\ ;\ \overline{dk} \leftarrow \mathrm{KD}(\phi, u).$$

If M is playing game KMReal then its KD oracle will behave as line 213 in game G_2, while if M is playing game KMSim its KD oracle will behave as lines 313,314 in game G_3. When $\overline{A}$ makes its $\mathrm{LR}(\overline{u}, M_0, M_1)$ query M sets $u^* \leftarrow \mathrm{PI}(\pi, \overline{u}, \mathsf{id})$ and checks if $u^* \in I$, returning $\bot$ if so. M then computes $C \leftarrow\!\!\$\ \mathcal{E}(\pi, u^*, M_b)$ which it returns to $\overline{A}$. When $\overline{A}$ halts with output b', M returns the result of $(b' = b)$. If M is playing game KMReal then game G_2 is perfectly simulated, while if M is playing KMSim then game G_3 is perfectly simulated, so M returns 1 with the same probability that $\overline{A}$ wins in each case and by the key-malleability of $\mathcal{IBE}$ Equation (6) holds.

Finally, we design A so that

$$\mathbf{Adv}^{\mathrm{ibe}}_{\mathcal{IBE}}(A) \;=\; 2\Pr[\mathrm{G}_3^{\overline{A}}] - 1\;. \tag{7}$$

On input π, adversary A runs $\overline{A}(\pi)$. When the latter makes a $\mathrm{KD}(\phi, \overline{u})$ query, A does the following:

$$u \leftarrow \mathrm{PI}(\pi, \overline{u}, \phi)\ ;\ dk \leftarrow \mathrm{KD}(\mathsf{id}, u)\ ;\ \overline{dk} \leftarrow T(\pi, u, dk, \phi).$$

It then returns $\overline{dk}$ to $\overline{A}$. The KD invoked in this code is A's own oracle. Compatibility tells us that $u = \mathrm{SI}(\phi(s), \overline{u})$ and thus from the definition of $\overline{\mathcal{IBE}}$, the response to $\overline{A}$'s query is distributed according to $\mathcal{K}(\phi(s), u)$. But key-malleability then tells us that $\overline{dk}$ is distributed identically to this, so the response provided by A is perfectly correct. When $\overline{A}$ makes a $\mathrm{LR}(\overline{u}, M_0, M_1)$ query, A does the following:

$$u \leftarrow \mathrm{PI}(\pi, \overline{u}, \mathsf{id})\ ;\ C \leftarrow \mathrm{LR}(u, M_0, M_1).$$

It then returns C to $\overline{A}$. The LR invoked in this code is A's own oracle. The definition of $\overline{\mathcal{IBE}}$ implies that the response provided by A is again perfectly correct. Finally when $\overline{A}$ halts with output a bit b', adversary A does the same. ∎

5 Applying the framework

Affine RKD functions for Boneh-Franklin and Waters. We show how the framework can be instantiated with the IBE schemes of Boneh-Franklin and Waters to achieve IBE schemes secure against affine related-key attacks. First we look at key-malleability. Keys in the Boneh-Franklin IBE scheme are of the form $dk' = H_1(u)^s$, so the algorithm T is as follows:

$$T(\pi, u, dk', \phi_{a,b})\text{: } dk \leftarrow dk'^{a} \cdot H_1(u)^b\text{; Return } dk$$

The output of T is a valid key for user u under master secret key $\phi_{a,b}(s)$, since:

$$dk'^{a} \cdot H_1(u)^b = H_1(u)^{sa} \cdot H_1(u)^b = H_1(u)^{as+b}\;.$$

Since the key derivation algorithm is deterministic, the keys output by T are distributed identically to the keys output by $\mathcal{K}(\phi(s), u)$, and so the Boneh-Franklin IBE scheme is key-malleable.

Keys in the Waters IBE scheme are of the form $(dk'_1, dk'_2) = (g_1^s \cdot H(u)^r, g^r)$ for some r in $\mathbb{Z}_p$, so the algorithm T is as follows:

$T(\pi, u, dk', \phi_{a,b})$:
If $(a = 0)$ then $r \leftarrow\!\!\$\ \mathbb{Z}_p$; $dk_1 \leftarrow g_1^b \cdot H(u)^r$; $dk_2 \leftarrow g^r$
Else $dk_1 \leftarrow dk_1'^a \cdot g_1^b$; $dk_2 \leftarrow dk_2'^a$
Return (dk_1, dk_2)

When the RKD function is a constant function, T behaves exactly as the key derivation algorithm under master secret key b, so its output is valid and correctly distributed. Otherwise, the output of T is still a valid key for user u under master secret key $\phi_{a,b}(s)$, now under randomness ra, since:

$$dk_1'^a \cdot g_1^b = (g_1^s \cdot H(u)^r)^a \cdot g_1^b = g_1^{as+b} H(u)^{ra} \qquad dk_2'^a = g^{ra} \ .$$

Since r is uniformly distributed in $\mathbb{Z}_p$, ra is also uniformly distributed in $\mathbb{Z}_p$ and so the keys output by T are distributed identically to those output by $\mathcal{K}(\phi(s), u)$. Hence the Waters IBE scheme is key-malleable.

The same identity renaming scheme can be used for both IBE schemes. Namely, $\mathrm{SI}(s, \overline{u})$ returns $\overline{u}||g^s$ and $\mathrm{PI}(\pi, \overline{u}, \phi_{a,b})$ returns $\overline{u}||\pi^a \cdot g^b$. The compatibility requirement is satisfied and the renaming scheme is clearly collision-resistant since $\overline{u}_1||g^{\phi(s)} = \overline{u}_2||g^s \Rightarrow \overline{u}_1 = \overline{u}_2 \wedge \phi(s) = s$. Thus the IBE schemes of Boneh-Franklin and Waters are key-malleable and admit a suitable identity renaming scheme, and so satisfy the requirements of Theorem 4.1. For concreteness, the algorithms of the transformed schemes are given in Appendix A. Notice that in the Waters case, we must increase the parameter n by the bit length of elements of $\mathbb{G}_1$ (and hence increase the size of the description of the scheme parameters) to allow identities of the form $\overline{u}||g^s$ to be used in the renaming scheme.

The following theorem is obtained by combining Theorem 4.1 with [14], and the running time of B below may be obtained in the same way.

Theorem 5.1 *Let $\overline{\mathit{IBE}} = (\mathcal{S}, \mathcal{P}, \overline{\mathcal{K}}, \overline{\mathcal{E}}, \mathcal{D})$ be the Boneh-Franklin IBE scheme shown in Figure 3 under the above identity renaming transform. Let $\overline{A}$ be a Φ^{aff}-RKA adversary against $\overline{\mathit{IBE}}$ making q_{KD} key derivation queries and q_{H_2} queries to random oracle H_2. Then there is an algorithm B solving the Decision Bilinear Diffie-Hellman problem such that*

$$\mathbf{Adv}^{\mathrm{ibe-rka}}_{\overline{\mathit{IBE}},\Phi^{\mathrm{aff}}}(\overline{A}) \leq \frac{e(1 + q_{KD})q_{H_2}}{2} \cdot \mathbf{Adv}^{\mathrm{dbdh}}(B) \ . \tag{8}$$

The following theorem is obtained by combining Theorem 4.1 with [27], and the running time of B below may be obtained in the same way. Concrete-security improvements would be obtained by using instead the analysis of Waters' scheme from [6].

Theorem 5.2 *Let $\overline{\mathit{IBE}} = (\mathcal{S}, \mathcal{P}, \overline{\mathcal{K}}, \overline{\mathcal{E}}, \mathcal{D})$ be the Waters scheme shown in Figure 3 under the above identity renaming transform. Let $\overline{A}$ be a Φ^{aff}-RKA adversary against $\overline{\mathit{IBE}}$ making q_{KD} key derivation queries. Then there is an algorithm B solving the Decision Bilinear Diffie-Hellman problem such that*

$$\mathbf{Adv}^{\mathrm{ibe-rka}}_{\overline{\mathit{IBE}},\Phi^{\mathrm{aff}}}(\overline{A}) \leq 32(n + 1) \cdot q_{KD} \cdot \mathbf{Adv}^{\mathrm{dbdh}}(B) \ . \tag{9}$$

We recall from [4] that, given a Φ-RKA-secure IBE scheme, the CHK transform [12] yields a Φ-RKA-secure CCA-PKE scheme at the cost of adding a strongly unforgeable one-time secure signature and its verification key to the IBE ciphertexts. In Appendix E we show that the more efficient Boneh-Katz transform [12] can also be used to the same effect. We omit the details of the Φ^{aff}-RKA-secure CCA-PKE schemes that result from applying these transforms to the above IBE schemes. We simply note that the resulting CCA-PKE schemes are as efficient as the pairing-based schemes of Wee [28], which are only Φ^{lin}-RKA-secure. Similarly, using a result of [4], we may apply the Naor transform to these IBE schemes to obtain Φ^{aff}-RKA-secure signature schemes that are closely related to (and as efficient as) the Boneh-Lynn-Shacham [16] and Waters [27] signature schemes. The verification algorithms of these signature schemes can be improved by replacing Naor's trial encryption and decryption procedure by bespoke algorithms, exactly as in [16, 27].

An IBE scheme handling RKAs for bounded degree polynomials. We show how to construct an IBE scheme that is RKA secure when the RKD function set equals $\Phi^{\mathrm{poly}(d)}$, the set of all polynomials

proc INITIALIZE
$g \leftarrow\$ \mathbb{G}_1$; $x, y, z \leftarrow\$ \mathbb{Z}_p$; $b \leftarrow\$ \{0,1\}$
If $(b = 1)$ $T \leftarrow e(g,g)^{xyz}$
Else $T \leftarrow\$ \mathbb{G}_T$
Return $g, g^x, g^{x^2}, \ldots, g^{x^q}, g^y, g^{(x^2)y}, g^{(x^3)y}, \ldots, g^{(x^q)y}, g^z, T$

proc FINALIZE(b')
Return $(b = b')$

Figure 5: q-Extended Decision Bilinear Diffie-Hellman (q-EDBDH) game.

of degree at most d, for an arbitrary d chosen at the time of master key generation. The scheme is obtained through a simple extension of the IBE scheme of Waters combined with the identity renaming transform used above. The only change we make to the Waters scheme is in the master public key, where we add the extra elements $g^{s^2}, \ldots, g^{s^d}, {g_1}^{s^2}, \ldots, {g_1}^{s^d}$ alongside g^s. These elements assist in achieving key-malleability for the set $\Phi^{\mathrm{poly}(d)}$. The master public-key generation algorithm of the extended Waters scheme is then

$$\mathcal{P}(s) : \pi_0 \leftarrow g^s \; ; \; \pi \leftarrow (\pi_0, g^{s^2}, \ldots, g^{s^d}, (g_1)^{s^2}, \ldots, (g_1)^{s^d}) \; ; \; \text{Return } \pi$$

The other algorithms and keys remain unchanged; in particular, key derivation does not make use of these new elements. This extended Waters IBE scheme is secure (in the usual IND-CPA sense for IBE) under the q-type extension of the standard DBDH assumption captured by the game in Figure 5. We define the advantage of an adversary A against the problem as $\mathbf{Adv}^{q\text{-edbdh}}(A) = 2\Pr[q\text{-EDBDH}^A] - 1$.

Theorem 5.3 *Let $\mathcal{IBE} = (\mathcal{S}, \mathcal{P}, \mathcal{K}, \mathcal{E}, \mathcal{D})$ be the extended Waters scheme. Let A be an adversary against $\mathcal{IBE}$ making q_{KD} key derivation queries. Then there is an algorithm B solving the q-Extended Decision Bilinear Diffie-Hellman problem for $q = d$ such that*

$$\mathbf{Adv}^{\mathrm{ibe}}_{\mathcal{IBE}}(A) \leq 32(n+1) \cdot q_{KD} \cdot \mathbf{Adv}^{q\text{-edbdh}}(B) \; . \tag{10}$$

To see this, observe that the original proof of security for Waters' scheme [27, 6] also goes through for the extended scheme, using the elements g, g^x, g^y, T from the q-EDBDH problem to run the simulation as in the original proof and using the additional elements from the q-EDBDH problem to set up the master public key in the extended scheme.

We give evidence for the validity of the q-EDBDH assumption by examining the difficulty of the problem in the generic group model. The problem falls within the framework of the generic group model "master theorem" of Boneh, Boyen and Goh [11]. In their notation, we have $P = \{1, x, x^2, \ldots, x^q, y, x^2y, \ldots, x^qy, z\}$, $Q = 1$, and $f = xyz$. It is clear by inspection that P, Q and f meet the independence requirement of the master theorem, and it gives a lower bound on an adversary's advantage of solving the q-EDBDH problem in a generic group of the form $(q+1)(q_\xi + 4q + 6)^2/p$ where q_ξ is a bound on the number of queries made by the adversary to the oracles computing the group operations in $\mathbb{G}, \mathbb{G}_T$. While a lower bound in the generic group model does not rule out an efficient algorithm when the group is instantiated, it lends heuristic support to our assumption.

The extended Waters IBE scheme is $\Phi^{\mathrm{poly}(d)}$-key malleable with algorithm T as follows:

$T(\pi, u, dk', \phi_{a_0,a_1,\ldots,a_d})$:
If $(a_1 = 0)$ then $r \leftarrow\$ \mathbb{Z}_p$; $dk_1 \leftarrow g_1^{a_0} \cdot H(u)^r \cdot ({g_1}^{s^2})^{a_2} \cdots ({g_1}^{s^d})^{a_d}$; $dk_2 \leftarrow g^r$
Else $dk_1 \leftarrow g_1^{a_0} \cdot dk_1'^{a_1} \cdot ({g_1}^{s^2})^{a_2} \cdots ({g_1}^{s^d})^{a_d}$; $dk_2 \leftarrow dk_2'^{a_1}$
Return (dk_1, dk_2)

The identity renaming scheme is then defined via

$$\mathrm{SI}(s, \overline{u}) = \overline{u} || g^s \quad \text{and} \quad \mathrm{PI}(\pi, \overline{u}, \phi_{a_0,a_1,\ldots,a_d}) = \overline{u} || g^{a_0} \cdot \pi_0^{a_1} \cdot (g^{s^2})^{a_2} \cdots (g^{s^d})^{a_d}$$

which clearly meets the compatibility and collision-resistance requirements. For concreteness, the algorithms of the transformed scheme are given in Appendix A. Combining Theorem 4.1 with Theorem 5.3 gives the following theorem.

Theorem 5.4 *Let $\overline{\mathcal{IBE}} = (\mathcal{S}, \mathcal{P}, \overline{\mathcal{K}}, \overline{\mathcal{E}}, \mathcal{D})$ be the extended Waters scheme under the above identity renaming transform. Let $\overline{A}$ be a $\Phi^{\mathrm{poly}(d)}$-RKA adversary against $\overline{\mathcal{IBE}}$ making q_{KD} key derivation queries. Then there is an algorithm B solving the q-Extended Decision Bilinear Diffie-Hellman problem for $q = d$ such that*

$$\mathbf{Adv}^{\mathrm{ibe-rka}}_{\overline{\mathcal{IBE}},\Phi^{\mathrm{poly}(d)}}(\overline{A}) \leq 32(n+1) \cdot q_{KD} \cdot \mathbf{Adv}^{q\text{-edbdh}}(B) \; . \tag{11}$$

As in the affine case, we may apply results of [4] to obtain a $\Phi^{\mathrm{poly}(d)}$-RKA-secure CCA-PKE scheme and a $\Phi^{\mathrm{poly}(d)}$-RKA-secure signature scheme. We omit the detailed but obvious description of these schemes, noting merely that they are efficient and secure in the standard model under the q-EDBDH assumption.

6 On joint security in the RKA setting

A combined signature and encryption (CSE) scheme [23] is a combination of a signature scheme and a PKE scheme that share a key generation algorithm and hence a key pair. Since the component schemes share a key pair, operations in one component may leak key-dependent material that can be used to break the other component, so the standard notions of security for signatures and PKE need to be enhanced with an additional oracle evaluating the key-dependent operation of the other component.

Paterson et al. [26] gave a construction of a secure CSE scheme from IBE using signatures obtained through the Naor transform [14] and public-key encryption obtained through the CHK transform [12]. Meanwhile, Bellare, Cash and Miller [4] showed that the Naor and CHK transforms individually preserve Φ-RKA-security, so that a Φ-RKA-secure IBE scheme leads to both a Φ-RKA-secure signature scheme and a Φ-RKA-secure CCA-PKE scheme.

In Appendix B, we combine the existing security definitions for Φ-RKA-security and joint security to produce a new security model for Φ-RKA joint security of CSE schemes. There, we also extend the results of [4] and [26] to show that the combined signature and encryption scheme of [26] has Φ-RKA joint security if the starting IBE scheme is Φ-RKA-secure. This construction can be applied to obtain efficient CSE schemes for interesting sets Φ using any of the RKA-secure IBE schemes from this section.

7 RKA for the KEM-DEM paradigm and an RKA-secure CCA-KEM from the BMW scheme

As noted above, the framework for constructing Φ-RKA-secure IBE schemes developed in the previous section can be combined with results from [4] to build Φ-RKA-secure CCA-PKE schemes. In this section, we give a more efficient, direct construction for a Φ-RKA-secure CCA-PKE scheme based on the KEM of Boyen, Mei and Waters [17]. Compared to the schemes of Section 5, our scheme enjoys shorter ciphertexts and smaller public parameters. We begin by providing appropriate definitions for RKA security for Key Encapsulation Mechanisms (KEMs) and show that the KEM/DEM paradigm of [19] extends to the RKA setting in a natural way. We then show how to modify the BMW scheme to build an efficient CCA-secure Key Encapsulation Mechanism (CCA-KEM) that is Φ^{aff}-RKA-secure under the Decision Bilinear Diffie-Hellman assumption for asymmetric pairings. We conclude with a comparison of our scheme to the results of [28].

proc INITIALIZE // KEM
$b \leftarrow_{\$} \{0,1\}$; $C^* \leftarrow \perp$; $(ek, dk) \leftarrow_{\$} \mathcal{K}$
Return ek

proc FINALIZE(b') // KEM
Return $(b = b')$

proc DEC(ϕ, C) // KEM
$dk' \leftarrow \phi(dk)$
If $(dk' = dk \wedge C = C^*)$
  Return $\perp$
Return $K \leftarrow_{\$} \mathcal{D}(dk', C)$

proc CHALLENGE() // KEM
$(C^*, K^*) \leftarrow_{\$} \mathcal{E}(ek)$
If $(b = 0)$ $K^* \leftarrow_{\$} \mathsf{Ksp}$
Return (C^*, K^*)

Figure 6: Game KEM defining Φ-RKA-security for a CCA-KEM.

Algorithm $\mathcal{K}$:
$s \leftarrow_{\$} \mathbb{Z}_p$
$h_0 \leftarrow h^s$
$Z \leftarrow e(g, h_0)$
$y_1, y_2 \leftarrow_{\$} \mathbb{Z}_p$
$u_1 \leftarrow g^{y_1}$; $u_2 \leftarrow g^{y_2}$
$ek \leftarrow (Z, u_1, u_2)$
$dk \leftarrow (h_0, y_1, y_2)$
Return ek, dk

Algorithm $\mathcal{E}(ek)$:
$t \leftarrow_{\$} \mathbb{Z}_p$
$C_1 \leftarrow g^t$
$w \leftarrow H_k(C_1)$
$C_2 \leftarrow u_1^t u_2^{tw}$
$K \leftarrow Z^t$
Return $((C_1, C_2), K)$

Algorithm $\mathcal{D}(dk, C)$:
$w \leftarrow H_k(C_1)$
$w' \leftarrow y_1 + y_2 \cdot w \mod p$
If $(C_1^{w'} \neq C_2)$
  Return $\perp$
Return $e(C_1, h_0)$

Algorithm $\mathcal{K}$:
$s \leftarrow_{\$} \mathbb{Z}_p$
$Z \leftarrow e(g, h)^s$
$y_1, y_2 \leftarrow_{\$} \mathbb{Z}_p$
$u_1 \leftarrow h^{y_1}$; $u_2 \leftarrow h^{y_2}$
$ek \leftarrow (Z, u_1, u_2)$
$dk \leftarrow s$
Return ek, dk

Algorithm $\mathcal{E}(ek)$:
$t \leftarrow_{\$} \mathbb{Z}_p$
$C_1 \leftarrow g^t$
$w \leftarrow H_k(C_1 || Z)$
$C_2 \leftarrow u_1^t u_2^{tw}$
$K \leftarrow Z^t$
Return $((C_1, C_2), K)$

Algorithm $\mathcal{D}(dk, C)$:
$w \leftarrow H_k(C_1 || e(g, h)^{dk})$
If $(e(C_1, u_1 u_2^w) \neq e(g, C_2))$
  Return $\perp$
Return $e(C_1, h)^{dk}$

Figure 7: Original Boyen-Mei-Waters CCA-KEM on the left, RKA-secure variant $\mathcal{BMW}$-$\mathcal{KEM}$ on the right.

7.1 RKA for the KEM-DEM paradigm

We recall the functional definitions of KEMs and Symmetric Encryption (SE) schemes from [19]. We define Φ-RKA-security for a CCA-KEM through the game in Figure 6, which is the natural extension of CCA security for KEMs from [19] to the RKA setting.

We recall in Appendix C the definitions of Φ-RKA-security for CCA-PKE from [4] and of OT-CCA security for SE from [19]. Figure 20 in Appendix C shows how to combine a CCA-KEM $\mathcal{KEM}$ and an SE scheme $\mathcal{SE}$ to build a *hybrid* CCA-PKE scheme $\mathcal{HPKE}$.

We now show that the KEM-DEM composition theorem of [19] also holds in the RKA setting:

Theorem 7.1 *Suppose $\mathcal{KEM}$ is Φ-RKA-secure, and $\mathcal{SE}$ is one-time-CCA secure. Then the hybrid encryption scheme $\mathcal{HPKE}$ is Φ-RKA-secure.*

The proof of Theorem 7.1 is given in Appendix C.

7.2 An RKA-secure CCA-KEM from the BMW scheme

Boyen, Mei and Waters [17] build a very efficient CCA-KEM based on the first IBE scheme of Boneh and Boyen [10] and show the CCA-KEM is secure under the DBDH assumption in the setting of asymmetric pairings. The algorithms of their CCA-KEM are shown on the left-hand side of Figure 7. Here, we are in the setting where $\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T$ are groups of prime order p, $e : \mathbb{G}_1 \times \mathbb{G}_2 \rightarrow \mathbb{G}_T$ is an asymmetric pairing, g is a generator of $\mathbb{G}_1$, h is a generator of $\mathbb{G}_2$, and $H_k : \{0,1\}^* \rightarrow \mathbb{Z}_p$ is a member of a collision-resistant hash function family defined by key k.

The BMW CCA-KEM from [17] is insecure in the RKA setting when the set Φ contains a function $\phi_a(h_0) = h_0^a$ operating on the first element s of the secret key, admitting the following attack. On

receipt of the public key (Z, u_1, u_2), the adversary makes a challenge query and receives a key K^* and a ciphertext C^* of the form $(g^t, u_1^t u_2^{tw})$, where $w = H_k(g^t)$ and t is the randomness used to generate the challenge ciphertext. The adversary then makes a query $\textsc{Dec}(\phi_a, C^*)$ and receives in response the key $K' = e(g,h)^{ast}$. The adversary can then compute the key encapsulated by the challenge ciphertext as $K'^{a^{-1}} = e(g,h)^{st}$ and check if this is equal to the challenge key K^*, outputting $b' = 1$ if so and $b' = 0$ if not, winning the Φ-RKA-security game.

The right half of Figure 7 shows the algorithms of $\mathcal{BMW}$-$\mathcal{KEM}$, an enhanced version of the CCA-KEM of [17] that resists such an attack. We modify the decapsulation algorithm so that it uses the pairing to evaluate ciphertext validity, allowing us to eliminate y_1, y_2 from the secret key. We move some public key elements from $\mathbb{G}_1$ to $\mathbb{G}_2$ to accommodate these changes. We store the exponent s instead of the group element h^s so that the secret key is a single element of $\mathbb{Z}_p$. We then apply the technique of Section 4 by including the secret key dependent part of the public key in the hash, resulting in a Φ^{aff}-RKA-secure CCA-KEM. These modifications result in a scheme that is slightly less efficient than the original CCA-KEM (it has a larger public key since elements in $\mathbb{G}_2$ generally have larger bit representations than elements in $\mathbb{G}_1$, and slower decapsulation since an extra 2 pairings are required).

Theorem 7.2 *Assume DBDH is hard and that H_k is a collision resistant hash function. Then the CCA-KEM $\mathcal{BMW}$-$\mathcal{KEM}$ is Φ^{aff}-RKA-secure.*

The proof of Theorem 7.2 is given in Appendix D.

Our scheme bears comparison to the CCA-PKE schemes of Wee [28]. We reiterate that Wee only achieves security for the claw-free RKD set Φ^{lin}, whereas our scheme $\mathcal{BMW}$-$\mathcal{KEM}$ is Φ^{aff}-RKA-secure. The most directly comparable scheme is the one in [28, Section 5.2], which is presented in the symmetric setting $e : \mathbb{G}_1 \times \mathbb{G}_1 \to \mathbb{G}_T$. To make an accurate comparison with $\mathcal{BMW}$-$\mathcal{KEM}$, and for efficiency at high security levels, this scheme needs to be translated to the asymmetric setting $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$. This can be done in such a way that ciphertexts are kept short, consisting of two elements of $\mathbb{G}_1$ plus a verification key and a signature from a one-time signature scheme, while the public key is also an element of $\mathbb{G}_1$. Here, we view the scheme as a CCA-KEM and so ignore the element $\psi \in \mathbb{G}_T$. The modified scheme's security is based on an asymmetric version of the DBDH assumption, like $\mathcal{BMW}$-$\mathcal{KEM}$.

By comparison, our scheme $\mathcal{BMW}$-$\mathcal{KEM}$ has ciphertexts that consist of two group elements (one in $\mathbb{G}_1$ and one in $\mathbb{G}_2$), while its public keys contain 3 group elements (two in $\mathbb{G}_2$ and one in $\mathbb{G}_T$). Avoiding the overhead of a one-time signature and verification key more than compensates for having an element of $\mathbb{G}_2$ in place of an element of $\mathbb{G}_1$ in ciphertexts, and so our ciphertexts are more compact. On the other hand, our public key is larger. The costs of encapsulation and decapsulation for $\mathcal{BMW}$-$\mathcal{KEM}$ are roughly the same as for Wee's scheme: in both schemes, encapsulation is pairing-free while decapsulation requires 3 pairings (a more detailed comparison would count the number of exponentiations needed in the two schemes).

In summary, the two schemes have roughly comparable performance, but our scheme $\mathcal{BMW}$-$\mathcal{KEM}$ is RKA-secure for a significantly larger class of RKD functions.

8 RKA-secure CCA-SE

We now show how to build RKA-secure CCA-SE starting from an IBE scheme meeting certain malleability properties and admitting a collision-resistant identity renaming scheme. We show that applying the identity renaming transform of Section 4 to such an IBE scheme results in an IBE scheme meeting a notion of strong RKA security, wherein the challenge ciphertext is encrypted under a related key of the adversary's choosing. Waters' IBE scheme and the extended Waters IBE scheme are shown to have the required properties.

We then show that the CHK transform preserves strong RKA security, leading to strong Φ-RKA-secure PKE from selective-ID strong Φ-RKA-secure IBE. We then show that the natural method for

proc INITIALIZE // strIBE
$s \leftarrow\$ \mathcal{S}; \pi \leftarrow \mathcal{P}(s)$
$b \leftarrow\$ \{0,1\}$; $u^*, s^* \leftarrow \perp$
$I \leftarrow \emptyset$
Return π

proc FINALIZE(b') // strIBE
Return $(b = b')$

proc KD(ϕ, u) // strIBE
$s' \leftarrow \phi(s)$
$I \leftarrow I \cup \{(u, s')\}$
If $((u^*, s^*) \in I)$ Return $\perp$
Return $dk \leftarrow\$ \mathcal{K}(s', u)$

proc ENC(ϕ, u, M) // strIBE
$s' \leftarrow \phi(s)$
$\pi' \leftarrow \mathcal{P}(s')$
Return $C \leftarrow\$ \mathcal{E}(\pi', u, M)$

proc LR(ϕ, u, M_0, M_1) // strIBE
If $(|M_0| \neq |M_1|)$ Return $\perp$
$u^* \leftarrow u$
$s^* \leftarrow \phi(s)$
If $((u^*, s^*) \in I)$ Return $\perp$
$\pi^* \leftarrow \mathcal{P}(s^*)$
Return $C \leftarrow\$ \mathcal{E}(\pi^*, u^*, M_b)$

Figure 8: Game strIBE defining strong Φ-RKA-security of IBE scheme $\mathcal{IBE} = (\mathcal{S}, \mathcal{P}, \mathcal{K}, \mathcal{E}, \mathcal{D})$.

converting a PKE scheme into a symmetric encryption scheme gives a Φ-RKA-secure CCA-SE scheme when the PKE scheme is strong Φ-RKA-secure.

8.1 Strong RKA-secure IBE

Figure 8 gives a definition of strong Φ-RKA security for an IBE scheme. In strong RKA security, in addition to obtaining user-level keys computed under related master keys, the adversary may obtain the challenge encryption under a related master key. As in standard RKA security, the adversary is allowed only one call to the LR oracle. An alternative definition of security could be considered where instead of having access to an ENC oracle the adversary is given access to an oracle returning the public key corresponding to a related key deriving function. The schemes considered here can be shown to meet this definition, but meeting that of Figure 8 is sufficient as a step on the way to our end goal of RKA-secure CCA-SE.

MASTER PUBLIC KEY-MALLEABILITY. In Section 4 we showed how to use an identity renaming transform to construct an RKA-secure IBE scheme from an IBE scheme that is key malleable and has a collision-resistant identity renaming scheme. The additional property required to prove strong RKA security is that of master public key-malleability. We say that an IBE scheme $\mathcal{IBE} = (\mathcal{S}, \mathcal{P}, \mathcal{K}, \mathcal{E}, \mathcal{D})$ is Φ-mpk-malleable if there exists an algorithm R and a function $f \in \mathsf{Fun}(\mathsf{MSp}, \mathsf{MSp})$ such that

$$R(\mathcal{E}(\mathcal{P}(s), u, f(M); t), \phi) = \mathcal{E}(\mathcal{P}(\phi(s)), u, M; t)$$

for all $s \in \mathcal{S}$, $u \in \mathsf{USp}$, $M \in \mathsf{MSp}$, $\phi \in \Phi$ and randomness t. This property says that we can take a ciphertext encrypting a message $f(M)$ related to M under master public key $\mathcal{P}(s)$ and use it to build a ciphertext encrypting M under master public key $\mathcal{P}(\phi(s))$, for the same user u and under the same randomness, t.

Waters' IBE scheme is Φ-mpk-malleable for affine ϕ with $a \neq 0$. The function f is $f(M) = M^{a^{-1}}$ and the algorithm R taking ciphertext $C = (C'_1, C'_2, C'_3)$ is as follows:

$R(C'_1, C'_2, C'_3, \phi_{a,b})$:
  $C_1 \leftarrow C'_1$; $C_2 \leftarrow C'_2$
  $C_3 \leftarrow (C'_3)^a \cdot e((C'_1)^b, g_1)$
  Return (C_1, C_2, C_3)

Let $C \leftarrow \mathcal{E}(\mathcal{P}(s), u, M^{a^{-1}}; t) = (g^t, H(u)^t, e(\mathcal{P}(s), g_1)^t \cdot M^{a^{-1}})$ be the ciphertext input to R. Then

$$\begin{aligned} C_3 &= (C'_3)^a \cdot e((C'_1)^b, g_1) \\ &\quad (e(g^s, g_1)^t \cdot M^{a^{-1}})^a \cdot e((g^t)^b, g_1) \\ &\quad e(g, g_1)^{ast} \cdot M \cdot e(g, g_1)^{bt} \end{aligned}$$

$$
\begin{array}{l}
e(g,g_1)^{(as+b)t} \cdot M \\
e(g^{(as+b)},g_1)^{t} \cdot M \\
e(\mathcal{P}(\phi(s)),g_1)^{t} \cdot M \quad ,
\end{array}
$$

so $(C_1, C_2, C_3) = (g^t, H(u)^t, e(\mathcal{P}(\phi(s)), g_1)^t \cdot M) = \mathcal{E}(\mathcal{P}(\phi(s)), u, M; t)$ and R's output is correct.

The extended Waters IBE scheme is Φ-mpk-malleable for polynomial $\phi_{a_0,a_1,\ldots,a_d}$ with linear coefficient $a_1 \neq 0$. The function f is $f(M) = M^{a_1^{-1}}$ and the algorithm R taking ciphertext $C = (C'_1, C'_2, C'_3)$ is as follows:

$R(C'_1, C'_2, C'_3, \phi_{a_0,a_1,\ldots,a_d})$:
$C_1 \leftarrow C'_1$; $C_2 \leftarrow C'_2$
$C_3 \leftarrow (C'_3)^{a_1} \cdot e((C'_1)^{a_0}, g_1) \cdot e((C'_1)^{a_2}, (g_1^{s^2})) \cdots e((C'_1)^{a_d}, (g_1^{s^d}))$
Return (C_1, C_2, C_3)

Let $C \leftarrow \mathcal{E}(\mathcal{P}(s), u, M^{a_1^{-1}}; t) = (g^t, H(u)^t, e(\mathcal{P}(s), g_1)^t \cdot M^{a_1^{-1}})$ be the ciphertext input to R. Then

$$
\begin{array}{l}
C_3 = (C'_3)^{a_1} \cdot e((C'_1)^{a_0}, g_1) \cdot e((C'_1)^{a_2}, (g_1^{s^2})) \cdots e((C'_1)^{a_d}, (g_1^{s^d})) \\
\quad (e(g^s, g_1)^t \cdot M^{a_1^{-1}})^{a_1} \cdot e((g^t)^{a_0}, g_1) \cdot e((g^t)^{a_2}, (g_1^{s^2})) \cdots e((g^t)^{a_d}, (g_1^{s^d})) \\
\quad e(g,g_1)^{a_1 st} \cdot M \cdot e(g,g_1)^{a_0 t} \cdot e(g,g_1)^{a_2 s^2 t} \cdots e(g,g_1)^{a_d s^d t} \\
\quad e(g,g_1)^{(a_0 + a_1 s + a_2 s^2 + \cdots + a_d s^d)t} \cdot M \\
\quad e(g^{(a_0 + a_1 s + a_2 s^2 + \cdots + a_d s^d)}, g_1)^{t} \cdot M \\
\quad e(\mathcal{P}(\phi(s)), g_1)^{t} \cdot M \quad ,
\end{array}
$$

so $(C_1, C_2, C_3) = (g^t, H(u)^t, e(\mathcal{P}(\phi(s)), g_1)^t \cdot M) = \mathcal{E}(\mathcal{P}(\phi(s)), u, M; t)$ and R's output is correct.

The restriction that the non-linear coefficient of the RKD polynomial be non-zero comes from the algorithm R inverting this coefficient. Strong RKA security is unachievable for constant RKD functions $\phi(s) = a_0$ as the adversary can submit such a ϕ to the LR oracle to obtain the challenge ciphertext under a public key for which it knows the corresponding secret key is a_0. Thus strong Φ-RKA security for the full set of polynomials $\phi_{a_0,a_1,\ldots,a_d}$ is unachievable. However there is still a gap between the RKD set consisting of polynomials of non-zero degree and our RKD set of polynomials with non-zero linear coefficient.

Theorem 8.1 *Let $\mathcal{IBE} = (\mathcal{S}, \mathcal{P}, \mathcal{K}, \mathcal{E}, \mathcal{D})$ be a Φ-key-malleable IBE scheme with key simulator T. Suppose further that $\mathcal{IBE}$ is Φ-mpk-key-malleable with functions R and f . Let $\overline{\mathcal{IBE}} = (\mathcal{S}, \mathcal{P}, \overline{\mathcal{K}}, \overline{\mathcal{E}}, \mathcal{D})$ be obtained from $\mathcal{IBE}$ and renaming scheme* (SI, PI) *via the transform* **IRT** *described in Section 4. Assume the renaming scheme is statistically collision-resistant. Let $\overline{A}$ be a strong Φ-RKA adversary against $\overline{\mathcal{IBE}}$ that makes q key derivation queries. Then there is an adversary A making q key derivation queries such that*

$$\mathbf{Adv}^{\mathrm{ibe-srka}}_{\overline{\mathcal{IBE}},\Phi}(\overline{A}) \leq \mathbf{Adv}^{\mathrm{ibe}}_{\mathcal{IBE}}(A) \; . \tag{12}$$

Proof: The proof is very similar to that of Theorem 4.1. Consider the games of Figure 9. Game G_0 is written to be equivalent to game $\mathrm{strIBE}_{\overline{\mathcal{IBE}}}$, so that

$$\mathbf{Adv}^{\mathrm{ibe-srka}}_{\overline{\mathcal{IBE}},\Phi}(\overline{A}) \;=\; 2\Pr[\mathrm{G}_0^{\overline{A}}] - 1 \; . \tag{13}$$

In answering a $\mathrm{KD}(\phi, \overline{u})$ query, G_0 must use the key-generation algorithm $\overline{\mathcal{K}}$ of the new scheme $\overline{\mathcal{IBE}}$ but with master secret key $s' = \phi(s)$. From the definition of $\overline{\mathcal{K}}$, it follows that not only is the key-generation at line 014 done under s', but also the identity renaming at line 013. LR, correspondingly, should use $\overline{\mathcal{E}}$, and thus the public renaming function PI. The compatibility property however allows us at line 025 to use SI instead. This will be useful in exploiting statistical collision-resistance in the next step, after which we will revert back to PI.

proc INITIALIZE // G_0
000 $s \leftarrow\!\!\$\ \mathcal{S}$; $\pi \leftarrow \mathcal{P}(s)$
001 $b \leftarrow\!\!\$\ \{0,1\}$; $\overline{u}^*, s^* \leftarrow \bot$
002 $\overline{I} \leftarrow \emptyset$
003 Return π

proc KD($\phi, \overline{u}$) // G_0
010 $s' \leftarrow \phi(s)$
011 $\overline{I} \leftarrow \overline{I} \cup \{(\overline{u}, s')\}$
012 If $((\overline{u}^*, s^*) \in \overline{I})$ Return $\bot$
013 $u \leftarrow \mathrm{SI}(s', \overline{u})$
014 Return $\overline{dk} \leftarrow\!\!\$\ \mathcal{K}(s', u)$

proc LR($\phi, \overline{u}, M_0, M_1$) // G_0
020 If $(|M_0| \neq |M_1|)$ Return $\bot$
021 $\overline{u}^* \leftarrow \overline{u}$
022 $s^* \leftarrow \phi(s)$
023 If $((\overline{u}^*, s^*) \in \overline{I})$ Return $\bot$
024 $\pi^* \leftarrow \mathcal{P}(s^*)$
025 $u^* \leftarrow \mathrm{SI}(s^*, \overline{u}^*)$
026 Return $C \leftarrow\!\!\$\ \mathcal{E}(\pi^*, u^*, M_b)$

proc ENC($\phi, \overline{u}, M$) // G_0, G_1, G_2
030 $s' \leftarrow \phi(s)$
031 $\pi' \leftarrow \mathcal{P}(s')$
032 $u \leftarrow \mathrm{PI}(\pi', \overline{u}, \mathsf{id})$
033 Return $C \leftarrow\!\!\$\ \mathcal{E}(\pi', u, M)$

proc FINALIZE(b') // All
040 Return $(b = b')$

proc INITIALIZE // G_1, G_2, G_3, G_4
100 $s \leftarrow\!\!\$\ \mathcal{S}$; $\pi \leftarrow \mathcal{P}(s)$
101 $b \leftarrow\!\!\$\ \{0,1\}$; $u^* \leftarrow \bot$
102 $I \leftarrow \emptyset$
103 Return π

proc KD($\phi, \overline{u}$) // G_1
110 $s' \leftarrow \phi(s)$
111 $u \leftarrow \mathrm{SI}(s', \overline{u})$
112 $I \leftarrow I \cup \{u\}$
113 If $(u^* \in I)$ Return $\bot$
114 Return $\overline{dk} \leftarrow\!\!\$\ \mathcal{K}(s', u)$

proc LR($\phi, \overline{u}, M_0, M_1$) // G_1
120 If $(|M_0| \neq |M_1|)$ Return $\bot$
121 $s^* \leftarrow \phi(s)$
122 $u^* \leftarrow \mathrm{SI}(s^*, \overline{u})$
123 If $(u^* \in I)$ Return $\bot$
124 $\pi^* \leftarrow \mathcal{P}(s^*)$
125 Return $C \leftarrow\!\!\$\ \mathcal{E}(\pi^*, u^*, M_b)$

proc KD($\phi, \overline{u}$) // G_2, G_3
210 $u \leftarrow \mathrm{PI}(\pi, \overline{u}, \phi)$
211 $I \leftarrow I \cup \{u\}$
212 If $(u^* \in I)$ Return $\bot$
213 Return $\overline{dk} \leftarrow\!\!\$\ \mathcal{K}(\phi(s), u)$

proc LR($\phi, \overline{u}, M_0, M_1$) // G_2
220 If $(|M_0| \neq |M_1|)$ Return $\bot$
221 $u^* \leftarrow \mathrm{PI}(\pi, \overline{u}, \phi)$
222 If $(u^* \in I)$ Return $\bot$
223 $\pi^* \leftarrow \mathcal{P}(s^*)$
224 Return $C \leftarrow\!\!\$\ \mathcal{E}(\pi^*, u^*, M_b)$

proc KD($\phi, \overline{u}$) // G_4
410 $u \leftarrow \mathrm{PI}(\pi, \overline{u}, \phi)$
411 $I \leftarrow I \cup \{u\}$
412 If $(u^* \in I)$ Return $\bot$
413 $dk \leftarrow\!\!\$\ \mathcal{K}(s, u)$
414 Return $\overline{dk} \leftarrow T(\pi, u, dk, \phi)$

proc LR($\phi, \overline{u}, M_0, M_1$) // G_3, G_4
420 If $(|M_0| \neq |M_1|)$ Return $\bot$
421 $M \leftarrow f(M_b)$
422 $u^* \leftarrow \mathrm{PI}(\pi, \overline{u}, \phi)$
423 If $(u^* \in I)$ Return $\bot$
424 $C' \leftarrow\!\!\$\ \mathcal{E}(\pi, u^*, M)$
425 Return $C \leftarrow\!\!\$\ R(C, \phi)$

proc ENC($\phi, \overline{u}, M$) // G_3, G_4
430 $M' \leftarrow f(M)$
431 $u \leftarrow \mathrm{PI}(\pi, \overline{u}, \phi)$
432 $C' \leftarrow\!\!\$\ \mathcal{E}(\pi, u, M')$
433 Return $C \leftarrow\!\!\$\ R(C, \phi)$

Figure 9: Games for proof of Theorem 8.1.

The adversary A we aim to construct will not know s. A central difficulty in the simulation is thus lines 011, 012 of G_0 where the response provided to $\overline{A}$ depends on the result of a test involving s, a test that A cannot perform. Before we can design A we must get rid of this test. Statistical collision-resistance is what will allow us to do so. KD of game G_1 moves the identity renaming up before the list of queried identities is updated to line 111 and then, at line 112, adds the transformed identity to the list. LR is likewise modified so its test now involves the transformed (rather than original) identities. We claim this makes no difference, meaning

$$\Pr[\mathrm{G}_0^{\overline{A}}] \;=\; \Pr[\mathrm{G}_1^{\overline{A}}]\,. \tag{14}$$

Indeed, statistical collision-resistance tell us that $(s', \overline{u}) = (s^*, \overline{u}^*)$ iff $\mathrm{SI}(s', \overline{u}) = \mathrm{SI}(s^*, \overline{u}^*)$. This means that lines 011, 012 and lines 112, 113 are equivalent.

Compatibility is invoked to use PI in place of SI in both KD and in LR in G_2, so that

$$\Pr[\mathrm{G}_1^{\overline{A}}] \;=\; \Pr[\mathrm{G}_2^{\overline{A}}]\,. \tag{15}$$

Game G_3 makes changes to encryption and challenge ciphertext generation. Φ-mpk-malleability means these changes are invisible to the adversary and so

$$\Pr[G_2^{\overline{A}}] \;=\; \Pr[G_3^{\overline{A}}]\,. \tag{16}$$

By compatibility the user u computed at lines 032 and 431 is the same, and by Φ-mpk-malleability the ciphertext C output at line 033 is identical to that output at line 433, so from the adversary's point of view ENC behaves the same way in games G_2 and G_3. Φ-mpk-malleability means LR also outputs the same ciphertexts in both games.

Rather than use s' for key generation as at 213, G_4 uses s at 413 and then applies the key simulator T. We claim the key-malleability implies

$$\Pr[G_3^{\overline{A}}] \;=\; \Pr[G_4^{\overline{A}}]\,. \tag{17}$$

To justify this we show that there is an adversary M such that

$$\Pr[\mathrm{KMReal}^{M}_{\mathcal{IBE},\Phi}] \;=\; \Pr[G_3^{\overline{A}}] \qquad \text{and} \qquad \Pr[\mathrm{KMSim}^{M}_{\mathcal{IBE},\Phi,T}] \;=\; \Pr[G_4^{\overline{A}}]\,.$$

Adversary M, on input π, begins with the initializations $u^* \leftarrow \perp$; $I \leftarrow \emptyset$; $b \leftarrow\!\!\$\ \{0,1\}$ and then runs $\overline{A}$ on input π. M responds to encryption queries as specified in the game. When $\overline{A}$ makes a $\mathrm{KD}(\phi, \overline{u})$ query, M does the following:

$$u \leftarrow \mathrm{PI}(\pi, \overline{u}, \phi)\ ;\ I \leftarrow I \cup \{u\}\ ;\ \text{If } (u^* \in I) \text{ Return } \perp\ ;\ \overline{dk} \leftarrow \mathrm{KD}(\phi, u).$$

If M is playing game KMReal then its KD oracle will behave as line 213 in game G_3, while if M is playing game KMSim its KD oracle will behave as lines 413,414 in game G_4. When $\overline{A}$ makes its $\mathrm{LR}(\phi, \overline{u}, M_0, M_1)$ query M sets $u^* \leftarrow \mathrm{PI}(\pi, \overline{u}, \phi)$ and checks if $u^* \in I$, returning $\perp$ if so. M then computes $C \leftarrow R(\mathcal{E}(\pi, u^*, f(M_b)), \phi)$ which it returns to $\overline{A}$. When $\overline{A}$ halts with output b', M returns the result of $(b' = b)$. If M is playing game KMReal then game G_3 is perfectly simulated, while if M is playing KMSim then game G_4 is perfectly simulated, so M returns 1 with the same probability that $\overline{A}$ wins in each case and by the key-malleability of $\mathcal{IBE}$ Equation (6) holds.

Finally, we design A so that

$$\mathbf{Adv}^{\mathrm{ibe}}_{\mathcal{IBE}}(A) \;=\; 2\Pr[G_4^{\overline{A}}] - 1\,. \tag{18}$$

On input π, adversary A runs $\overline{A}(\pi)$. When the latter makes a $\mathrm{KD}(\phi, \overline{u})$ query, A does the following:

$$u \leftarrow \mathrm{PI}(\pi, \overline{u}, \phi)\ ;\ dk \leftarrow \mathrm{KD}(\mathsf{id}, u)\ ;\ \overline{dk} \leftarrow T(\pi, u, dk, \phi).$$

It then returns $\overline{dk}$ to $\overline{A}$. The KD invoked in this code is A's own oracle. Compatibility tells us that $u = \mathrm{SI}(\phi(s), \overline{u})$ and thus from the definition of $\overline{\mathcal{IBE}}$, the response to $\overline{A}$'s query is distributed according to $\mathcal{K}(\phi(s), u)$. But key-malleability then tells us that $\overline{dk}$ is distributed identically to this, so the response provided by A is perfectly correct. When $\overline{A}$ makes a $\mathrm{LR}(\phi, \overline{u}, M_0, M_1)$ query, A does the following:

$$u \leftarrow \mathrm{PI}(\pi, \overline{u}, \phi)\ ;\ C' \leftarrow \mathrm{LR}(u, f(M_0), f(M_1))\ ;\ C \leftarrow R(C', \phi).$$

It then returns C to $\overline{A}$. The LR invoked in this code is A's own oracle. The definition of $\overline{\mathcal{IBE}}$ implies that the response provided by A is again perfectly correct. Finally when $\overline{A}$ halts with output a bit b', adversary A does the same. ∎

8.2 Strong RKA secure PKE from IBE

Figure 10 shows the definition of strong Φ-RKA-security for CCA-PKE, an extension of Φ-RKA security where the adversary may request the challenge ciphertext be encrypted under a related key. As with

proc INITIALIZE // strPKE
$b \leftarrow\!\!\$\ \{0,1\}$; $sk^*, C^* \leftarrow \bot$
$sk \leftarrow\!\!\$\ \mathcal{S}$; $pk \leftarrow \mathcal{P}(sk)$
Return pk

proc ENC(ϕ, M) // strPKE
$sk' \leftarrow \phi(sk)$
$pk' \leftarrow \mathcal{P}(sk')$
Return $C^* \leftarrow\!\!\$\ \mathcal{E}(pk', M)$

proc DEC(ϕ, C) // strPKE
$sk' \leftarrow \phi(sk)$
If $((sk' = sk^*) \wedge (C = C^*))$
  Return $\bot$
Return $M \leftarrow \mathcal{D}(sk', C)$

proc LR(ϕ, M_0, M_1) // strPKE
If $(|M_0| \neq |M_1|)$
  Return $\bot$
$sk^* \leftarrow \phi(sk)$
$pk^* \leftarrow \mathcal{P}(sk^*)$
Return $C^* \leftarrow\!\!\$\ \mathcal{E}(pk^*, M_b)$

proc FINALIZE(b') // strPKE
Return $(b = b')$

proc INITIALIZE // strsIBE
$s \leftarrow\!\!\$\ \mathcal{S}$; $\pi \leftarrow \mathcal{P}(s)$
$b \leftarrow\!\!\$\ \{0,1\}$; $u^*, s^* \leftarrow \bot$

proc SID(u) // strsIBE
$u^* \leftarrow u$
Return π

proc KD(ϕ, u) // strsIBE
If $(u^* = \bot)$ Return $\bot$
$s' \leftarrow \phi(s)$
If $(s' = s^* \wedge u = u^*)$ Return $\bot$
Return $dk \leftarrow\!\!\$\ \mathcal{K}(s', u)$

proc ENC(ϕ, u, M) // strsIBE
If $(u^* = \bot)$ Return $\bot$
$s' \leftarrow \phi(s)$
$\pi' \leftarrow \mathcal{P}(s')$
Return $C \leftarrow\!\!\$\ \mathcal{E}(\pi', u, M)$

proc LR(ϕ, M_0, M_1) // strsIBE
If $(|M_0| \neq |M_1|)$ Return $\bot$
If $(u^* = \bot)$ Return $\bot$
$s^* \leftarrow \phi(s)$
$\pi^* \leftarrow \mathcal{P}(s^*)$
Return $C \leftarrow\!\!\$\ \mathcal{E}(\pi^*, u^*, M_b)$

proc FINALIZE(b') // strsIBE
Return $(b = b')$

Algorithm $\mathcal{E}'(\pi, M)$:
$(\overline{vk}, \overline{sk}) \leftarrow\!\!\$\ \overline{\mathcal{K}}$
$c \leftarrow\!\!\$\ \mathcal{E}(\pi, \overline{vk}, M)$
$\overline{\sigma} \leftarrow\!\!\$\ \overline{\mathcal{S}}(\overline{sk}, c)$
$C \leftarrow (c, \overline{vk}, \overline{\sigma})$
Return C

Algorithm $\mathcal{D}'(s, (c, \overline{vk}, \overline{\sigma}))$:
If $(\overline{\mathcal{V}}(\overline{vk}, c, \overline{\sigma}) = 0)$ Return $\bot$
$dk \leftarrow\!\!\$\ \mathcal{K}(s, \overline{vk})$
$M \leftarrow \mathcal{D}(dk, c)$
Return M

Figure 10: Game strPKE defining strong Φ-RKA-security for CCA-PKE, game strsIBE defining strong Φ-RKA-selective-ID security for $\mathcal{IBE}$, CHK PKE scheme.

IBE the definition assumes the secret key s is chosen at random from a secret key space $\mathcal{S}$ and the public key is deterministically derived from it through an algorithm $\mathcal{P}$. We show that applying the CHK transform to a strong Φ-RKA-selective-ID secure IBE scheme results in a strong Φ-RKA-secure CCA-PKE scheme.

Theorem 8.2 *Let $\mathcal{PKE} = (\mathcal{S}, \mathcal{P}, \mathcal{E}', \mathcal{D}')$ be the public key encryption scheme constructed from strong Φ-RKA-selective-ID secure $\mathcal{IBE} = (\mathcal{S}, \mathcal{P}, \mathcal{K}, \mathcal{E}, \mathcal{D})$ and strongly unforgeable one time signature scheme $\overline{\mathcal{DS}} = (\overline{\mathcal{K}}, \overline{\mathcal{S}}, \overline{\mathcal{V}})$ through the CHK transform as in Figure 10. Then $\mathcal{PKE}$ is strong Φ-RKA-secure.*

Proof: The proof uses the sequence of games in Figure 11. Game G_0 implements strPKE^A with $\mathcal{PKE}$, so we have

$$\mathbf{Adv}^{\mathrm{pke-srka}}_{\mathcal{PKE},\Phi}(A) = 2\Pr[\mathrm{G}_0^A] - 1.$$

Game G_1 has an additional check in the DEC oracle: now if $\overline{vk} = \overline{vk}^*$ and $(c, \overline{\sigma}) \neq (c^*, \overline{\sigma}^*)$, the game sets bad and responds to the query with $\bot$. Since G_1 and G_0 are identical until bad, we have

$$\Pr[\mathrm{G}_1^A] - \Pr[\mathrm{G}_0^A] \leq \Pr[E_1],$$

where E_1 is the event that G_1 sets bad. We now construct an adversary B that breaks the strong unforgeability of $\overline{\mathcal{DS}}$ with probability $\Pr[E_1]$. B is given as input $\overline{vk}^*$ and starts by selecting $b \leftarrow\!\!\$\ \{0,1\}$, $s \leftarrow\!\!\$\ \mathcal{S}$ and computing $\pi \leftarrow \mathcal{P}(s)$. It runs $A(\pi)$ and simulates the response to the LR query by computing

If $(|M_0| \neq |M_1|)$ Return $\bot$

proc INITIALIZE // G_0, G_1, G_2
$b \leftarrow\$ \{0,1\}$; $s^*, C^* \leftarrow \bot$
$s \leftarrow\$ \mathcal{S}$; $\pi \leftarrow \mathcal{P}(s)$
Return π

proc LR(ϕ, M_0, M_1) // G_0, G_1, G_2
If $(|M_0| \neq |M_1|)$ Return $\bot$
$s^* \leftarrow \phi(s)$
$\pi^* \leftarrow \mathcal{P}(s^*)$
$(\overline{vk}^*, \overline{sk}^*) \leftarrow\$ \overline{\mathcal{K}}$
$c^* \leftarrow\$ \mathcal{E}(\pi^*, \overline{vk}^*, M_b)$
$\overline{\sigma}^* \leftarrow\$ \overline{\mathcal{S}}(\overline{sk}^*, c^*)$
$C^* \leftarrow (c^*, \overline{vk}^*, \overline{\sigma}^*)$
Return C^*

proc FINALIZE(b') // G_0, G_1, G_2
Return $(b = b')$

proc ENC(ϕ, M) // G_0, G_1, G_2
$s' \leftarrow \phi(s)$
$\pi' \leftarrow \mathcal{P}(s')$
$(\overline{vk}, \overline{sk}) \leftarrow\$ \overline{\mathcal{K}}$
$c' \leftarrow\$ \mathcal{E}(\pi', \overline{vk}, M)$
$\overline{\sigma} \leftarrow\$ \overline{\mathcal{S}}(\overline{sk}, c)$
$C \leftarrow (c, \overline{vk}, \overline{\sigma})$
Return C

proc DEC$(\phi, (c, \overline{vk}, \overline{\sigma}))$ // G_0,$\boxed{G_1}$
$s' \leftarrow \phi(s)$
If $((s' = s^*) \wedge ((c, \overline{vk}, \overline{\sigma}) = C^*))$
  Return $\bot$
If $(\overline{\mathcal{V}}(\overline{vk}, c, \overline{\sigma}) = 0)$
  Return $\bot$
If $((\overline{vk} = \overline{vk}^*) \wedge ((c, \overline{\sigma}) \neq (c^*, \overline{\sigma}^*)))$
  bad $\leftarrow$ true ; $\boxed{\text{Return } \bot}$
$dk \leftarrow\$ \mathcal{K}(s', \overline{vk})$
Return $M \leftarrow \mathcal{D}(dk, c)$

proc DEC$(\phi, (c, \overline{vk}, \overline{\sigma}))$ // G_2
If $(\overline{\mathcal{V}}(\overline{vk}, c, \overline{\sigma}) = 0)$
  Return $\bot$
If $((\overline{vk} = \overline{vk}^*) \wedge ((c, \overline{\sigma}) \neq (c^*, \overline{\sigma}^*)))$
  Return $\bot$
$s' \leftarrow \phi(s)$
If $((s' = s^*) \wedge (\overline{vk} = \overline{vk}^*))$
  Return $\bot$
$dk \leftarrow \mathcal{K}(s', \overline{vk})$
Return $M \leftarrow \mathcal{D}(dk, c)$

Figure 11: Games for the proof of Theorem 8.2.

$s^* \leftarrow \phi(s)$; $\pi^* \leftarrow \mathcal{P}(s^*)$
$c^* \leftarrow\$ \mathcal{E}(\pi^*, \overline{vk}^*, M_b)$;
$\overline{\sigma}^* \leftarrow\$ \overline{\mathcal{S}}(\overline{sk}^*, c^*)$; $C^* \leftarrow (c^*, \overline{vk}^*, \overline{\sigma}^*)$; Return C^*

using its own signing oracle to generate $\overline{\sigma}^*$ on c^*. It also simulates DEC and ENC query responses exactly as specified in G_1, noting the first query that triggers bad, if any. If there is such a query DEC$(\phi, c, \overline{vk}, \overline{\sigma})$, then B outputs $(c, \overline{\sigma})$ as its forgery. This is a valid forgery for B because this query must have passed the validity check $\overline{\mathcal{V}}(\overline{vk}, c, \overline{\sigma})$ (otherwise the oracle would have returned before going to set bad), and we have that $(c, \overline{\sigma}) \neq (c^*, \overline{\sigma}^*)$ by the check immediately before bad is set.

Game G_2 rearranges some of the validity checks in DEC oracle processing, but these changes don't affect oracle responses. This change is valid because the same ciphertexts end up being rejected in either game. We have

$$\Pr[G_2^A] = \Pr[G_1^A] \ .$$

We are now in a position to show that an adversary winning G_2 with good probability can be used to break the strong Φ-RKA-selective-ID security of $I\mathcal{BE}$. We construct A' such that

$$\mathbf{Adv}^{\text{sibe-srka}}_{I\mathcal{BE},\Phi}(A') = 2\Pr[G_2^A] - 1.$$

A' generates $(\overline{vk}^*, \overline{sk}^*) \leftarrow\$ \overline{\mathcal{K}}$ and calls its SID oracle with $\overline{vk}^*$, receiving π in return. A' runs $A(\pi)$, simulating the response to LR(ϕ, M_0, M_1) by querying its own oracle for $c^* \leftarrow$ LR(ϕ, M_0, M_1), and then signing c^* using $\overline{sk}^*$, finally returning the ciphertext $C^* = (c^*, \overline{vk}^*, \overline{\sigma}^*)$. A' simulates responses to ENC(ϕ, M) by querying its own oracle for $c \leftarrow$ ENC(ϕ, M) and generating $(\overline{vk}, \overline{sk}) \leftarrow\$ \overline{\mathcal{K}}$ then signing c using $\overline{sk}$, returning the ciphertext $C = (c, \overline{vk}, \overline{\sigma})$. A' simulates responses to DEC$(\phi, (c, \overline{vk}, \overline{\sigma}))$ by computing

If $((\overline{\mathcal{V}}(\overline{vk}, c, \overline{\sigma}) = 0) \vee ((\overline{vk} = \overline{vk}^*) \wedge ((c, \overline{\sigma}) \neq (c^*, \overline{\sigma}^*))))$ Return $\bot$;
$dk \leftarrow \mathrm{KD}(\phi, \overline{vk})$; $M \leftarrow \mathcal{D}(dk, c)$; Return M.

A' runs A until it halts, and outputs whatever A outputs.

$\underline{\text{proc INITIALIZE}}$ // SE
$b \leftarrow\!\!\$\ \{0,1\}\ ;\ K^*, C^* \leftarrow \bot$
$K \leftarrow\!\!\$\ \mathcal{K}$
$\underline{\text{proc ENC}(\phi, M)}$ // SE
$K' \leftarrow \phi(K)$
Return $C \leftarrow\!\!\$\ \mathcal{E}(K', M)$
$\underline{\text{proc DEC}(\phi, C)}$ // SE
$K' \leftarrow \phi(K)$
If $((K' = K^*) \wedge (C = C^*))$
  Return $\bot$
Return $M \leftarrow \mathcal{D}(K', C)$
$\underline{\text{proc LR}(\phi, M_0, M_1)}$ // SE
If $(|M_0| \neq |M_1|)$
  Return $\bot$
$K^* \leftarrow \phi(K)$
Return $C^* \leftarrow\!\!\$\ \mathcal{E}(K^*, M_b)$
$\underline{\text{proc FINALIZE}(b')}$ // SE
Return $(b = b')$

$\underline{\text{Algorithm } \mathcal{K}'\text{:}}$
$K \leftarrow\!\!\$\ \mathcal{S}$
Return K
$\underline{\text{Algorithm } \mathcal{E}'(K, M)\text{:}}$
$\pi \leftarrow \mathcal{P}(K)$
Return $C \leftarrow\!\!\$\ \mathcal{E}(\pi, M)$
$\underline{\text{Algorithm } \mathcal{D}'(K, C)\text{:}}$
Return $M \leftarrow \mathcal{D}(K, C)$

$\underline{\text{proc INITIALIZE}}$ // G_0
$b \leftarrow\!\!\$\ \{0,1\}\ ;\ K^*, C^* \leftarrow \bot$
$K \leftarrow\!\!\$\ \mathcal{S}$
$\underline{\text{proc ENC}(\phi, M)}$ // G_0
$K' \leftarrow \phi(K)$
$\pi \leftarrow \mathcal{P}(K')$
Return $C \leftarrow\!\!\$\ \mathcal{E}(\pi, M)$
$\underline{\text{proc DEC}(\phi, C)}$ // G_0
$K' \leftarrow \phi(K)$
If $((K' = K^*) \wedge (C = C^*))$
  Return $\bot$
Return $M \leftarrow \mathcal{D}(K', C)$
$\underline{\text{proc LR}(\phi, M_0, M_1)}$ // G_0
If $(|M_0| \neq |M_1|)$
  Return $\bot$
$K^* \leftarrow \phi(K)$
$\pi^* \leftarrow \mathcal{P}(K')$
Return $C^* \leftarrow\!\!\$\ \mathcal{E}(\pi^*, M_b)$
$\underline{\text{proc FINALIZE}(b')}$ // G_0
Return $(b = b')$

Figure 12: Game SE defining Φ-RKA-security of CCA-SE scheme $\mathcal{SE} = (\mathcal{K}, \mathcal{E}, \mathcal{D})$, algorithms defining CCA-SE scheme $\mathcal{SE} = (\mathcal{K}', \mathcal{E}', \mathcal{D}')$ from strong Φ-RKA secure CCA-PKE scheme $\mathcal{PKE} = (\mathcal{S}, \mathcal{P}, \mathcal{E}, \mathcal{D})$, and games for the proof of Theorem 8.3.

To complete the claim we need to argue that A' properly simulates G_2. The only subtlety is in how decryption queries are handled. In simulating decryption query responses the first two checks are exactly as in G_2. The check the KD oracle performs rejects the same ciphertexts as the last check in G_2, so the rest of DEC is properly simulated and so A' performs as claimed.

The proof is completed by collecting the relationships between games $\mathrm{G}_0, \mathrm{G}_1$ and G_2. ∎

Since strong Φ-RKA-security for IBE implies strong Φ-RKA-selective-ID security the CHK transform can be applied to the Waters and extended Waters IBE schemes under the identity renaming transform of Section 4 to obtain strong Φ-RKA-secure CCA-PKE for affine and polynomial ϕ with non-zero linear coefficients.

8.3 RKA-secure CCA-SE

A definition of Φ-RKA-security for a CCA-SE scheme is presented in Figure 12. It is a find-then-guess style definition where the adversary is allowed just one query to the LR oracle. This definition is equivalent to the left-or-right definition presented in [4], losing a factor of q in the reduction where q is the number of LR queries made in the same way as for standard CCA security. Figure 12 shows the natural way of constructing a symmetric encryption scheme from a public key encryption scheme, and Theorem 8.3 states that this construction applied to a strong Φ-RKA-secure CCA-PKE scheme gives a Φ-RKA-secure CCA-SE scheme.

Theorem 8.3 *Let $\mathcal{PKE} = (\mathcal{S}, \mathcal{P}, \mathcal{E}, \mathcal{D})$ be a strong Φ-RKA-secure CCA-PKE scheme. Then the CCA-SE scheme constructed as in Figure 12 is Φ-RKA secure.*

Proof: Game G_0 in Figure 12 implements SE^A with $\mathcal{SE}$, so we have

$$\mathbf{Adv}^{\mathrm{se-rka}}_{\mathcal{SE},\Phi}(A) = 2\Pr[\mathrm{G}_0^A] - 1.$$

It is straightforward to build from A an adversary A' against the strong Φ-RKA security of $\mathcal{PKE}$ such that

$$\mathbf{Adv}^{\mathrm{pke-srka}}_{\mathcal{PKE},\Phi}(A') = 2\Pr[\mathrm{G}^A_0] - 1.$$

The adversary simply runs A, forwarding all its queries to its own oracles. When A halts and outputs a bit b', A' does the same, winning precisely when A does. ▮

Constructing a symmetric encryption scheme in this way from the PKE scheme obtained by applying the CHK transform to the Waters and extended Waters IBE schemes under the identity renaming transform of Section 4 gives Φ-RKA-secure CCA-SE for affine and polynomial ϕ with non-zero linear coefficients. It is interesting to note here that the restriction on the RKD function polynomials that the scheme is secure against differs from that of [22]. In [22] a CPA-SE scheme RKA secure against RKD functions consisting of polynomials of non-zero degree is presented, while our CCA-SE scheme is RKA secure against the more restricted RKD set consisting of polynomials with non-zero linear coefficients.

References

[1] B. Applebaum. Garbling XOR gates "for free" in the standard model. Cryptology ePrint Archive, Report 2012/516, 2012. `http://eprint.iacr.org/`. (Cited on page 4.)

[2] B. Applebaum, D. Harnik, and Y. Ishai. Semantic security under related-key attacks and applications. In A. C.-C. Yao, editor, *ICS 2011*. Tsinghua University Press, 2011. (Cited on page 4, 5.)

[3] M. Bellare and D. Cash. Pseudorandom functions and permutations provably secure against related-key attacks. In T. Rabin, editor, *CRYPTO 2010*, volume 6223 of *LNCS*, pages 666–684. Springer, Aug. 2010. (Cited on page 3, 4, 5, 6, 7.)

[4] M. Bellare, D. Cash, and R. Miller. Cryptography secure against related-key attacks and tampering. In D. H. Lee and X. Wang, editors, *ASIACRYPT 2011*, volume 7073 of *LNCS*, pages 486–503. Springer, Dec. 2011. (Cited on page 3, 4, 5, 6, 7, 13, 15, 16, 24.)

[5] M. Bellare and T. Kohno. A theoretical treatment of related-key attacks: RKA-PRPs, RKA-PRFs, and applications. In E. Biham, editor, *EUROCRYPT 2003*, volume 2656 of *LNCS*, pages 491–506. Springer, May 2003. (Cited on page 3, 4, 6, 7.)

[6] M. Bellare and T. Ristenpart. Simulation without the artificial abort: Simplified proof and improved concrete security for Waters' IBE scheme. In A. Joux, editor, *EUROCRYPT 2009*, volume 5479 of *LNCS*, pages 407–424. Springer, Apr. 2009. (Cited on page 13, 14.)

[7] M. Bellare and P. Rogaway. The security of triple encryption and a framework for code-based game-playing proofs. In S. Vaudenay, editor, *EUROCRYPT 2006*, volume 4004 of *LNCS*, pages 409–426. Springer, May / June 2006. (Cited on page 6.)

[8] E. Biham. New types of cryptoanalytic attacks using related keys (extended abstract). In T. Helleseth, editor, *EUROCRYPT'93*, volume 765 of *LNCS*, pages 398–409. Springer, May 1993. (Cited on page 3.)

[9] E. Biham and A. Shamir. Differential fault analysis of secret key cryptosystems. In B. S. Kaliski Jr., editor, *CRYPTO'97*, volume 1294 of *LNCS*, pages 513–525. Springer, Aug. 1997. (Cited on page 3.)

[10] D. Boneh and X. Boyen. Efficient selective-ID secure identity based encryption without random oracles. In C. Cachin and J. Camenisch, editors, *EUROCRYPT 2004*, volume 3027 of *LNCS*, pages 223–238. Springer, May 2004. (Cited on page 16.)

[11] D. Boneh, X. Boyen, and E.-J. Goh. Hierarchical identity based encryption with constant size ciphertext. In R. Cramer, editor, *EUROCRYPT 2005*, volume 3494 of *LNCS*, pages 440–456. Springer, May 2005. (Cited on page 14.)

[12] D. Boneh, R. Canetti, S. Halevi, and J. Katz. Chosen-ciphertext security from identity-based encryption. *SIAM J. Comput.*, 36(5):1301–1328, 2007. (Cited on page 3, 5, 13, 15, 27, 35, 36.)

[13] D. Boneh, R. A. DeMillo, and R. J. Lipton. On the importance of checking cryptographic protocols for faults (extended abstract). In W. Fumy, editor, *EUROCRYPT'97*, volume 1233 of *LNCS*, pages 37–51. Springer, May 1997. (Cited on page 3.)

[14] D. Boneh and M. Franklin. Identity-based encryption from the Weil pairing. *SIAM J. Comput.*, 32(3):586–615, 2003. (Cited on page 3, 5, 8, 13, 15, 27.)

[15] D. Boneh and J. Katz. Improved efficiency for CCA-secure cryptosystems built using identity-based encryption. In A. Menezes, editor, *CT-RSA 2005*, volume 3376 of *LNCS*, pages 87–103. Springer, Feb. 2005. (Cited on page 5.)

[16] D. Boneh, B. Lynn, and H. Shacham. Short signatures from the Weil pairing. In C. Boyd, editor, *ASIACRYPT 2001*, volume 2248 of *LNCS*, pages 514–532. Springer, Dec. 2001. (Cited on page 13.)

[17] X. Boyen, Q. Mei, and B. Waters. Direct chosen ciphertext security from identity-based techniques. In V. Atluri, C. Meadows, and A. Juels, editors, *ACM CCS 05*, pages 320–329. ACM Press, Nov. 2005. (Cited on page 6, 15, 16, 17.)

[18] R. Cramer, Y. Dodis, S. Fehr, C. Padró, and D. Wichs. Detection of algebraic manipulation with applications to robust secret sharing and fuzzy extractors. In N. P. Smart, editor, *EUROCRYPT 2008*, volume 4965 of *LNCS*, pages 471–488. Springer, Apr. 2008. (Cited on page 5.)

[19] R. Cramer and V. Shoup. Design and analysis of practical public-key encryption schemes secure against adaptive chosen ciphertext attack. *IACR Cryptology ePrint Archive*, 2001:108, 2001. (Cited on page 15, 16.)

[20] R. Gennaro, A. Lysyanskaya, T. Malkin, S. Micali, and T. Rabin. Algorithmic tamper-proof (ATP) security: Theoretical foundations for security against hardware tampering. In M. Naor, editor, *TCC 2004*, volume 2951 of *LNCS*, pages 258–277. Springer, Feb. 2004. (Cited on page 4.)

[21] D. Goldenberg and M. Liskov. On related-secret pseudorandomness. In D. Micciancio, editor, *TCC 2010*, volume 5978 of *LNCS*, pages 255–272. Springer, Feb. 2010. (Cited on page 4, 7.)

[22] V. Goyal, A. O'Neill, and V. Rao. Correlated-input secure hash functions. In Y. Ishai, editor, *TCC 2011*, volume 6597 of *LNCS*, pages 182–200. Springer, Mar. 2011. (Cited on page 4, 5, 6, 7, 25.)

[23] S. Haber and B. Pinkas. Securely combining public-key cryptosystems. In *ACM CCS 01*, pages 215–224. ACM Press, Nov. 2001. (Cited on page 15.)

[24] L. R. Knudsen. Cryptanalysis of LOKI91. In J. Seberry and Y. Zheng, editors, *AUSCRYPT'92*, volume 718 of *LNCS*, pages 196–208. Springer, Dec. 1992. (Cited on page 3.)

[25] S. Lucks. Ciphers secure against related-key attacks. In B. K. Roy and W. Meier, editors, *FSE 2004*, volume 3017 of *LNCS*, pages 359–370. Springer, Feb. 2004. (Cited on page 7.)

[26] K. G. Paterson, J. C. N. Schuldt, M. Stam, and S. Thomson. On the joint security of encryption and signature, revisited. In D. H. Lee and X. Wang, editors, *ASIACRYPT 2011*, volume 7073 of *LNCS*, pages 161–178. Springer, Dec. 2011. (Cited on page 6, 15, 27, 29.)

[27] B. R. Waters. Efficient identity-based encryption without random oracles. In R. Cramer, editor, *EUROCRYPT 2005*, volume 3494 of *LNCS*, pages 114–127. Springer, May 2005. (Cited on page 3, 4, 5, 8, 13, 14.)

[28] H. Wee. Public key encryption against related key attacks. In M. Fischlin, J. Buchmann, and M. Manulis, editors, *Public Key Cryptography*, volume 7293 of *Lecture Notes in Computer Science*, pages 262–279. Springer, 2012. (Cited on page 4, 5, 7, 13, 15, 17.)

A RKA-secure schemes

For completeness, we provide in Figure 13 the RKA-secure IBE schemes that result from applying our framework to the Boneh-Franklin, Waters, and extended Waters IBE schemes.

Algorithm $\mathcal{P}(s)$:
$\pi \leftarrow g^s$
Return π

Algorithm $\overline{\mathcal{K}}(s, \overline{u})$:
$\overline{dk} \leftarrow H_1(\overline{u}||g^s)^s$
Return $\overline{dk}$

Algorithm $\overline{\mathcal{E}}(\pi, \overline{u}, M)$:
$t \leftarrow\$ \mathbb{Z}_p$
$C_1 \leftarrow g^t$
$C_2 \leftarrow H_2(e(\pi, H_1(\overline{u}||\pi))^t) \oplus M$
Return (C_1, C_2)

Algorithm $\mathcal{D}(\overline{dk}, C)$:
$M \leftarrow C_2 \oplus H_2(e(\overline{dk}, C_1))$
Return M

Algorithm $\mathcal{P}(s)$:
$\pi \leftarrow g^s$
Return π

Algorithm $\overline{\mathcal{K}}(s, \overline{u})$:
$r \leftarrow\$ \mathbb{Z}_p$
$\overline{dk}_1 \leftarrow g_1^s \cdot H(\overline{u}||g^s)^r$
$\overline{dk}_2 \leftarrow g^r$
Return $(\overline{dk}_1, \overline{dk}_2)$

Algorithm $\overline{\mathcal{E}}(\pi, \overline{u}, M)$:
$t \leftarrow\$ \mathbb{Z}_p$
$C_1 \leftarrow g^t$
$C_2 \leftarrow H(\overline{u}||\pi)^t$
$C_3 \leftarrow e(\pi, g_1)^t \cdot M$
Return (C_1, C_2, C_3)

Algorithm $\mathcal{D}(\overline{dk}, C)$:
$M \leftarrow C_3 \cdot \frac{e(\overline{dk}_2, C_2)}{e(\overline{dk}_1, C_1)}$
Return M

Algorithm $\mathcal{P}(s)$:
$\pi_0 \leftarrow g^s$
$\pi \leftarrow (\pi_0, g^{s^2}, \ldots, g^{s^d}, (g_1)^{s^2}, \ldots, (g_1)^{s^d})$
Return π

Algorithm $\overline{\mathcal{K}}(s, \overline{u})$:
$r \leftarrow\$ \mathbb{Z}_p$
$\overline{dk}_1 \leftarrow g_1^s \cdot H(\overline{u}||g^s)^r$
$\overline{dk}_2 \leftarrow g^r$
Return $(\overline{dk}_1, \overline{dk}_2)$

Algorithm $\overline{\mathcal{E}}(\pi, \overline{u}, M)$:
$t \leftarrow\$ \mathbb{Z}_p$
$C_1 \leftarrow g^t$
$C_2 \leftarrow H(\overline{u}||\pi_0)^t$
$C_3 \leftarrow e(\pi, g_1)^t \cdot M$
Return (C_1, C_2, C_3)

Algorithm $\mathcal{D}(\overline{dk}, C)$:
$M \leftarrow C_3 \cdot \frac{e(\overline{dk}_2, C_2)}{e(\overline{dk}_1, C_1)}$
Return M

Figure 13: Φ^{aff}-RKA-secure Boneh-Franklin IBE scheme on the left, Φ^{aff}-RKA-secure Waters IBE scheme in the middle, $\Phi^{\text{poly}(d)}$-RKA-secure extended Waters IBE scheme on the right.

B RKA-secure combined signature and encryption

The games associated with the notions of joint security for CSE schemes from [26] can be extended to Φ-RKA-security and are given in Figure 15. Game PKESO defines Φ-RKA-security of the encryption component of a CSE scheme in the presence of a signing oracle, with $\mathbf{Adv}^{\text{pkeso}}_{\mathcal{CSE},\Phi}(A) = 2\Pr[\text{PKESO}^A] - 1$. Game SigDO defines Φ-RKA-security of the signature component of a CSE scheme in the presence of a decryption oracle, with $\mathbf{Adv}^{\text{sigdo}}_{\mathcal{CSE},\Phi}(A) = \Pr[\text{SigDO}^A]$. When both of these are satisfied we say the scheme is Φ-RKA jointly secure (or has Φ-RKA joint security).

Paterson et al. [26] gave a construction of a secure CSE scheme from IBE using signatures obtained through the Naor transform [14] and public-key encryption obtained through the CHK transform [12]. The algorithms of this scheme are given in Figure 16.

Theorem B.1 *Let $\mathcal{CSE} = (\mathcal{K}', \mathcal{S}', \mathcal{V}', \mathcal{E}', \mathcal{D}')$ be the combined signature and encryption scheme constructed from Φ-RKA-selective-ID secure $\mathcal{IBE} = (\mathcal{S}, \mathcal{P}, \mathcal{K}, \mathcal{E}, \mathcal{D})$ and strongly unforgeable one time signature scheme $\overline{\mathcal{DS}} = (\overline{\mathcal{K}}, \overline{\mathcal{S}}, \overline{\mathcal{V}})$ as in Figure 16. Then the encryption component of $\mathcal{CSE}$ is Φ-RKA-secure in the presence of a signing oracle.*

Proof: The proof uses the sequence of games in Figure 17. Game G_0 implements PKESO^A with $\mathcal{CSE}$, so we have

$$\mathbf{Adv}^{\text{pkeso}}_{\mathcal{CSE},\Phi}(A) = 2\Pr[\text{G}_0^A] - 1.$$

Game G_1 has an additional check in the DEC oracle: now if $\overline{vk} = \overline{vk}^*$ and $(c, \overline{\sigma}) \neq (c^*, \overline{\sigma}^*)$, the game sets bad and responds to the query with $\perp$. Since G_1 and G_0 are identical until bad, we have

$$\Pr[\text{G}_1^A] - \Pr[\text{G}_0^A] \leq \Pr[E_1],$$

proc INITIALIZE // sIBE
$s \leftarrow\$ \mathcal{S}$; $\pi \leftarrow \mathcal{P}(s)$
$b \leftarrow\$ \{0,1\}$; $u^* \leftarrow \perp$

proc SID(u) // sIBE
$u^* \leftarrow u$
Return π

proc KD(ϕ, u) // sIBE
If ($u^* = \perp$) then return $\perp$
$s' \leftarrow \phi(s)$
If ($s' = s \wedge u = u^*$) Return $\perp$
Return $dk \leftarrow\$ \mathcal{K}(s', u)$

proc LR(M_0, M_1) // sIBE
If ($|M_0| \neq |M_1|$) Return $\perp$
If ($u^* = \perp$) Return $\perp$
Return $C \leftarrow\$ \mathcal{E}(\pi, u^*, M_b)$

proc FINALIZE(b') // sIBE
Return ($b = b'$)

proc INITIALIZE // OWIBE
$s \leftarrow\$ \mathcal{S}$; $\pi \leftarrow \mathcal{P}(s)$
$b \leftarrow\$ \{0,1\}$; $u^* \leftarrow \perp$; $I \leftarrow \emptyset$
Return π

proc KD(ϕ, u) // OWIBE
$s' \leftarrow \phi(s)$
If ($s' = s$) $I \leftarrow I \cup \{u\}$
If ($u^* \in I$) Return $\perp$
Return $dk \leftarrow\$ \mathcal{K}(s', u)$

proc SID(u) // OWIBE
$u^* \leftarrow u$
If ($u^* \in I$) Return $\perp$
$M \leftarrow\$ \mathsf{MSp}$
Return $C \leftarrow\$ \mathcal{E}(\pi, u^*, M)$

proc FINALIZE(M') // OWIBE
Return ($M = M'$)

Figure 14: Games sIBE, OWIBE defining various flavors of Φ-RKA-security of IBE scheme $\mathcal{IBE} = (\mathcal{S}, \mathcal{P}, \mathcal{K}, \mathcal{E}, \mathcal{D})$.

where E_1 is the event that G_1 sets bad. We now construct an adversary B that breaks the strong unforgeability of $\overline{\mathcal{DS}}$ with probability $\Pr[E_1]$. B is given as input $\overline{vk}^*$ and starts by selecting $b \leftarrow\$ \{0,1\}$, $s \leftarrow\$ \mathcal{S}$ and computing $\pi \leftarrow \mathcal{P}(s)$. It runs $A(\pi)$ and simulates the response to the LR query by computing

If ($|M_0| \neq |M_1|$) Return $\perp$; $c^* \leftarrow\$ \mathcal{E}(\pi, 1||\overline{vk}^*, M_b)$;
$\overline{\sigma}^* \leftarrow\$ \overline{\mathcal{S}}(\overline{sk}^*, c^*)$; $C^* \leftarrow (c^*, \overline{vk}^*, \overline{\sigma}^*)$; Return C^*

using its own signing oracle to generate $\overline{\sigma}^*$ on c^*. It also simulates DEC and SIGN query responses exactly as specified in G_1, noting the first query that triggers bad, if any. If there is such a query DEC($\phi, c, \overline{vk}, \overline{\sigma}$), then B outputs $(c, \overline{\sigma})$ as its forgery. This is a valid forgery for B because this query must have passed the validity check $\overline{\mathcal{V}}(\overline{vk}, c, \overline{\sigma})$ (otherwise the oracle would have returned before going to set bad), and we have that $(c, \overline{\sigma}) \neq (c^*, \overline{\sigma}^*)$ by the check immediately before bad is set.

Game G_2 rearranges some of the validity checks in DEC oracle processing, but these changes don't affect oracle responses. This change is valid because the same ciphertexts end up being rejected in either game. We have

$$\Pr[G_2^A] = \Pr[G_1^A] \ .$$

We are now in a position to show that an adversary winning G_2 with good probability can be used to break the Φ-RKA-selective-ID security of $\mathcal{IBE}$. We construct A' such that

$$\mathbf{Adv}^{\text{sibe-rka}}_{\mathcal{IBE},\Phi}(A') = 2\Pr[G_2^A] - 1.$$

A' generates $(\overline{vk}^*, \overline{sk}^*) \leftarrow\$ \overline{\mathcal{K}}$ and calls its SID oracle with $1||\overline{vk}^*$, receiving π in return. A' runs $A(\pi)$, simulating the response to $\text{LR}(M_0, M_1)$ by querying its own oracle for $c^* \leftarrow \text{LR}(M_0, M_1)$, and then signing c^* using $\overline{sk}^*$, finally returning the ciphertext $C^* = (c^*, \overline{vk}^*, \overline{\sigma}^*)$. A' simulates responses to SIGN(ϕ, M) by querying its oracle $\overline{\sigma} \leftarrow \text{KD}(\phi, 0||M)$, and responses to DEC($\phi, (c, \overline{vk}, \overline{\sigma})$) by computing

proc INITIALIZE // PKESO
$b \leftarrow\$ \{0,1\}$; $C^* \leftarrow \perp$
$(pk, sk) \leftarrow\$ \mathcal{K}$
Return pk

proc DEC(ϕ, C) // PKESO
$sk' \leftarrow \phi(sk)$
If $((sk' = sk) \wedge (C = C^*))$ Return $\perp$
Return $M \leftarrow \mathcal{D}(sk', C)$

proc LR(M_0, M_1) // PKESO
If $(|M_0| \neq |M_1|)$ Return $\perp$
Return $C^* \leftarrow\$ \mathcal{E}(pk, M_b)$

proc SIGN(ϕ, M) // PKESO
$sk' \leftarrow \phi(sk)$
Return $\sigma \leftarrow\$ \mathcal{S}(sk', M)$

proc FINALIZE(b') // PKESO
Return $(b = b')$

proc INITIALIZE // SigDO
$S \leftarrow \emptyset$
$(pk, sk) \leftarrow\$ \mathcal{K}$
Return pk

proc SIGN(ϕ, M) // SigDO
$sk' \leftarrow \phi(sk)$
If $(sk' = sk)$ $S \leftarrow S \cup \{M\}$
Return $\sigma \leftarrow\$ \mathcal{S}(sk', M)$

proc DEC(ϕ, C) // SigDO
$sk' \leftarrow \phi(sk)$
Return $M \leftarrow \mathcal{D}(sk', C)$

proc FINALIZE(M, σ) // SigDO
Return $((\mathcal{V}(pk, M, \sigma) = 1) \wedge (M \notin S))$

Figure 15: Game PKESO defining Φ-RKA-security for the encryption component of a CSE scheme in the presence of a signing oracle, and game SigDO defining Φ-RKA-security for the signature component of a CSE scheme in the presence of a decryption oracle.

Algorithm $\mathcal{K}'$:
$s \leftarrow\$ \mathcal{S}$
$\pi \leftarrow \mathcal{P}(s)$
Return (s, π)

Algorithm $\mathcal{S}'(s, M)$:
$\sigma \leftarrow\$ \mathcal{K}(s, 0||M)$
Return σ

Algorithm $\mathcal{V}'(\pi, \sigma, M)$:
$x \leftarrow\$ \mathsf{MSp}$
$c \leftarrow\$ \mathcal{E}(\pi, 0||M, x)$
If $(\mathcal{D}(\sigma, c) = x)$ Return 1
Return 0

Algorithm $\mathcal{E}'(\pi, M)$:
$(\overline{vk}, \overline{sk}) \leftarrow\$ \overline{\mathcal{K}}$
$c \leftarrow\$ \mathcal{E}(\pi, 1||\overline{vk}, M)$
$\overline{\sigma} \leftarrow\$ \overline{\mathcal{S}}(\overline{sk}, c)$
$C \leftarrow (c, \overline{vk}, \overline{\sigma})$
Return C

Algorithm $\mathcal{D}'(s, (c, \overline{vk}, \overline{\sigma}))$:
If $(\overline{\mathcal{V}}(\overline{vk}, c, \overline{\sigma}) = 0)$ Return $\perp$
$dk \leftarrow\$ \mathcal{K}(s, 1||\overline{vk})$
$M \leftarrow \mathcal{D}(dk, c)$
Return M

Figure 16: Combined signature and encryption scheme from IBE [26].

If $((\overline{\mathcal{V}}(\overline{vk}, c, \overline{\sigma}) = 0) \vee ((\overline{vk} = \overline{vk}^*) \wedge ((c, \overline{\sigma}) \neq (c^*, \overline{\sigma}^*))))$ Return $\perp$;
$dk \leftarrow \mathrm{KD}(\phi, 1||\overline{vk})$; $M \leftarrow \mathcal{D}(dk, c)$; Return M.

A' runs A until it halts, and outputs whatever A outputs.

To complete the claim we need to argue that A' properly simulates G_2. The only subtlety is in how decryption and signature queries are handled. In simulating decryption query responses the first two checks are exactly as in G_2. The check the KD oracle performs rejects the same ciphertexts as the last check in G_2, so the rest of DEC is properly simulated. The differing bits prepended to identities in encryption and signature ensure the KD oracle's check will never be true in a signing query, so SIGN is properly simulated and so A' performs as claimed.

The proof is completed by collecting the relationships between games $\mathrm{G}_0, \mathrm{G}_1$ and G_2. ∎

$\underline{\text{proc INITIALIZE}}$ // $\mathrm{G}_0, \mathrm{G}_1, \mathrm{G}_2$
$b \leftarrow\!\!\$\ \{0,1\}$; $C^* \leftarrow \perp$; $S \leftarrow \emptyset$
$s \leftarrow\!\!\$\ \mathcal{S}$; $\pi \leftarrow \mathcal{P}(s)$
Return π

$\underline{\text{proc LR}(M_0, M_1)}$ // $\mathrm{G}_0, \mathrm{G}_1, \mathrm{G}_2$
If $(|M_0| \neq |M_1|)$ Return $\perp$
$(\overline{vk}^*, \overline{sk}^*) \leftarrow\!\!\$\ \overline{\mathcal{K}}$
$c^* \leftarrow\!\!\$\ \mathcal{E}(\pi, 1||\overline{vk}^*, M_b)$
$\overline{\sigma}^* \leftarrow\!\!\$\ \overline{\mathcal{S}}(\overline{sk}^*, c^*)$
$C^* \leftarrow (c^*, \overline{vk}^*, \overline{\sigma}^*)$
Return C^*

$\underline{\text{proc FINALIZE}(b')}$ // $\mathrm{G}_0, \mathrm{G}_1, \mathrm{G}_2$
Return $(b = b')$

$\underline{\text{proc SIGN}(\phi, M)}$ // $\mathrm{G}_0, \mathrm{G}_1, \mathrm{G}_2$
$s' \leftarrow \phi(s)$
$\sigma \leftarrow\!\!\$\ \mathcal{K}(s', 0||M)$
Return σ

$\underline{\text{proc DEC}(\phi, (c, \overline{vk}, \overline{\sigma}))}$ // $\mathrm{G}_0, \boxed{\mathrm{G}_1}$
$s' \leftarrow \phi(s)$
If $((s' = s) \wedge ((c, \overline{vk}, \overline{\sigma}) = C^*))$
  Return $\perp$
If $(\overline{\mathcal{V}}(\overline{vk}, c, \overline{\sigma}) = 0)$
  Return $\perp$
If $((\overline{vk} = \overline{vk}^*) \wedge ((c, \overline{\sigma}) \neq (c^*, \overline{\sigma}^*)))$
  $\mathsf{bad} \leftarrow \mathsf{true}$; $\boxed{\text{return } \perp}$
$dk \leftarrow\!\!\$\ \mathcal{K}(s', 1||\overline{vk})$
Return $M \leftarrow \mathcal{D}(dk, c)$

$\underline{\text{proc DEC}(\phi, (c, \overline{vk}, \overline{\sigma}))}$ // G_2
If $(\overline{\mathcal{V}}(\overline{vk}, c, \overline{\sigma}) = 0)$
  Return $\perp$
If $((\overline{vk} = \overline{vk}^*) \wedge ((c, \overline{\sigma}) \neq (c^*, \overline{\sigma}^*)))$
  Return $\perp$
$s' \leftarrow \phi(s)$
If $((s' = s) \wedge (\overline{vk} = \overline{vk}^*))$
  Return $\perp$
$dk \leftarrow \mathcal{K}(s', 1||\overline{vk})$
Return $M \leftarrow \mathcal{D}(dk, c)$

Figure 17: Games for the proof of Theorem B.1.

$\underline{\text{proc INITIALIZE}}$ // G_3
$b \leftarrow\!\!\$\ \{0,1\}$; $C^* \leftarrow \perp$; $S \leftarrow \emptyset$
$s \leftarrow\!\!\$\ \mathcal{S}$; $\pi \leftarrow \mathcal{P}(s)$
Return π

$\underline{\text{proc FINALIZE}(M, \sigma)}$ // G_3
$x \leftarrow\!\!\$\ \mathcal{M}$
$c \leftarrow\!\!\$\ \mathcal{E}(\pi, 0||M, x)$
Return $((\mathcal{D}(\sigma, c) = x) \wedge (0||M \notin S))$

$\underline{\text{proc DEC}(\phi, (c, \overline{vk}, \overline{\sigma}))}$ // G_3
$s' \leftarrow \phi(s)$
If $(\overline{\mathcal{V}}(\overline{vk}, c, \overline{\sigma}) = 0)$
  Return $\perp$
$dk \leftarrow\!\!\$\ \mathcal{K}(s', 1||\overline{vk})$
Return $M \leftarrow \mathcal{D}(dk, c)$

$\underline{\text{proc SIGN}(\phi, M)}$ // G_3
$s' \leftarrow \phi(s)$
If $(s' = s)$ $S \leftarrow S \cup \{0||M\}$
Return $\sigma \leftarrow\!\!\$\ \mathcal{K}(s, 0||M)$

Figure 18: Games for the proof of Theorem B.2.

Theorem B.2 *Let $\mathcal{CSE} = (\mathcal{K}', \mathcal{S}', \mathcal{V}', \mathcal{E}', \mathcal{D}')$ be the combined signature and encryption scheme constructed from Φ-RKA-secure $\mathcal{IBE} = (\mathcal{S}, \mathcal{P}, \mathcal{K}, \mathcal{E}, \mathcal{D})$ and strongly unforgeable one time signature scheme $\overline{\mathcal{DS}} = (\overline{\mathcal{K}}, \overline{\mathcal{S}}, \overline{\mathcal{V}})$ as in Figure 16. Then the signature component of $\mathcal{CSE}$ is Φ-RKA-secure in the presence of a decryption oracle.*

Proof: The proof uses the game in Figure 18. Game G_3 implements SigDO^A with $\mathcal{CSE}$, so we have

$$\mathbf{Adv}^{\text{sigdo}}_{\mathcal{CSE},\Phi}(A) = \Pr[\mathrm{G}_3^A].$$

We now show that an adversary winning G_3 with good probability can be used to break the Φ-RKA one-way security of $\mathcal{IBE}$. We construct A' such that

$$\mathbf{Adv}^{\text{owibe-rka}}_{\mathcal{IBE},\Phi}(A') = \Pr[\mathrm{G}_3^A].$$

A' takes input π and runs $A(\pi)$. A' simulates the response to signing oracle queries $\text{SIGN}(\phi, M)$ with a call to $\mathrm{KD}(\phi, 0||M)$. The restriction that a user private key corresponding to the challenge identity cannot be requested ensures no signature will be generated for the message output as a forgery. A' simulates the response to decryption oracle queries $\text{DEC}(\phi, (c, \overline{vk}, \overline{\sigma}))$ by computing

$$\text{If } (\overline{\mathcal{V}}(\overline{vk}, c, \overline{\sigma}) = 0) \text{ Return } \perp ;\ dk \leftarrow \mathrm{KD}(\phi, 1||\overline{vk})\ ;\ M \leftarrow \mathcal{D}(dk, c)\ ;\ \text{Return } M.$$

The bit prefix ensures this call to KD does not request a key for the challenge identity.

When A halts and returns a forgery σ^* on a message M^*, A' makes a call to its SID oracle with identity $0||M^*$. If a signature has already been issued for M^* under the original master secret key then $0||M^*$

proc INITIALIZE // PKE	proc INITIALIZE // OT-CCA
$b \leftarrow\!\!\$\ \{0,1\}\ ;\ C^* \leftarrow \bot$	$b \leftarrow\!\!\$\ \{0,1\}\ ;\ C^* \leftarrow \bot$
$(pk, sk) \leftarrow\!\!\$\ \mathcal{K}$	$K \leftarrow\!\!\$\ \mathcal{K}$
Return pk	
proc DEC(ϕ, C) // PKE	proc DEC(C) // OT-CCA
$sk' \leftarrow \phi(sk)$	If $(C^* = \bot)$
If $((sk' = sk) \wedge (C = C^*))$	Return $\bot$
Return $\bot$	If $(C = C^*)$
Return $M \leftarrow \mathcal{D}(sk', C)$	Return $\bot$
	Return $M \leftarrow \mathcal{D}(K, C)$
proc LR(M_0, M_1) // PKE	proc LR(M_0, M_1) // OT-CCA
If $(\lvert M_0\rvert \neq \lvert M_1\rvert)$	If $(\lvert M_0\rvert \neq \lvert M_1\rvert)$
Return $\bot$	Return $\bot$
Return $C^* \leftarrow\!\!\$\ \mathcal{E}(pk, M_b)$	Return $C^* \leftarrow\!\!\$\ \mathcal{E}(K, M_b)$
proc FINALIZE(b') // PKE	proc FINALIZE(b') // OT-CCA
Return $(b = b')$	Return $(b = b')$

Figure 19: Game PKE defining Φ-RKA-security for CCA-PKE and game OT-CCA defining one-time-CCA security for SE.

Algorithm $\mathcal{K}$:	Algorithm $\mathcal{E}(ek, M)$:	Algorithm $\mathcal{D}(dk, C)$:
$(ek, dk) \leftarrow\!\!\$\ \mathcal{KEM}.\mathcal{K}$	$(C_1, K) \leftarrow\!\!\$\ \mathcal{KEM}.\mathcal{E}(ek)$	$K \leftarrow \mathcal{KEM}.\mathcal{D}(dk, C_1)$
Return ek	$C_2 \leftarrow\!\!\$\ \mathcal{SE}.\mathcal{E}(K, M)$	Return $M \leftarrow \mathcal{SE}.\mathcal{D}(K, C_2)$
	Return $C^* \leftarrow (C_1, C_2)$	

Figure 20: Algorithms for the hybrid encryption scheme $\mathcal{HPKE}$

will be on the list of identities A' requested secret keys for under the original master secret key. In this case SID will return $\bot$ and A' will halt without output. If no signature has been issued for M^* then SID will return an encryption of a random message under identity $0||M^*$ which A' will attempt to decrypt using σ^* as the secret key. If σ^* is a valid signature on $0||M^*$ this decryption will be successful and A' will output the resulting message. A' succeeds in outputting the correct message precisely when A makes a valid forgery, and the theorem follows. ∎

C Proof of Theorem 7.1

Proof: The proof uses the sequence of games in Figure 21. Game G_0 implements PKE^A with $\mathcal{HPKE}$. Game G_1 is as game G_0, except that in game G_1 if the adversary submits for decryption a ciphertext whose first component is C_1^* and a ϕ such that $\phi(dk) = dk$, the key K^* generated in the LR query is used directly rather than obtaining this key through decapsulating C_1^* under key dk. The behavior of the games is identical, so we have that

$$\Pr[\mathrm{G}_0^A] = \Pr[\mathrm{G}_1^A] \ .$$

Game G_2 is as game G_1 except that the key K^* encapsulated by the first component of the challenge ciphertext is replaced by a random key K'. We now construct an adversary B against the Φ-RKA-security of $\mathcal{KEM}$ such that

$$\left|\Pr[\mathrm{G}_1^A] - \Pr[\mathrm{G}_2^A]\right| = \mathbf{Adv}^{\text{kem-rka}}_{\mathcal{KEM},\Phi}(B) \ .$$

B is given a key pair for $\mathcal{KEM}$. It chooses a random bit σ and passes the keypair to A. When A makes a decryption query, B passes ϕ and the first component of the ciphertext to its decapsulation oracle, obtaining in response a key which it uses to decrypt the symmetric component of the ciphertext, returning the result to A. When A makes its LR query, B calls its CHALLENGE oracle to obtain (C_1^*, K^*), then computes $C_2^* \leftarrow \mathcal{SE}.\mathcal{E}(K^*, M_\sigma)$ and passes (C_1^*, C_2^*) to A. B handles A's decryption queries as before, except in the case where $C_1 = C_1^*$. In this case B forwards ϕ, C_1^* to its decapsulation oracle. If the response to this query is a key K then $\phi(dk) \neq dk$ and B uses K to decrypt the symmetric component and returns the result to A. If instead the response is $\perp$ then $\phi(dk) = dk$. B checks whether $C_2 = C_2^*$, returning $\perp$ if so, decrypting the symmetric component with the key K^* it received in response to its CHALLENGE query and returning the result to A if not. Eventually A will halt and output a bit σ'. If A has guessed correctly $\sigma' = \sigma$, B outputs $b' = 1$, else B outputs $b' = 0$. When the challenge bit b in B's game is 1, the key K^* returned by the CHALLENGE oracle is the real key, so A is provided with a perfect simulation of game G_1, while when B's challenge bit is 0, K^* is a random key so A is playing game G_2. This gives us that

$$\begin{aligned}
\mathbf{Adv}^{\text{kem-rka}}_{\mathcal{KEM},\Phi}(B) &= \left|\Pr[b' = 1 | b = 1] - \Pr[b' = 1 | b = 0]\right| \\
&= \left|\Pr[\sigma' = \sigma | b = 1] - \Pr[\sigma' = \sigma | b = 0]\right| \\
&= \left|\Pr[\mathrm{G}_1^A] - \Pr[\mathrm{G}_2^A]\right| \ ,
\end{aligned}$$

as required.

We finally construct an adversary C against the one-time-CCA security of $\mathcal{SE}$ such that

$$\mathbf{Adv}^{\text{se-ot-cca}}_{\mathcal{SE}}(C) = \left|\Pr[\mathrm{G}_2^A] - 1/2\right| .$$

C begins by generating a keypair $(ek, dk) \leftarrow\!\!\$\ \mathcal{KEM}.\mathcal{K}$ which it passes to A. A makes decryption queries which C handles by computing $dk' \leftarrow \phi(dk)$ and using dk' to decapsulate the first ciphertext component, then decrypting the second component with the session key, passing the result to A. When A calls its LR oracle with messages M_0, M_1, C calls its own LR oracle with the same messages, receiving C_2^* in response. B then generates $(C_1^*, K^*) \leftarrow\!\!\$\ \mathcal{KEM}.\mathcal{E}(ek)$ and passes (C_1^*, C_2^*) to A. A makes further decryption queries to which C responds as before, except when $\phi(dk) = dk$ and the first component of the ciphertext is C_1^*. If the ciphertext submitted for decryption is C^* then C returns $\perp$. If instead the first component of the ciphertext is C_1^* and $C_2 \neq C_2^*$, C calls its decryption oracle with ciphertext C_2 and returns the response to A. Eventually A halts outputting a bit b' which C forwards as its own output. C provides a perfect simulation of G_2 for A and correctly determines the challenge bit with the same probability as A, so the above equation holds.

Collecting the equations gives us

$$\begin{aligned}
\mathbf{Adv}^{\text{pke-rka}}_{\mathcal{HPKE},\Phi}(A) &= \left|\Pr[\mathrm{G}_0^A] - 1/2\right| \\
&= \left|\Pr[\mathrm{G}_1^A] - 1/2\right| \\
&= \left|\Pr[\mathrm{G}_1^A] - \Pr[\mathrm{G}_2^A] + \Pr[\mathrm{G}_2^A] - 1/2\right| \\
&\leq \mathbf{Adv}^{\text{kem-rka}}_{\mathcal{KEM},\Phi}(B) + \mathbf{Adv}^{\text{se-ot-cca}}_{\mathcal{SE}}(C) \ ,
\end{aligned}$$

establishing the theorem. ∎

D Proof of Theorem 7.2

Proof: The proof uses the sequence of games in Figure 22. Game G_0 implements KEM^A with $\mathcal{KEM}$. Game G_1 moves the generation of the challenge ciphertext and key to the initialize phase. It is identical

$\underline{\text{proc INITIALIZE}}$ // $\mathrm{G}_0, \mathrm{G}_1, \mathrm{G}_2$
$b \leftarrow\!\!\$\ \{0,1\}$; $C^* \leftarrow \bot$
$(ek, dk) \leftarrow\!\!\$\ \mathcal{KEM}.\mathcal{K}$
Return ek

$\underline{\text{proc DEC}(\phi, C)}$ // G_0
$dk' \leftarrow \phi(dk)$
If $((dk' = dk) \wedge (C = C^*))$
  Return $\bot$
$K \leftarrow \mathcal{KEM}.\mathcal{D}(dk', C_1)$
Return $M \leftarrow \mathcal{SE}.\mathcal{D}(K, C_2)$

$\underline{\text{proc LR}(M_0, M_1)}$ // $\mathrm{G}_0, \mathrm{G}_1$
If $(|M_0| \neq |M_1|)$
  Return $\bot$
$(C_1^*, K^*) \leftarrow\!\!\$\ \mathcal{KEM}.\mathcal{E}(ek)$
$C_2^* \leftarrow\!\!\$\ \mathcal{SE}.\mathcal{E}(K^*, M_b)$
Return $C^* \leftarrow (C_1^*, C_2^*)$

$\underline{\text{proc FINALIZE}(b')}$ // $\mathrm{G}_0, \mathrm{G}_1, \mathrm{G}_2$
Return $(b = b')$

$\underline{\text{proc DEC}(\phi, C)}$ // G_1
$dk' \leftarrow \phi(dk)$
If $((dk' = dk) \wedge (C = C^*))$
  Return $\bot$
If $((dk' = dk) \wedge C_1 = C_1^*)$
  Return $M \leftarrow \mathcal{SE}.\mathcal{D}(K^*, C_2)$
$K \leftarrow \mathcal{KEM}.\mathcal{D}(dk', C_1)$
Return $M \leftarrow \mathcal{SE}.\mathcal{D}(K, C_2)$

$\underline{\text{proc DEC}(\phi, C)}$ // G_2
$dk' \leftarrow \phi(dk)$
If $((dk' = dk) \wedge (C = C^*))$
  Return $\bot$
If $((dk' = dk) \wedge C_1 = C_1^*)$
  Return $M \leftarrow \mathcal{SE}.\mathcal{D}(K', C_2)$
$K \leftarrow \mathcal{KEM}.\mathcal{D}(dk', C_1)$
Return $M \leftarrow \mathcal{SE}.\mathcal{D}(K, C_2)$

$\underline{\text{proc LR}(M_0, M_1)}$ // G_2
If $(|M_0| \neq |M_1|)$
  Return $\bot$
$(C_1^*, K^*) \leftarrow\!\!\$\ \mathcal{KEM}.\mathcal{E}(ek)$
$K' \leftarrow\!\!\$\ \mathcal{SE}.\mathsf{Ksp}$
$C_2^* \leftarrow\!\!\$\ \mathcal{SE}.\mathcal{E}(K', M_b)$
Return $C^* \leftarrow (C_1^*, C_2^*)$

Figure 21: Games used in the proof of Theorem 7.1.

to game G_0 from the adversary's point of view, unless the adversary makes a decapsulation query with $\phi(dk) = dk$ for a valid ciphertext with the same random value t as the challenge ciphertext before it has made a challenge query. We call this event E_1, and argue that its probability is q/p, where q is the number of decapsulation queries the adversary makes before making a challenge query, since until a challenge query is made no information is leaked about t. By the fundamental lemma of game playing we have that

$$\begin{aligned}
\mathbf{Adv}^{\text{kem-rka}}_{\mathcal{KEM},\Phi}(A) &= 2\Pr[\mathrm{G}_0^A] - 1 \\
&= 2(\Pr[\mathrm{G}_0^A] - \Pr[\mathrm{G}_1^A] + \Pr[\mathrm{G}_1^A]) - 1 \\
&\leq 2(\Pr[\mathrm{G}_1^A] + \Pr[E_1]) - 1 \\
&\leq 2(\Pr[\mathrm{G}_1^A] + q/p) - 1 \ .
\end{aligned}$$

Game G_2 is as game G_1, but sets the flag bad and returns $\bot$ if the adversary makes a decapsulation query for a valid ciphertext with the same hash value as the challenge ciphertext. Games G_1 and G_2 are identical from the adversary's point of view until this event occurs, so we have that

$$\Pr[\mathrm{G}_1^A] - \Pr[\mathrm{G}_2^A] \leq \Pr[E_2] \ ,$$

where E_2 is the event that G_2 sets bad. We construct an adversary B against the collision resistance of H such that

$$\Pr[E_2] = \mathbf{Adv}^{\text{cr}}_H(B) \ .$$

B is given the key k for the hash function H and runs $\mathcal{K}$ to get a keypair ek, dk for the KEM. It chooses $t \leftarrow\!\!\$\ \mathbb{Z}_p$ and computes $C_1^* \leftarrow g^t$, $w^* \leftarrow H_k(g^t || e(g,h)^{dk})$, $C_2^* \leftarrow u_1^t u_2^{tw^*}$, and $K^* \leftarrow e(g,h)^{(dk)t}$. B chooses a random bit $b \leftarrow\!\!\$\ \{0,1\}$, then runs A, supplying ek as the public key. When A makes a decapsulation query B acts exactly as game G_2's decapsulation oracle, which it is able to do since it knows dk. When A makes a challenge query B responds with $((C_1^*, C_2^*), K^*)$, first setting $K^* \leftarrow\!\!\$\ \mathsf{Ksp}$ if $b = 0$. Eventually

proc INITIALIZE // G_0
$dk \leftarrow\!\!\$\ \mathbb{Z}_p$
$Z \leftarrow e(g,h)^{dk}$
$y_1, y_2 \leftarrow\!\!\$\ \mathbb{Z}_p$
$u_1 \leftarrow h^{y_1}$; $u_2 \leftarrow h^{y_2}$
$ek \leftarrow (Z, u_1, u_2)$
$b \leftarrow\!\!\$\ \{0,1\}$; $C^* \leftarrow \perp$
Return ek

proc DEC(ϕ, C) // $\mathrm{G}_0, \mathrm{G}_1$
$dk' \leftarrow \phi(dk)$
If $(dk' = dk \wedge C = C^*)$
  Return $\perp$
$w \leftarrow H_k(C_1 || e(g,h)^{dk'})$
If $(e(C_1, u_1 u_2^w) \neq e(g, C_2))$
  Return $\perp$
Return $e(C_1, h)^{dk'}$

proc CHALLENGE() // G_0
$t \leftarrow\!\!\$\ \mathbb{Z}_p$
$C_1^* \leftarrow g^t$
$w^* \leftarrow H_k(C_1 || Z)$
$C_2^* \leftarrow u_1^t u_2^{tw^*}$
$K^* \leftarrow Z^t$
If $(b = 0)$ $K^* \leftarrow\!\!\$\ \mathsf{Ksp}$
Return $((C_1^*, C_2^*), K^*)$

proc FINALIZE(b') // $\mathrm{G}_0, \mathrm{G}_1, \mathrm{G}_2$
Return $(b = b')$

proc INITIALIZE // $\mathrm{G}_1, \mathrm{G}_2$
$dk \leftarrow\!\!\$\ \mathbb{Z}_p$
$Z \leftarrow e(g,h)^{dk}$
$y_1, y_2 \leftarrow\!\!\$\ \mathbb{Z}_p$
$u_1 \leftarrow h^{y_1}$; $u_2 \leftarrow h^{y_2}$
$ek \leftarrow (Z, u_1, u_2)$
$b \leftarrow\!\!\$\ \{0,1\}$
$t \leftarrow\!\!\$\ \mathbb{Z}_p$
$C_1^* \leftarrow g^t$
$w^* \leftarrow H_k(C_1 || Z)$
$C_2^* \leftarrow u_1^t u_2^{tw^*}$
$K^* \leftarrow Z^t$
Return ek

proc DEC(ϕ, C) // G_2
$dk' \leftarrow \phi(dk)$
If $(dk' = dk \wedge C = C^*)$
  Return $\perp$
$w \leftarrow H_k(C_1 || e(g,h)^{dk'})$
If $(e(C_1, u_1 u_2^w) \neq e(g, C_2))$
  Return $\perp$
If $(w = w*)$
  $\mathsf{bad} \leftarrow \mathsf{true}$; Return $\perp$
Return $e(C_1, h)^{dk'}$

proc CHALLENGE() // $\mathrm{G}_1, \mathrm{G}_2$
If $(b = 0)$ $K^* \leftarrow\!\!\$\ \mathsf{Ksp}$
Return $((C_1^*, C_2^*), K^*)$

Figure 22: Games $\mathrm{G}_0, \mathrm{G}_1, \mathrm{G}_2$ used in proving the Φ^{aff}-RKA-security of $\mathcal{BMW}$-$\mathcal{KEM}$.

A halts and outputs a bit b'. If event E_2 occurred then there was a query (ϕ, C) with $w = w^*$, where $C = (C_1, C_2)$ is a valid ciphertext so is of the form $(g^x, u_1^x u_2^{xw})$ where $w = H_k(g^x || e(g,h)^{\phi(dk)})$. Then we have a pair $m = g^t || e(g,h)^{dk}$, $m' = g^x || e(g,h)^{\phi(dk)}$ with $H_k(m) = H_k(m')$. To show this is a collision it remains to show that $m \neq m'$. If $m = m'$ then $e(g,h)^{dk} = e(g,h)^{\phi(dk)}$, so $\phi(dk) = dk$ (since $\mathbb{G}_T$ is of prime order and $e(g,h) \neq 1_{\mathbb{G}_T}$). Since C is valid, it follows from the decapsulation oracle's second check that if it has $C_1^* = C_1$, then it must also have $C_2^* = C_2$. But by the first check performed by the decapsulation, we have that $C^* \neq C$, so the ciphertexts must differ in the first component. Then $g^t \neq g^x$, contradicting the assumption that $m = m'$. We deduce that m and m' are not equal. B outputs (m, m') and wins the game with probability $\Pr[E_2]$.

We now show that an adversary winning G_2 with good probability can be used to solve the DBDH problem. We build C such that

$$\mathbf{Adv}^{\mathrm{dbdh}}(C) = 2\Pr[\mathrm{G}_2^A] - 1 \ .$$

C is given as input a DBDH challenge tuple $(g, g^x, g^y, g^z, h, h^x, h^y, h^z, T)$. C computes $Z \leftarrow e(g^x, h^y)$ so that the private key dk is the unknown value xy, and then computes $w^* \leftarrow H_k(g^z || Z)$. It then chooses $\delta \leftarrow\!\!\$\ \mathbb{Z}_p$ and sets

$$u_1 \leftarrow (h^x)^{-w^*} h^\delta \qquad \text{and} \qquad u_2 \leftarrow h^x \ .$$

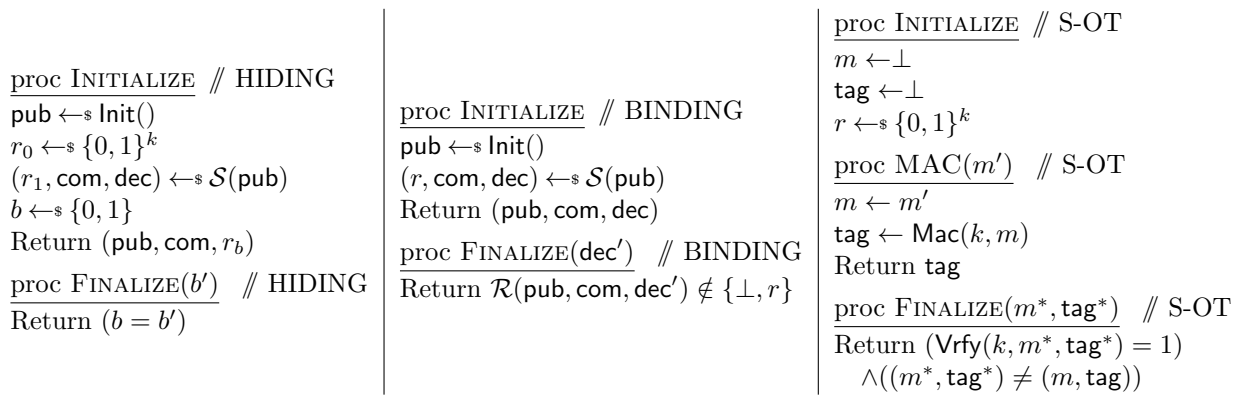

Figure 23: Games defining security (hiding and binding) for an encapsulation scheme, and one-time strong unforgeability for a message authentication code.

C then computes the challenge ciphertext as $C^* \leftarrow (g^z, (h^z)^\delta)$. This is a valid ciphertext as $C^* = (g^z, u_1^z u_2^{zw^*})$. C sets the challenge session key $K^* \leftarrow T$, so that it is the correct session key when the DBDH tuple has $T = e(g,h)^{xyz}$ and a random element of $\mathbb{G}_T$ otherwise. C then runs A with public key (Z, u_1, u_2). C responds to A's decapsulation queries $(\phi_{a,b}, C_1, C_2)$ in the following way. Let $Z' \leftarrow Z^a \cdot e(g,h)^b = e(g,h)^{a(dk)+b}$ and calculate $w \leftarrow H_k(C_1 || Z')$. First check whether $\phi_{a,b}(dk) = dk$ by checking whether $Z' = Z$. If this equality holds and $C = C^*$ then return $\perp$. Next check the ciphertext is valid, that is check that $e(C_1, u_1 u_2^w) = e(g, C_2)$. If this equality does not hold, the ciphertext is invalid so C returns $\perp$ to A. Next, check whether $w = w^*$ and return $\perp$ if so. Now compute

$$K \leftarrow \frac{e(C_1, (h^y)^{\frac{-\delta}{w-w*}} u_1 u_2^w)}{e((g^y)^{\frac{-1}{w-w^*}} g, C_2)}$$

and return K to A. Note that, assuming all the checks hold, we have:

$$K = \left(\frac{e(g,h)^{x(w-w*)+\delta-y\frac{\delta}{w-w^*}}}{e(g,h)^{x(w-w*)+\delta-y\frac{\delta}{w-w^*}-xy\frac{w-w^*}{w-w^*}}} \right)^t = e(g,h)^{xyt} \ ,$$

where $C_1 = g^t$, i.e. t is the randomness used in creating the ciphertext. Hence this decapsulation procedure is correct. When A makes its challenge query, C responds with the challenge ciphertext C^* and session key K^*. When A halts and outputs its guess b', C forwards this guess as its own output. C provides a perfect simulation of G_2 for C, so C wins its game precisely when A wins, giving $\mathbf{Adv}^{\mathrm{dbdh}}(C) = 2\Pr[\mathrm{G}_2^A] - 1$ as required.

Gathering terms, we get that

$$\mathbf{Adv}^{\text{kem-rka}}_{\mathcal{KEM},\Phi}(A) \leq \mathbf{Adv}^{\mathrm{dbdh}}(C) + 2\mathbf{Adv}^{\mathrm{cr}}_H(B) + 2q/p \ ,$$

establishing the theorem. ∎

E RKA security of the Boneh-Katz transform

The Boneh-Katz transform [12] constructs a CCA-PKE scheme from a selective-ID secure IBE scheme, making use of an encapsulation scheme and a message authentication code.

An encapsulation scheme consists of three algorithms $(\mathsf{Init}, \mathcal{S}, \mathcal{R})$ such that $\mathsf{Init}()$ outputs a string pub of public parameters, $\mathcal{S}(\mathsf{pub})$ outputs $(r, \mathsf{com}, \mathsf{dec})$ where $r \in \{0,1\}^k$, and $\mathcal{R}(\mathsf{pub.com}, \mathsf{dec})$ outputs

Algorithm $\mathcal{K}$:
$s \leftarrow\!\!\$\ \mathcal{S}$
$\pi \leftarrow \mathcal{P}(s)$
$\mathsf{pub} \leftarrow\!\!\$\ \mathsf{Init}()$
$pk \leftarrow (\pi, \mathsf{pub}))$
Return (s, pk)

Algorithm $\mathcal{E}(pk, M)$:
$(r, \mathsf{com}, \mathsf{dec}) \leftarrow\!\!\$\ \mathcal{S}(\mathsf{pub})$
$c \leftarrow\!\!\$\ \mathcal{E}(\pi, \mathsf{com}, M||\mathsf{dec})$
$\mathsf{tag} \leftarrow \mathsf{Mac}(r, c)$
$C \leftarrow (\mathsf{com}, c, \mathsf{tag})$
Return C

Algorithm $\mathcal{D}(s, (\mathsf{com}, c, \mathsf{tag}))$:
$dk \leftarrow\!\!\$\ \mathcal{K}(s, \mathsf{com})$
$M' \leftarrow \mathcal{D}(dk, c)$
If $(M' = \perp)$ Return $\perp$
$M||\mathsf{dec} \leftarrow M'$
$r \leftarrow \mathcal{R}(\mathsf{pub}, \mathsf{com}, \mathsf{dec})$
If $(r = \perp)$
  Return $\perp$
If $(\mathsf{Vrfy}(r, c, \mathsf{tag}) = 0)$
  Return $\perp$
Return M

Figure 24: Boneh-Katz transform algorithms.

$r \in \{0,1\}^k \cup \{\perp\}$. The correctness property requires that when pub is output by $\mathsf{Init}()$ and $(r, \mathsf{com}, \mathsf{dec})$ is output by $\mathcal{S}(\mathsf{pub})$, we have that $\mathcal{R}(\mathsf{pub}, \mathsf{com}, \mathsf{dec}) = r$. An encapsulation scheme should satisfy the hiding and binding properties defined by the games in Figure 23.

A message authentication code consists of two algorithms $(\mathsf{Mac}, \mathsf{Vrfy})$ such that $\mathsf{Mac}(k, m)$ outputs a tag tag, and $\mathsf{Vrfy}(k, m, \mathsf{tag})$ outputs 0 or 1. The correctness property requires that when tag is output by $\mathsf{Mac}(k, m)$ we have that $\mathsf{Vrfy}(k, m, \mathsf{tag}) = 1$. The game defining one-time strong unforgeability of a message authentication code is given in Figure 23. The adversary is allowed to make only one call to $\mathrm{MAC}(m')$.

The transform uses an encapsulation scheme where com and dec are n bit strings, and an IBE scheme with identities of length n. The algorithms of the transform are described in Figure 24.

We show that the Boneh-Katz transform preserves RKA security, that is that the transform can be applied to a Φ-RKA-selective-ID secure IBE scheme to obtain a Φ-RKA-secure CCA-PKE scheme. The encapsulation scheme and message authentication scheme do not need to be RKA-secure, instead they need only meet regular hiding and binding, and one-time strong unforgeability notions respectively. The proof follows closely the original proof in [12].

Theorem E.1 *Let $\mathcal{PKE} = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ be the public-key encryption scheme constructed as in Figure 24 from Φ-RKA-selective-ID secure $\mathcal{IBE} = (\mathcal{S}, \mathcal{P}, \mathcal{K}, \mathcal{E}, \mathcal{D})$, hiding and binding encapsulation scheme $\mathcal{ECP} = (\mathsf{Init}, \mathcal{S}, \mathcal{R})$ and one-time strongly unforgeable message authentication code $\mathcal{MAC} = (\mathsf{Mac}, \mathsf{Vrfy})$. Then $\mathcal{PKE}$ is a Φ-RKA-secure CCA-PKE scheme.*

Proof: Let A be an adversary against the Φ-RKA-security of $\mathcal{PKE}$. We will prove the theorem by bounding A's probability of success. The proof uses the sequence of games in Figure 25. Game G_0 implements PKE^A with $\mathcal{PKE}$, so we have

$$\mathbf{Adv}^{\text{pke-rka}}_{\mathcal{PKE},\Phi}(A) = 2 \cdot \Pr[\mathrm{G}_0^A] - 1 \ . \tag{19}$$

In game G_0 the values $(r^*, \mathsf{com}^*, \mathsf{dec}^*)$ are generated in the initialization phase rather than when A makes its call to LR. This makes no difference since A's actions have no effect on their generation.

Game G_1 is as G_0 except that the decryption oracle outputs $\perp$ on input an RKD function ϕ where $\phi(s) = s$ and a ciphertext of the form $(\mathsf{com}^*, c, \mathsf{tag})$. We say that a ciphertext $(\mathsf{com}, c, \mathsf{tag})$ is valid if its decryption under s does not return $\perp$. Let Valid be the event that A playing game G_1 submits to its decryption oracle an RKD function ϕ where $\phi(s) = s$ and a valid ciphertext of the form $(\mathsf{com}^*, c, \mathsf{tag})$. Since G_0 and G_1 are identical until Valid occurs we have that

$$\left|\Pr[\mathrm{G}_0^A] - \Pr[\mathrm{G}_1^A]\right| \leq \Pr[\mathsf{Valid}] \ . \tag{20}$$

We now want to bound $\Pr[\mathsf{Valid}]$. Let NoBind be the event that A playing G_1 ever submits to its decryption oracle an RKD function ϕ where $\phi(s) = s$ and a ciphertext of the form $(\mathsf{com}^*, c, \mathsf{tag})$, such that c decrypts under $dk \leftarrow\!\!\$\, \mathcal{K}(s, \mathsf{com}^*)$ to $M||\mathsf{dec}$, and $\mathcal{R}(\mathsf{pub}, \mathsf{com}^*, \mathsf{dec}) = r$ with $r \notin \{r^*, \perp\}$. Let Forge_i be the event that A ever submits to its decryption oracle an RKD function ϕ where $\phi(s) = s$ and a ciphertext $(\mathsf{com}^*, c, \mathsf{tag})$ such that $\mathsf{Vrfy}(r^*, c, \mathsf{tag}) = 1$ in game G_i, for $i \in \{1, 2\}$. Then we have a bound on $\Pr[\mathsf{Valid}]$ as

$$\Pr[\mathsf{Valid}] \leq \Pr[\mathsf{NoBind}] + \Pr[\mathsf{Forge}_1] \ . \tag{21}$$

We build an adversary A_1 against the binding property of the encapsulation scheme such that

$$\mathbf{Adv}^{\mathrm{bind}}_{\mathcal{ECP}}(A_1) = \Pr[\mathsf{NoBind}] \ . \tag{22}$$

The adversary A_1 behaves as follows. On input $(\mathsf{pub}, \mathsf{com}^*, \mathsf{dec}^*)$, A_1 chooses $s \leftarrow\!\!\$\, \mathcal{S}$, then computes $\pi \leftarrow \mathcal{P}(s)$ and runs A on (π, pub). A_1 uses its knowledge of s to respond to A's decryption queries as specified by game G_1, returning $\perp$ in response to queries of the form $(\phi, (\mathsf{com}^*, c, \mathsf{tag}))$ with $\phi(s) = s$ and storing them in a list. When A makes its LR query, A_1 chooses a random bit $b \leftarrow\!\!\$\, \{0, 1\}$ and computes $c^* \leftarrow\!\!\$\, \mathcal{E}(\pi, \mathsf{com}^*, M_b||\mathsf{dec}^*)$. A_1 then recovers r^* as $r^* \leftarrow \mathcal{R}(\mathsf{pub}, \mathsf{com}^*, \mathsf{dec}^*)$ and computes $\mathsf{tag}^* \leftarrow \mathsf{Mac}(r^*, c^*)$, returning to A the challenge ciphertext $C^* \leftarrow (\mathsf{com}^*, c^*, \mathsf{tag}^*)$. When A halts, A_1 ignores its output and decrypts under s each of A's queries of the form $(\phi, (\mathsf{com}^*, c, \mathsf{tag}))$ where $\phi(s) = s$ to see if NoBind occurred, in which case one such query reveals a value dec in the IBE ciphertext such that $\mathcal{R}(\mathsf{pub}, \mathsf{com}^*, \mathsf{dec}) = r$ with $r \notin \{r^*, \perp\}$. If such a query was submitted, A_1 outputs dec, violating the binding property of the encapsulation scheme and establishing Equation (22).

Game G_2 is as G_1 with a modification to the way the challenge ciphertext is generated. The IBE ciphertext component of the challenge ciphertext in game G_2 is computed as $c \leftarrow\!\!\$\, \mathcal{E}(\pi, \mathsf{com}^*, 0^{M_0}||0^n)$, where n is the length of a decommitment string in $\mathcal{ECP}$. The tag component $\mathsf{tag}^* \leftarrow \mathsf{Mac}(r^*, c^*)$ is computed as usual and the challenge ciphertext is $(\mathsf{com}^*, c^*, \mathsf{tag}^*)$. The challenge ciphertext is now independent of b, so we have that

$$\Pr[\mathrm{G}_2^A] = \frac{1}{2} \ . \tag{23}$$

We now want to bound $\Pr[\mathrm{G}_2^A] - \Pr[\mathrm{G}_1^A]$. To achieve this, we build an adversary A_2 against the Φ-RKA-selective-ID security of $\mathcal{IBE}$. The adversary A_2 generates public parameters for the encapsulation scheme as $\mathsf{pub} \leftarrow\!\!\$\, \mathsf{Init}()$, then generates $(r^*, \mathsf{com}^*, \mathsf{dec}^*) \leftarrow\!\!\$\, \mathcal{S}(\mathsf{pub})$. It outputs challenge identity com^* and receives in return the master public key π. A_2 then runs A on (π, pub).

A_2 computes appropriate responses to A's decryption queries $(\phi, (\mathsf{com}, c, \mathsf{tag}))$ by using its key derivation oracle to obtain $dk \leftarrow (\phi(s), \mathsf{com})$ then decrypting in the usual way, responding with $\perp$ to decryption queries of the form $(\phi, (\mathsf{com}^*, c, \mathsf{tag}))$ where $\phi(s) = s$. Note that A_2 never makes a key derivation query for the challenge identity com^* under an RKD function ϕ with $\phi(s) = s$.

When A makes its LR query with messages M_0, M_1, A_2 chooses $d \leftarrow\!\!\$\, \{0, 1\}$ and submits to its own LR oracle the messages $M_d||\mathsf{dec}^*$ and $0^{M_0}||0^n$. A_2 gets in return an IBE ciphertext c^*, then computes $\mathsf{tag}^* \leftarrow \mathsf{Mac}(r^*, c^*)$ and returns $(\mathsf{com}^*, c^*, \mathsf{tag}^*)$ to A.

A may make further decryption queries to which A_2 responds as before, then A halts outputting a bit d'. If $d' = d$ then A_2 outputs 1, otherwise A_2 outputs 0.

When the hidden bit in A_2's game is $b = 0$ then c^* is an encryption of $M_d||\mathsf{dec}^*$, so A's view is that of game G_1, while when the hidden bit in A_2's game is $b = 1$ then c^* is an encryption of $0^{M_0}||0^n$, so A's

view is that of game G_2.

We then have that

$$\begin{aligned}\mathbf{Adv}^{\text{sibe-rka}}_{\mathcal{IBE},\Phi}(A_2) &= 2 \cdot \Pr[b' = b] - 1 \\ &= \Pr[G_2^A] - \Pr[G_1^A] \ . \end{aligned} \tag{24}$$

A similar adversary A_3 can be built bounding $\Pr[\mathsf{Forge}_2] - \Pr[\mathsf{Forge}_1]$. A_3 behaves exactly as A_2 until A halts, at which point A_3 checks if A has made a decryption query of the form $(\phi, (\mathsf{com}^*, c, \mathsf{tag}))$ with $\phi(s) = s$ and $\mathsf{Vrfy}(r^*, c^*, \mathsf{tag}) = 1$. If such a query has occurred A_3 outputs 1, otherwise A_3 outputs 0. We then have that

$$\begin{aligned}\mathbf{Adv}^{\text{sibe-rka}}_{\mathcal{IBE},\Phi}(A_3) &= 2 \cdot \Pr[b' = b] - 1 \\ &= \Pr[\mathsf{Forge}_2] - \Pr[\mathsf{Forge}_1] \ . \end{aligned} \tag{25}$$

Game G_3 is as game G_2 but with a modification to the way the challenge ciphertext is generated. The tag component of the challenge ciphertext is now generated by choosing $r \leftarrow\!\!\$\ \{0,1\}^k$ and computing $\mathsf{tag}^* \leftarrow \mathsf{Mac}(r, c^*)$. We let Forge_3 denote the event that A playing game G_3 makes a decryption query of the form $(\phi, (\mathsf{com}^*, c, \mathsf{tag}))$ with $\phi(s) = s$ and $\mathsf{Vrfy}(r, c, \mathsf{tag}) = 1$.

We now want to bound $\Pr[\mathsf{Forge}_2] - \Pr[\mathsf{Forge}_3]$ which we do by constructing an adversary A_4 against the hiding property of the encapsulation scheme. On input $(\mathsf{pub}, \mathsf{com}^*, \tilde{r})$, A_4 chooses $s \leftarrow\!\!\$\ \mathcal{S}$, then computes $\pi \leftarrow \mathcal{P}(s)$ and runs A on (π, pub). A_4 uses its knowledge of s to respond to A's decryption queries, returning $\perp$ in response to queries of the form $(\phi, (\mathsf{com}^*, c, \mathsf{tag}))$ where $\phi(s) = s$. When A makes its LR query, A_4 computes $c^* \leftarrow\!\!\$\ \mathcal{E}(\pi, \mathsf{com}^*, 0^{M_0}||0^n)$. A_4 then computes $\mathsf{tag}^* \leftarrow \mathsf{Mac}(\tilde{r}, c^*)$, returning to A the challenge ciphertext $C^* \leftarrow (\mathsf{com}^*, c^*, \mathsf{tag}^*)$. When A halts, A_4 checks if any of A's decryption queries were of the form $(\phi, (\mathsf{com}^*, c, \mathsf{tag}))$ with $\phi(s) = s$ and $\mathsf{Vrfy}(\tilde{r}, c, \mathsf{tag}) = 1$, outputting 1 if so and 0 otherwise. If $b = 1$ in the hiding game then A's view is that of game G_2, so A_4 outputs 1 with probability Forge_2, while if $b = 0$ in the hiding game then $\tilde{r}$ is independent of com^* and A's view is that of game G_3, so A_4 outputs 1 with probability Forge_3. Then we have that

$$\begin{aligned}\mathbf{Adv}^{\text{hide}}_{\mathcal{ECP}}(A_4) &= 2 \cdot \Pr[b' = b] - 1 \\ &= \Pr[\mathsf{Forge}_2] - \Pr[\mathsf{Forge}_3] \ . \end{aligned} \tag{26}$$

Finally, we want to bound $\Pr[\mathsf{Forge}_3]$, which we do by constructing an adversary A_5 against the one-time unforgeability of the message authentication code. A_5 chooses a random index $j \in \{1, \ldots, q\}$, where q is the maximum number of decryption queries A can make. A_5 generates $\mathsf{pub} \leftarrow\!\!\$\ \mathsf{Init}()$, then chooses $s \leftarrow\!\!\$\ \mathcal{S}$ and computes $\pi \leftarrow \mathcal{P}(s)$, then runs A on (π, pub). A_5 uses its knowledge of s to respond to A's decryption queries as specified by game G_3, returning $\perp$ in response to queries of the form $(\phi, (\mathsf{com}^*, c, \mathsf{tag}))$ where $\phi(s) = s$. If A makes its jth decryption query $(\phi_j, (\mathsf{com}_j, c_j, \mathsf{tag}_j))$ before its LR query, A_5 halts and outputs (c_j, tag_j). Otherwise, when A makes its LR query A_5 computes $(r^*, \mathsf{com}^*, \mathsf{dec}^*) \leftarrow\!\!\$\ \mathcal{S}(\mathsf{pub})$, then computes $c^* \leftarrow\!\!\$\ \mathcal{E}(\pi, \mathsf{com}^*, 0^{M_0}||0^n)$. A_5 then makes its MAC query with message c^* to obtain tag^*, returning to A the challenge ciphertext $C^* \leftarrow (\mathsf{com}^*, c^*, \mathsf{tag}^*)$. When A makes its jth decryption query $(\phi_j, (\mathsf{com}_j, c_j, \mathsf{tag}_j))$, A_5 halts and outputs (c_j, tag_j). If the event Forge_3 occurred, that is if A made a decryption query of the form $(\phi, (\mathsf{com}^*, c, \mathsf{tag}))$ with $\phi(s) = s$ and $\mathsf{Vrfy}(r, c, \mathsf{tag}) = 1$, then with probability $1/q$ it was on A's jth query, so the probability that A_5 makes a successful forgery is $\Pr[\mathsf{Forge}_3]/q$. We now have that

$$\mathbf{Adv}^{\text{forge}}_{\mathcal{MAC}}(A_5) = \Pr[\mathsf{Forge}_3]/q \ . \tag{27}$$

$\underline{\text{proc INITIALIZE}}$ // All
$b \leftarrow\!\!\$\ \{0,1\}$; $C^* \leftarrow \bot$
$s \leftarrow\!\!\$\ \mathcal{S}$; $\pi \leftarrow \mathcal{P}(s)$
$\mathsf{pub} \leftarrow\!\!\$\ \mathsf{Init}()$
$pk \leftarrow (\pi, \mathsf{pub})$
$(r^*, \mathsf{com}^*, \mathsf{dec}^*) \leftarrow\!\!\$\ \mathcal{S}(\mathsf{pub})$
Return (s, pk)

$\underline{\text{proc LR}(M_0, M_1)}$ // $\mathrm{G}_0, \mathrm{G}_1$
If $(|M_0| \neq |M_1|)$
  Return $\bot$
$c^* \leftarrow\!\!\$\ \mathcal{E}(\pi, \mathsf{com}^*, M_b||\mathsf{dec}^*)$
$\mathsf{tag}^* \leftarrow \mathsf{Mac}(r^*, c^*)$
$C^* \leftarrow (\mathsf{com}^*, c^*, \mathsf{tag}^*)$
Return C^*

$\underline{\text{proc LR}(M_0, M_1)}$ // G_2
If $(|M_0| \neq |M_1|)$
  Return $\bot$
$c^* \leftarrow\!\!\$\ \mathcal{E}(\pi, \mathsf{com}^*, 0^{M_0}||0^n)$
$\mathsf{tag}^* \leftarrow \mathsf{Mac}(r^*, c^*)$
$C^* \leftarrow (\mathsf{com}^*, c^*, \mathsf{tag}^*)$
Return C^*

$\underline{\text{proc DEC}(\phi, (\mathsf{com}, c, \mathsf{tag}))}$ // G_0
If $((\phi(s) = s) \wedge ((\mathsf{com}, c, \mathsf{tag}) = C^*))$
  Return $\bot$
$dk \leftarrow \mathcal{K}(s, \mathsf{com})$
$M' \leftarrow \mathcal{D}(dk, c)$
If $(M' = \bot)$
  Return $\bot$
$M||\mathsf{dec} \leftarrow M'$
$r \leftarrow \mathcal{R}(\mathsf{pub}, \mathsf{com}, \mathsf{dec})$
If $(r = \bot)$
  Return $\bot$
If $(\mathsf{Vrfy}(r, c, \mathsf{tag}) = 0)$
  Return $\bot$
Return M

$\underline{\text{proc LR}(M_0, M_1)}$ // G_3
If $(|M_0| \neq |M_1|)$
  Return $\bot$
$c^* \leftarrow\!\!\$\ \mathcal{E}(\pi, \mathsf{com}^*, 0^{M_0}||0^n)$
$r \leftarrow\!\!\$\ \{0,1\}^k$
$\mathsf{tag}^* \leftarrow \mathsf{Mac}(r, c^*)$
$C^* \leftarrow (\mathsf{com}^*, c^*, \mathsf{tag}^*)$
Return C^*

$\underline{\text{proc DEC}(\phi, (\mathsf{com}, c, \mathsf{tag}))}$ // $\mathrm{G}_1, \mathrm{G}_2, \mathrm{G}_3$
If $((\phi(s) = s) \wedge ((\mathsf{com}, c, \mathsf{tag}) = C^*))$
  Return $\bot$
If $((\phi(s) = s) \wedge (\mathsf{com} = \mathsf{com}^*))$
  Return $\bot$
$dk \leftarrow \mathcal{K}(s, \mathsf{com})$
$M' \leftarrow \mathcal{D}(dk, c)$
If $(M' = \bot)$
  Return $\bot$
$M||\mathsf{dec} \leftarrow M'$
$r \leftarrow \mathcal{R}(\mathsf{pub}, \mathsf{com}, \mathsf{dec})$
If $(r = \bot)$
  Return $\bot$
If $(\mathsf{Vrfy}(r, c, \mathsf{tag}) = 0)$
  Return $\bot$
Return M

$\underline{\text{proc FINALIZE}(b')}$ // All
Return $(b = b')$

Figure 25: Games for the proof of Theorem E.1.

Collecting equations 19-27, we get that

$$
\begin{aligned}
\mathbf{Adv}^{\text{pke-rka}}_{\mathcal{PKE},\Phi}(A) &= \left|2 \cdot \Pr[\mathrm{G}_0^A] - 1\right| \\
&\leq 2 \cdot \left|\Pr[\mathrm{G}_0^A] - \Pr[\mathrm{G}_1^A]\right| + 2 \cdot \left|\Pr[\mathrm{G}_1^A] - \Pr[\mathrm{G}_2^A]\right| + 2 \cdot \left|\Pr[\mathrm{G}_2^A] - \tfrac{1}{2}\right| \\
&\leq 2 \cdot \Pr[\mathsf{Valid}] + 2 \cdot \mathbf{Adv}^{\text{sibe-rka}}_{\mathcal{IBE},\Phi}(A_2) + 2 \cdot \left|\tfrac{1}{2} - \tfrac{1}{2}\right| \\
&\leq 2 \cdot \mathbf{Adv}^{\text{sibe-rka}}_{\mathcal{IBE},\Phi}(A_2) + 2 \cdot \Pr[\mathsf{NoBind}] + 2 \cdot \Pr[\mathsf{Forge}_1] \\
&\leq 2 \cdot \Big(\mathbf{Adv}^{\text{bind}}_{\mathcal{ECP}}(A_1) + \mathbf{Adv}^{\text{sibe-rka}}_{\mathcal{IBE},\Phi}(A_2)\Big) + 2 \cdot |\Pr[\mathsf{Forge}_1] - \Pr[\mathsf{Forge}_2]| \\
&\qquad + 2 \cdot |\Pr[\mathsf{Forge}_2] - \Pr[\mathsf{Forge}_3]| + 2 \cdot \Pr[\mathsf{Forge}_3] \\
&\leq 2 \cdot \Big(\mathbf{Adv}^{\text{bind}}_{\mathcal{ECP}}(A_1) + \mathbf{Adv}^{\text{sibe-rka}}_{\mathcal{IBE},\Phi}(A_2) + \mathbf{Adv}^{\text{sibe-rka}}_{\mathcal{IBE},\Phi}(A_3) + \mathbf{Adv}^{\text{hide}}_{\mathcal{ECP}}(A_4) + q\mathbf{Adv}^{\text{forge}}_{\mathcal{MAC}}(A_5)\Big) \ ,
\end{aligned}
$$

establishing the theorem. ∎