Public-Key Cryptography with Joint and Related-Key Security

Susan Thomson

Thesis submitted to the University of London
for the degree of Doctor of Philosophy

Information Security Group
School of Mathematics and Information Security
Royal Holloway, University of London

2014

Declaration

These doctoral studies were conducted under the supervision of Prof. Kenneth G. Paterson.

The work presented in this thesis is the result of original research carried out by myself, in collaboration with others, whilst enrolled in the Department of Information Security as a candidate for the degree of Doctor of Philosophy. This work has not been submitted for any other degree or award in any other university or educational establishment.

Susan Thomson
April, 2014

Abstract

The principle of key separation dictates using different keys for different cryptographic operations. We investigate the topic of joint security, where a single keypair is used in multiple primitives in a secure manner. We concentrate mainly on the case of encryption and signature under a shared keypair, giving a generic construction and a more efficient direct construction, both secure in the standard model, and show how these results relate to signcryption.

We then turn our attention to security under related-key attacks (RKA), where an adversary can modify a stored secret key and observe the outputs of the system as it operates under this new key. We provide a framework enabling the construction of RKA-secure identity-based encryption (IBE) schemes, and show how specific instantiations of the framework yield IBE schemes secure against adversaries deriving new keys through affine and polynomial transformations of the master secret key. From this we obtain the first constructions of RKA-secure schemes for a variety of primitives under the same non-linear key transformations.

Since achieving joint or RKA security often depends on the format of the stored keys, we introduce key-versatile signatures, where the public key is an arbitrary one-way function of the secret key, and show how these can be used to obtain further results in joint and RKA security and beyond.

Contents

Publications

This thesis is based on the following three published papers.

1. Kenneth G. Paterson, Jacob C. N. Schuldt, Martijn Stam, and Susan Thomson. “On the Joint Security of Encryption and Signature, Revisited”. In: *ASIACRYPT 2011*. Ed. by Dong Hoon Lee and Xiaoyun Wang. Vol. 7073. LNCS. Seoul, South Korea: Springer, Berlin, Germany, 2011, pp. 161–178
2. Mihir Bellare, Kenneth G. Paterson, and Susan Thomson. “RKA Security beyond the Linear Barrier: IBE, Encryption and Signatures”. In: *ASIACRYPT 2012*. Ed. by Xiaoyun Wang and Kazue Sako. Vol. 7658. LNCS. Beijing, China: Springer, Berlin, Germany, 2012, pp. 331–348
3. Mihir Bellare, Sarah Meiklejohn, and Susan Thomson. “Key-Versatile Signatures and Applications: RKA, KDM and Joint Enc/Sig”. In: *EUROCRYPT 2014*. Ed. by Phong Q. Nguyen and Elisabeth Oswald. Vol. 8441. LNCS. Copenhagen, Denmark: Springer, Berlin, Germany, 2014, pp. 496–513

Acknowledgements

I would like to thank my supervisor Kenny Paterson, who took four years of backchat in good humour, and gave freely of his time and travel budget in return.

I would also like to thank my other excellent co-authors: Sarah Meiklejohn, Jacob Schuldt, Martijn Stam, and in particular Mihir Bellare, whose work inspired my love of provable security, and who was a welcoming host during my time at UCSD.

I am grateful for the financial support of the EPSRC.

Thanks go those who were there for lunches at the SCR and evenings in the Happy Man. Thanks especially to James, who graciously tolerated unreasonable office behaviour, from thinking aloud to improvised percussion, and to Shahram, for the nachos. Good work, team.

Thanks also to Niall, a pal and a confidant, and to Gaven, who wants it in writing. Unbelievable.

Finally I thank my family, whose love and support get me through.

Chapter 1

Introduction

Contents

This chapter gives an overview of the thesis. We provide the motivation for our research and present the overall structure of the thesis.

1.1 Motivation

We investigate the security of cryptographic schemes when additional information about the underlying secret key is leaked through either using the same key in multiple primitives, or through tampering with the key and observing the operation of the scheme under the modified key.

The first case, using the same key in multiple primitives, may come about in an attempt to reduce certificate overhead, public key size, or code footprint, or may simply be a result of reckless implementation of cryptography. While the folklore principle of key separation dictates fresh keys for every primitive, given the expense of generating and certifying keys, key reuse between primitives is tempting, and is even permitted by some standards [58, 42].

The second case, tampering with a stored key, can occur through physical interference with a device implementing cryptography. This may be through techniques such as exposing a chip to unusual voltages or temperatures, inducing clock glitches, or application of an intense light source [4, 97].

Since securing devices against physical attacks and preventing those in charge of how cryptography is implemented from making their own "optimisations" are challenging tasks, rather than trying to prevent these key-information leaking scenarios, our aim in this work is to build schemes secure in the face of them.

We approach this by looking at primitives under extensions of the usual notions of security that model this additional leakage. For example, in the case of key reuse between encryption and signature [72] this means allowing an adversary to obtain signatures when trying to distinguish between encrypted messages, and allowing an adversary to obtain decryptions while trying to forge a signature. Such additional information can be very helpful to an adversary. Consider the case of textbook RSA, where signing under a key is exactly the same operation as decryption under that key. An adversary trying to determine the message encrypted in a ciphertext can simply ask for a signature on that ciphertext, the response being the message in the clear. Similarly an adversary trying to forge a signature on a message can obtain such a forgery by submitting the message for decryption. Clearly key reuse here is fatal to the security of both encryption and signature. We give constructions of encryption and signature schemes that afford users the benefits of key reuse without degrading security, by showing that the schemes remain secure even against adversaries who can obtain both decryptions and signatures under the same key.

In modelling security against an adversary tampering with a stored key we consider what is called a related key attack (RKA) [18], where an adversary's tampering transforms a key into one related to the original key through a related-key deriving (RKD) function. The adversary then sees the result of operations under the modified key. An adversary with this capability may be able to break a scheme with ease. For example an adversary attacking a symmetric encryption scheme who is able to transform the stored key into a known constant value can decrypt any ciphertexts encrypted under that key. No symmetric scheme can be secure against an adversary who can transform the key in this way, that is, an adversary who can apply a constant RKD function to the stored key. For this reason, security against related-key attacks is parameterised by a set Φ specifying the RKD functions an adversary can apply. Our aim then is to construct schemes secure against RKAs for as large a set Φ as possible. We focus on schemes where the stored secret key is an element of a field where, prior to our work, RKA-secure constructions were known only for sets Φ consisting of linear transformations [13, 14, 8, 99], or in weaker security models than those we consider here [69]. We go beyond this "linear barrier", achieving RKA security for a range of primitives when the set Φ consists of affine and polynomial transformations of the key.

Lastly, revisiting key reuse, we introduce a new primitive, a signature scheme where the relationship between secret and public keys is an arbitrary one-way function F. These "key-versatile" signatures allow us to sign with keys already in use for another purpose, by setting this function F to be the one-way function defining the relationship between the secret and public keys of an existing scheme. We define strong security properties which we require of key-versatile signatures, and prove that these properties ensure both that the signatures are secure and that they do not impact on the security of the existing scheme, which might otherwise be affected by the leakage of key-information in the form of the signatures computed under the same key.

These results show that key-information leakage additional to that allowed for in standard notions of security while potentially damaging is not always so, and that carefully considered constructions can achieve security where such leakage causes others to fail.

1.2 Thesis Structure

Chapter 2. This chapter defines all the necessary primitives and notions of security, and some standard constructions that will be used later in the thesis.

Our contributions are then presented in the remaining chapters, each chapter beginning with an introduction to the topic and concluding with a summary of the results and some open problems in the area.

Chapter 3. In this chapter we revisit the topic of joint encryption and signature schemes, where a single keypair is used for both encryption and signature in a secure manner. We give a general construction for a joint encryption and signature scheme that uses identity-based encryption (IBE) as a component, and that is secure in the standard model. We then provide a more efficient direct construction, also secure in the standard model. Finally, we show how these results relate to signcryption.

Chapter 4. This chapter concerns security under related-key attacks, where an adversary can modify a stored secret key and observe the outputs of the system as it operates under this new key. We provide a framework enabling the construction of RKA-secure IBE schemes, and show how specific instantiations of the framework yield IBE schemes secure against adversaries deriving new keys through affine and polynomial transformations of the master secret key.

Chapter 5. In this chapter, we use the RKA-secure IBE schemes of the previous chapter, and the techniques used in their construction, to build further RKA-secure primitives. From RKA-secure IBE we immediately get RKA-secure PKE and signature schemes under the same assumptions. We construct RKA-secure PKE from IBE through the Boneh-Katz transform, and through applying the techniques of the previous chapter to build an RKA-secure KEM. We additionally build RKA-secure joint encryption and signature from IBE. When the base IBE scheme has a further malleability property, the PKE scheme obtained through the CHK transform can be converted into an RKA-secure CCA-SE (CCA-secure symmetric encryption) scheme. These results give the first RKA secure schemes for the primitives signature, PKE, and CCA-SE for non-linear RKAs.

Chapter 6. This chapter introduces key-versatile signatures, which allow us to sign with keys already in use for another purpose, without changing the keys and without impacting the security of the original purpose. This allows us to obtain advances across a collection of challenging domains including joint encryption and signature, security against related-key attack and security for key-dependent messages (KDM). Specifically we show how to (1) Add signing capability to existing encryption capability with zero overhead in the size of the public key (2) Obtain RKA-secure signatures from any RKA-secure one-way function, yielding new RKA-secure signature schemes (3) Add integrity to encryption while maintaining KDM-security.

Chapter 2

Preliminaries

Contents

This chapter defines all the necessary primitives and notions of security, and some standard constructions that will be used in the thesis.

2.1 Notation

If x is a binary string then $|x|$ is its bit length. If S is a finite set then $|S|$ denotes its size and $s \leftarrow\$ S$ denotes picking an element uniformly at random from S and assigning it to s. For sets X, Y let $\mathsf{Fun}(X, Y)$ be the set of all functions mapping X to Y. We denote by $\lambda \in \mathbb{N}$ the security parameter and by 1^λ its unary representation. The operator ! denotes logical negation.

Algorithms are randomised unless otherwise indicated. For all algorithms, whether randomised or deterministic, "PT" stands for "polynomial-time". We denote by $y \leftarrow \mathsf{A}(x_1, \ldots; R)$ the operation of running algorithm A on inputs $x_1, \ldots$ and coins R and letting y denote the output. By $y \leftarrow\$ \mathsf{A}(x_1, \ldots)$, we denote the operation of letting $y \leftarrow \mathsf{A}(x_1, \ldots; R)$ for random R. We denote by $[\mathsf{A}(x_1, \ldots)]$ the set of values that have positive probability of being output by A on inputs $x_1, \ldots$. Adversaries are algorithms.

We use games in definitions of security and in proofs. A game G (e.g. Figure 2.1) has a MAIN procedure whose output (what it returns) is the output of the game. We let Pr[G] denote the probability that this output is the boolean true (with $\Pr[\overline{\mathrm{G}}]$ denoting the probability that this output is false). A boolean flag (for example bad), if used in a game, is assumed initialised to false. 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. When a proof involves multiple games, a comment beside each procedure shows which games it belongs to. If the name of a game in this comment is surrounded by a box, the procedure in this game includes the boxed code, otherwise it does not. When there are sections of both dashed and $\boxed{\text{solid-boxed}}$ code, the dashed-boxed code belongs to the games whose names in the comment are enclosed in a dashed box, and the solid-boxed code belongs to the games whose names are enclosed in a solid box

Primitives may have a set of public parameters shared between all users. These could include for example a hash key or a description of a group. When there are multiple keys and parameters involved we will use a subscript to show which belong to which scheme, for example pp_{S} to denote public parameters for scheme S.

2.2 Cryptographic Primitives

FUNCTION FAMILIES. A function family F specifies the following. Via $pp \leftarrow\!\$\ \mathsf{F.Pg}(1^\lambda)$ one can in PT generate a description pp of a function $\mathsf{F.Eval}(pp, \cdot)$: $\mathsf{F.Dom}(pp) \rightarrow \mathsf{F.Rng}(pp)$. We assume that membership of x in the non-empty domain $\mathsf{F.Dom}(pp)$ can be tested in time polynomial in (pp, x) and one can in time polynomial in pp sample a point $x \leftarrow\!\$\ \mathsf{F.Dom}(pp)$ from the domain $\mathsf{F.Dom}(pp)$. The deterministic evaluation algorithm $\mathsf{F.Eval}$ is PT. The range is defined by $\mathsf{F.Rng}(pp) = \{\, \mathsf{F.Eval}(pp, x) : x \in \mathsf{F.Dom}(pp)\,\}$. Testing membership in the range is not required to be PT. (But is in many examples.) We say that F is *one-way* or F is a *one-way function (OWF)* if $\mathbf{Adv}^{\mathrm{ow}}_{\mathsf{F},I}(\cdot)$ is negligible for all PT I, where $\mathbf{Adv}^{\mathrm{ow}}_{\mathsf{F},I}(\lambda) = \Pr[\mathsf{F.Eval}(pp, x') = y]$ under the experiment $pp \leftarrow\!\$\ \mathsf{F.Pg}(1^\lambda)\,;\ x \leftarrow\!\$\ \mathsf{F.Dom}(pp)\,;\ y \leftarrow \mathsf{F.Eval}(pp, x)\,;\ x' \leftarrow\!\$\ I(pp, y)$. We say that F is *second-preimage resistant* if $\mathbf{Adv}^{\mathrm{sec}}_{\mathsf{F},A}(\cdot)$ is negligible for all PT A, where $\mathbf{Adv}^{\mathrm{sec}}_{\mathsf{F},A}(\lambda) = \Pr[(\mathsf{F.Eval}(pp, x') = \mathsf{F.Eval}(pp, x)) \wedge (x' \neq x)]$ under the experiment $pp \leftarrow\!\$\ \mathsf{F.Pg}(1^\lambda)\ ;\ x \leftarrow\!\$\ \mathsf{F.Dom}(pp)\ ;\ x' \leftarrow\!\$\ A(pp, x)$. We say that F is *collision resistant* if $\mathbf{Adv}^{\mathrm{coll}}_{\mathsf{F},A}(\cdot)$ is negligible for all PT A, where $\mathbf{Adv}^{\mathrm{coll}}_{\mathsf{F},A}(\lambda) = \Pr[(\mathsf{F.Eval}(pp, x') = \mathsf{F.Eval}(pp, x)) \wedge (x' \neq x)]$ under the experiment $pp \leftarrow\!\$\ \mathsf{F.Pg}(1^\lambda)\,;\ (x, x') \leftarrow\!\$\ A(pp)$.

```
main IND-CCA$^A_{\mathsf{PKE}}(\lambda)$
$b \leftarrow\!\!\$ \{0,1\}$ ; $c^* \leftarrow \perp$
$pp \leftarrow\!\!\$ \mathsf{PKE.Pg}(1^\lambda)$ ; $(sk, pk) \leftarrow\!\!\$ \mathsf{PKE.Kg}(pp)$
$b' \leftarrow\!\!\$ A^{\text{Dec,LR}}(pp, pk)$
Return $(b = b')$

proc Dec($c$)
If $(c = c^*)$ then Return $\perp$
Return $m \leftarrow \mathsf{PKE.Dec}(pp, sk, c)$

proc LR($m_0, m_1$)
If $(c^* \neq \perp)$ then Return $\perp$
If $(|m_0| \neq |m_1|)$ then Return $\perp$
$c^* \leftarrow\!\!\$ \mathsf{PKE.Enc}(pp, pk, m_b)$
Return $c^*$
```

Figure 2.1: Game IND-CCA defining indistinguishability of public key encryption scheme PKE under chosen-ciphertext attack.

Public-Key Encryption Schemes. A public-key encryption scheme PKE specifies the following PT algorithms: via $pp \leftarrow\!\!\$ \mathsf{PKE.Pg}(1^\lambda)$ one generates public parameters pp common to all users; via $(sk, pk) \leftarrow\!\!\$ \mathsf{PKE.Kg}(pp)$ a user can generate a decryption key sk and corresponding encryption key pk; via $c \leftarrow\!\!\$ \mathsf{PKE.Enc}(pp, pk, m)$ anyone can generate a ciphertext c encrypting message $m \in \mathsf{PKE.MSp}(pp)$ under pk; via $m \leftarrow \mathsf{PKE.Dec}(pp, sk, c)$ a user can deterministically decrypt ciphertext c to get a value $m \in \mathsf{PKE.MSp} \cup \{\perp\}$. Correctness requires that $\mathsf{PKE.Dec}(pp, sk, \mathsf{PKE.Enc}(pp, pk, m)) = m$ for all $\lambda \in \mathbb{N}$, all $pp \in [\mathsf{PKE.Pg}(1^\lambda)]$, all $(sk, pk) \in [\mathsf{PKE.Kg}(pp)]$, and all $m \in \mathsf{PKE.MSp}(pp)$.

We say a public-key encryption scheme PKE is *indistinguishable under chosen-ciphertext attack* or *IND-CCA secure* if $\mathbf{Adv}^{\text{ind-cca}}_{\mathsf{PKE},A}(\cdot)$ is negligible for all PT adversaries A, where $\mathbf{Adv}^{\text{ind-cca}}_{\mathsf{PKE},A}(\lambda) = 2\Pr[\text{IND-CCA}^A_{\mathsf{PKE}}(\lambda)] - 1$ and game IND-CCA is in Figure 2.1. The first "If" statement in the LR procedure ensures the adversary obtains only one challenge ciphertext, while the second "If" statement captures that encryption need not be length hiding, and requires a suitable length function $|\cdot|$ on $\mathsf{PKE.MSp}(pp)$.

Signature schemes. A signature scheme DS specifies the following PT algorithms: via $pp \leftarrow\!\!\$ \mathsf{DS.Pg}(1^\lambda)$ one generates public parameters pp common to all users; via $(sk, pk) \leftarrow\!\!\$ \mathsf{DS.Kg}(pp)$ a user can generate a signing key sk and corresponding public verification key pk; via $\sigma \leftarrow\!\!\$ \mathsf{DS.Sign}(pp, sk, m)$ the signer can generate a signa-

```
main SUF-CMA$^A_{\mathsf{DS}}(\lambda)$ / OT-SUF-CMA$^A_{\mathsf{DS}}(\lambda)$
$Q \leftarrow \emptyset$
$pp \leftarrow\$ \mathsf{DS.Pg}(1^\lambda)$
$(sk, pk) \leftarrow\$ \mathsf{DS.Kg}(pp)$
$(m, \sigma) \leftarrow\$ A^{\textsc{Sign}}(pp, pk)$
Return $(\mathsf{DS.Verify}(pp, pk, m, \sigma) \wedge ((m, \sigma) \notin Q))$

Sign($m$)  // SUF-CMA$^A_{\mathsf{DS}}(\lambda)$ / $\boxed{\text{OT-SUF-CMA}^A_{\mathsf{DS}}(\lambda)}$
$\boxed{\text{If } (Q \neq \emptyset) \text{ then Return } \perp}$
$\sigma \leftarrow\$ \mathsf{DS.Sign}(pp, sk, m)$
$Q \leftarrow Q \cup \{(m, \sigma)\}$
Return $\sigma$
```

Figure 2.2: Game SUF-CMA defining strong unforgeability of signature scheme DS under chosen-message attack, and game OT-SUF-CMA defining one-time strong unforgeability under chosen-message attack. The Sign procedure of game OT-SUF-CMA includes the boxed code, while that of game SUF-CMA does not.

ture σ on a message $m \in \mathsf{DS.MSp}(pp)$; via $d \leftarrow \mathsf{DS.Verify}(pp, pk, m, \sigma)$ a verifier can deterministically produce a decision $d \in \{\mathsf{true}, \mathsf{false}\}$ regarding whether σ is a valid signature of message m under public key pk. Correctness requires that $\mathsf{DS.Verify}(pp, pk, m, \mathsf{DS.Sign}(pp, sk, m)) = \mathsf{true}$ for all $\lambda \in \mathbb{N}$, all $pp \in [\mathsf{DS.Pg}(1^\lambda)]$, all $(sk, pk) \in [\mathsf{DS.Kg}(pp)]$, and all $m \in \mathsf{DS.MSp}(pp)$.

We say a signature scheme DS is *strongly unforgeable* if $\mathbf{Adv}^{\text{suf-cma}}_{\mathsf{DS},A}(\cdot)$ is negligible for all PT adversaries A, where $\mathbf{Adv}^{\text{suf-cma}}_{\mathsf{DS},A}(\lambda) = \Pr[\text{SUF-CMA}^A_{\mathsf{DS}}(\lambda)]$ and game SUF-CMA is in Figure 2.2. We say a signature scheme DS is *one-time strongly unforgeable* if $\mathbf{Adv}^{\text{ot-suf-cma}}_{\mathsf{DS},A}(\cdot)$ is negligible for all PT adversaries A, where $\mathbf{Adv}^{\text{ot-suf-cma}}_{\mathsf{DS},A}(\lambda) = \Pr[\text{OT-SUF-CMA}^A_{\mathsf{DS}}(\lambda)]$ and game OT-SUF-CMA is in Figure 2.2. The boxed code ensures the adversary obtains only one signature in this game. We say a signature scheme DS is *unforgeable under weak chosen-message attack* if $\mathbf{Adv}^{\text{euf-wma}}_{\mathsf{DS},A}(\cdot)$ is negligible for all PT adversaries A, where $\mathbf{Adv}^{\text{euf-wma}}_{\mathsf{DS},A}(\lambda) = \Pr[\text{EUF-WMA}^A_{\mathsf{DS}}(\lambda)]$ and game EUF-WMA is in Figure 2.3.

Identity-Based Encryption Schemes. An identity-based encryption scheme IBE specifies the following PT algorithms: via $pp \leftarrow\$ \mathsf{IBE.Pg}(1^\lambda)$ one generates public parameters pp common to all authorities; via $(msk, mpk) \leftarrow\$ \mathsf{IBE.MKg}(pp)$ an authority can generate a master secret key msk and corresponding master public key mpk; via $usk \leftarrow\$ \mathsf{IBE.UKg}(pp, msk, u)$ an authority can generate a decryption key usk for user $u \in \mathsf{IBE.USp}(pp)$ under master secret key msk; via $c \leftarrow\$ \mathsf{IBE.Enc}(pp, mpk, u,$

```
$\underline{\text{MAIN EUF-WMA}^{A}_{\mathsf{DS}}(\lambda)}$
$Q \leftarrow \emptyset$
$pp \leftarrow_{\$} \mathsf{DS.Pg}(1^{\lambda})$
$(sk, pk) \leftarrow_{\$} \mathsf{DS.Kg}(pp)$
$(m, \sigma) \leftarrow_{\$} A^{\text{SIGN}}(pp)$
Return $(\mathsf{DS.Verify}(pp, pk, m, \sigma) \wedge (m \notin Q))$

$\underline{\text{SIGN}(m_1, \ldots, m_q)}$
If $(Q \neq \emptyset)$ then Return $\perp$
For $i = 1$ to $q$
    $\sigma_i \leftarrow_{\$} \mathsf{DS.Sign}(pp, sk, m_i)$
$Q \leftarrow \{m_1, \ldots, m_q\}$
Return $(pk, \sigma_1, \ldots, \sigma_q)$
```

Figure 2.3: Game EUF-WMA defining existential unforgeability of signature scheme DS under weak chosen-message attack.

m) anyone can generate a ciphertext c encrypting message $m \in \mathsf{IBE.MSp}(pp)$ to user $u \in \mathsf{IBE.USp}(pp)$ under mpk; via $m \leftarrow \mathsf{IBE.Dec}(pp, usk, c)$ a user can deterministically decrypt ciphertext c to get a value $m \in \mathsf{IBE.MSp}(pp) \cup \{\perp\}$. Correctness requires that $\mathsf{IBE.Dec}(pp, \mathsf{IBE.UKg}(pp, msk, u), \mathsf{IBE.Enc}(pp, mpk, u, m)) = m$ for all $\lambda \in \mathbb{N}$, all $pp \in [\mathsf{IBE.Pg}(1^{\lambda})]$, all $(msk, mpk) \in [\mathsf{IBE.MKg}(pp)]$, all $m \in \mathsf{IBE.MSp}(pp)$, and all $u \in \mathsf{IBE.USp}(pp)$.

We say an identity-based encryption scheme IBE is *adaptive-ID indistinguishable under chosen plaintext attack* or *adaptively secure* if $\mathbf{Adv}^{\text{ind-aid}}_{\mathsf{IBE},A}(\cdot)$ is negligible for all PT adversaries A, where $\mathbf{Adv}^{\text{ind-aid}}_{\mathsf{PKE},A}(\lambda) = 2\Pr[\text{IND-aID}^{A}_{\mathsf{IBE}}(\lambda)] - 1$ and game IND-aID is on the top left-hand side of Figure 2.4. The set U is a list of users the adversary has requested a key for, and the "If" statements involving U ensure the adversary is not given a key for the user for whom the challenge ciphertext is encrypted. We say an identity-based encryption scheme IBE is *one-way under chosen plaintext attack* or *one-way* if $\mathbf{Adv}^{\text{ow-aid}}_{\mathsf{IBE},A}(\cdot)$ is negligible for all PT adversaries A, where $\mathbf{Adv}^{\text{ow-aid}}_{\mathsf{PKE},A}(\lambda) = \Pr[\text{OW-aID}^{A}_{\mathsf{IBE}}(\lambda)]$ and game OW-aID is on the top right-hand side of Figure 2.4. We say an identity-based encryption scheme IBE is *selective-ID indistinguishable under chosen plaintext attack* or *selectively secure* if $\mathbf{Adv}^{\text{ind-sid}}_{\mathsf{IBE},A}(\cdot)$ is negligible for all PT adversaries A, where $\mathbf{Adv}^{\text{ind-sid}}_{\mathsf{PKE},A}(\lambda) = 2\Pr[\text{IND-sID}^{A}_{\mathsf{IBE}}(\lambda)] - 1$ and game IND-sID is on the bottom of Figure 2.4. In this game there is no user list U as the challenge user u^* must be selected up-front, before the adversary receives the master public key or any user-level keys. The first "If" statement in each procedure ensures the first meaningful call is to SID to select u^*.

```
main IND-aID$^A_{\mathsf{IBE}}(\lambda)$
$b \leftarrow_\$ \{0,1\}$ ; $c^* \leftarrow \perp$ ; $u^* \leftarrow \perp$ ; $U \leftarrow \emptyset$
$pp \leftarrow_\$ \mathsf{IBE.Pg}(1^\lambda)$
$(msk, mpk) \leftarrow_\$ \mathsf{IBE.MKg}(pp)$
$b' \leftarrow_\$ A^{\mathrm{KD,LR}}(pp, mpk)$
Return $(b = b')$

proc KD$(u)$
$U \leftarrow U \cup \{u\}$
If $(u^* \in U)$ then Return $\perp$
Return $\mathsf{IBE.UKg}(pp, msk, u)$

proc LR$(u, m_0, m_1)$
If $(c^* \neq \perp)$ then Return $\perp$
If $(|m_0| \neq |m_1|)$ then Return $\perp$
$u^* \leftarrow u$
If $(u^* \in U)$ then Return $\perp$
$c^* \leftarrow_\$ \mathsf{IBE.Enc}(pp, mpk, u^*, m_b)$
Return $c^*$
```

```
main OW-aID$^A_{\mathsf{IBE}}(\lambda)$
$c^* \leftarrow \perp$ ; $u^* \leftarrow \perp$ ; $U \leftarrow \emptyset$
$pp \leftarrow_\$ \mathsf{IBE.Pg}(1^\lambda)$
$m^* \leftarrow_\$ \mathsf{IBE.MSp}(pp)$
$(msk, mpk) \leftarrow_\$ \mathsf{IBE.MKg}(pp)$
$m \leftarrow_\$ A^{\mathrm{KD,C\scriptsize{HAL}}}(pp, mpk)$
Return $(m^* = m)$

proc KD$(u)$
$U \leftarrow U \cup \{u\}$
If $(u^* \in U)$ then Return $\perp$
Return $\mathsf{IBE.UKg}(pp, msk, u)$

proc Chal$(u)$
If $(c^* \neq \perp)$ then Return $\perp$
$u^* \leftarrow u$
If $(u^* \in U)$ then Return $\perp$
$c^* \leftarrow_\$ \mathsf{IBE.Enc}(pp, mpk, u^*, m^*)$
Return $c^*$
```

```
main IND-sID$^A_{\mathsf{IBE}}(\lambda)$
$b \leftarrow_\$ \{0,1\}$ ; $c^* \leftarrow \perp$ ; $u^* \leftarrow \perp$
$pp \leftarrow_\$ \mathsf{IBE.Pg}(1^\lambda)$
$(msk, mpk) \leftarrow_\$ \mathsf{IBE.MKg}(pp)$
$b' \leftarrow_\$ A^{\mathrm{sID,KD,LR}}(pp)$
Return $(b = b')$

proc sID$(u)$
If $(u^* \neq \perp)$ then Return $\perp$
$u^* \leftarrow u$
Return $mpk$

proc KD$(u)$
If $(u^* = \perp)$ then Return $\perp$
If $(u = u^*)$ then Return $\perp$
Return $\mathsf{IBE.UKg}(pp, msk, u)$

proc LR$(m_0, m_1)$
If $(u^* = \perp)$ then Return $\perp$
If $(c^* \neq \perp)$ then Return $\perp$
If $(|m_0| \neq |m_1|)$ then Return $\perp$
$c^* \leftarrow_\$ \mathsf{IBE.Enc}(pp, mpk, u^*, m_b)$
Return $c^*$
```

Figure 2.4: Top left: Game IND-aID defining adaptive-ID indistinguishability of identity-based encryption scheme IBE. Top right: Game OW-aID defining one-wayness. Bottom: Game IND-sID defining selective-ID indistinguishability.

```
main IND-CCA$^A_{\mathsf{KEM}}(\lambda)$
$b \leftarrow\!\$ \{0,1\}$ ; $c^* \leftarrow \bot$
$pp \leftarrow\!\$ \mathsf{KEM.Pg}(1^\lambda)$ ; $(sk, pk) \leftarrow\!\$ \mathsf{KEM.Kg}(pp)$
$(c^*, K^*) \leftarrow\!\$ \mathsf{KEM.Enc}(pp, pk)$
If $(b = 0)$ then $K^* \leftarrow\!\$ \mathsf{KSp}(pp)$
$b' \leftarrow\!\$ A^{\mathrm{Dec}}(pp, c^*, K^*)$
Return $(b = b')$

proc Dec($c$)
If $(c = c^*)$ then Return $\bot$
Return $K \leftarrow \mathsf{KEM.Dec}(pp, sk, c)$
```

Figure 2.5: Game IND-CCA defining indistinguishability of key encapsulation mechanism KEM under chosen-ciphertext attack.

Key Encapsulation Mechanisms. A key encapsulation mechanism KEM specifies the following PT algorithms: via $pp \leftarrow\!\$ \mathsf{KEM.Pg}(1^\lambda)$ one generates public parameters pp common to all users; via $(sk, pk) \leftarrow\!\$ \mathsf{KEM.Kg}(pp)$ a user can generate a secret key sk and corresponding public key pk; via $(c, K) \leftarrow \mathsf{KEM.Enc}(pp, pk)$ anyone can encapsulate in a ciphertext c a random key $K \in \mathsf{KSp}(pp)$; via $K \leftarrow \mathsf{KEM.Dec}(pp, sk, c)$ a user can deterministically decapsulate c to get a value $K \in \mathsf{KEM.KSp}(pp) \cup \{\bot\}$. Correctness requires that $\mathsf{KEM.Dec}(pp, sk, c) = K$ for all $\lambda \in \mathbb{N}$, all $pp \in [\mathsf{KEM.Pg}(1^\lambda)]$, all $(sk, pk) \in [\mathsf{KEM.Kg}(pp)]$, and all $(c, K) \in [\mathsf{KEM.Enc}(pp, pk)]$.

We say a key encapsulation mechanism KEM is *indistinguishable under chosen-ciphertext attack* or *IND-CCA secure* if $\mathbf{Adv}^{\text{ind-cca}}_{\mathsf{KEM},A}(\cdot)$ is negligible for all PT adversaries A, where $\mathbf{Adv}^{\text{ind-cca}}_{\mathsf{KEM},A}(\lambda) = 2\Pr[\text{IND-CCA}^A_{\mathsf{KEM}}(\lambda)] - 1$ and game IND-CCA is in Figure 2.5.

Data Encapsulation Mechanisms. A data encapsulation mechanism DEM specifies the following PT algorithms: via $pp \leftarrow\!\$ \mathsf{DEM.Pg}(1^\lambda)$ one generates public parameters pp common to all users; via $K \leftarrow\!\$ \mathsf{DEM.Kg}(pp)$ a user can generate a secret key K; via $c \leftarrow \mathsf{DEM.Enc}(pp, K, m)$ a user can deterministically generate a ciphertext c encrypting message $m \in \mathsf{DEM.MSp}(pp)$; via $m \leftarrow \mathsf{DEM.Dec}(pp, K, c)$ a user can deterministically decrypt ciphertext c to get a value $m \in \mathsf{DEM.MSp}(pp) \cup \{\bot\}$. Correctness requires that $\mathsf{DEM.Dec}(pp, K, \mathsf{DEM.Enc}(pp, K, m)) = m$ for all $\lambda \in \mathbb{N}$, all $pp \in [\mathsf{DEM.Pg}(1^\lambda)]$, all $K \in [\mathsf{DEM.Kg}(pp)]$, and all $m \in \mathsf{DEM.MSp}(pp)$.

```
main OT-IND-CCA$^A_{\mathsf{DEM}}(\lambda)$
$b \leftarrow_{\$} \{0,1\}$ ; $c^* \leftarrow \perp$
$pp \leftarrow_{\$} \mathsf{DEM.Pg}(1^\lambda)$ ; $K \leftarrow_{\$} \mathsf{DEM.Kg}(pp)$
$b' \leftarrow_{\$} A^{\text{Dec,LR}}(pp)$
Return $(b = b')$

proc Dec$(c)$
If $(c = c^*)$ then Return $\perp$
Return $m \leftarrow \mathsf{DEM.Dec}(pp, K, c)$

proc LR$(m_0, m_1)$
If $(c^* \neq \perp)$ then Return $\perp$
If $(|m_0| \neq |m_1|)$ then Return $\perp$
$c^* \leftarrow_{\$} \mathsf{DEM.Enc}(pp, K, m_b)$
Return $c^*$
```

Figure 2.6: Game OT-IND-CCA defining one-time indistinguishability of data encapsulation mechanism DEM under chosen-ciphertext attack.

We say a data encapsulation mechanism DEM is *one-time indistinguishable under chosen-ciphertext attack* or *OT-IND-CCA secure* if $\mathbf{Adv}^{\text{ot-ind-cca}}_{\mathsf{DEM},A}(\cdot)$ is negligible for all PT adversaries A, where $\mathbf{Adv}^{\text{ot-ind-cca}}_{\mathsf{DEM},A}(\lambda) = 2\Pr[\text{OT-IND-CCA}^A_{\mathsf{DEM}}(\lambda)] - 1$ and game OT-IND-CCA is in Figure 2.6.

Symmetric Encryption Schemes. A symmetric encryption scheme SE specifies the following PT algorithms: via $pp \leftarrow_{\$} \mathsf{SE.Pg}(1^\lambda)$ one generates public parameters pp common to all users; via $K \leftarrow_{\$} \mathsf{SE.Kg}(pp)$ a user can generate a secret key K; via $c \leftarrow_{\$} \mathsf{SE.Enc}(pp, K, m)$ a user can generate a randomised ciphertext c encrypting message $m \in \mathsf{SE.MSp}(pp)$; via $m \leftarrow \mathsf{SE.Dec}(pp, K, c)$ a user can deterministically decrypt ciphertext c to get a value $m \in \mathsf{SE.MSp}(pp) \cup \{\perp\}$. Correctness requires that $\mathsf{SE.Dec}(pp, K, \mathsf{SE.Enc}(pp, K, m)) = m$ for all $\lambda \in \mathbb{N}$, all $pp \in [\mathsf{SE.Pg}(1^\lambda)]$, all $K \in [\mathsf{SE.Kg}(pp)]$, and all $m \in \mathsf{DEM.MSp}(pp)$.

Encapsulation Schemes. An encapsulation scheme [37] can be thought of as a commitment scheme that supports committing to a random rather than chosen string. An encapsulation scheme EC specifies the following PT algorithms: via $pp \leftarrow_{\$} \mathsf{EC.Pg}(1^\lambda)$ one generates public parameters pp common to all users; via $(r, com, dec) \leftarrow_{\$} \mathsf{EC.Enc}(pp)$ anyone can generate a commitment string com and corresponding decommitment string dec encapsulating a random string $r \in \{0,1\}^{\ell(\lambda)}$; via $r \leftarrow \mathsf{EC.Dec}(pp, com, dec)$ anyone can deterministically decapsulate a value $r \in \{0,1\}^{\ell(\lambda)} \cup \{\perp\}$. Correctness requires that $\mathsf{EC.Dec}(pp, com, dec) = r$ for all

```
main HIDE$^A_{\mathsf{EC}}(\lambda)$
$b \leftarrow_{\$} \{0,1\}$
$pp \leftarrow_{\$} \mathsf{EC.Pg}(1^\lambda)$
$r_0 \leftarrow_{\$} \{0,1\}^\lambda$
$(r_1, com, dec) \leftarrow_{\$} \mathsf{EC.Enc}(pp)$
$b' \leftarrow_{\$} A(pp, com, r_b)$
Return $(b = b')$
```

```
main BIND$^A_{\mathsf{EC}}(\lambda)$
$pp \leftarrow_{\$} \mathsf{EC.Pg}(1^\lambda)$
$(r, com, dec) \leftarrow_{\$} \mathsf{EC.Enc}(pp)$
$dec' \leftarrow_{\$} A(pp, com, dec)$
Return $(\mathsf{EC.Dec}(pp, com, dec') \notin \{\perp, r\})$
```

Figure 2.7: Left: Game HIDE defining hiding property of encapsulation scheme EC. Right: Game BIND defining binding property.

$\lambda \in \mathbb{N}$, all $pp \in [\mathsf{EC.Pg}(1^\lambda)]$, and all $(r, com, dec) \in [\mathsf{EC.Enc}(pp)]$.

We say an encapsulation scheme EC is *hiding* if $\mathbf{Adv}^{\mathrm{hide}}_{\mathsf{EC},A}(\cdot)$ is negligible for all PT adversaries A, where $\mathbf{Adv}^{\mathrm{hide}}_{\mathsf{EC},A}(\lambda) = 2\Pr[\mathrm{HIDE}^A_{\mathsf{EC}}(\lambda)] - 1$ and game HIDE is on the left-hand side of Figure 2.7, and *binding* if $\mathbf{Adv}^{\mathrm{bind}}_{\mathsf{EC},A}(\cdot)$ is negligible for all PT adversaries A, where $\mathbf{Adv}^{\mathrm{bind}}_{\mathsf{EC},A}(\lambda) = \Pr[\mathrm{BIND}^A_{\mathsf{EC}}(\lambda)]$ and game BIND is on the right-hand side of Figure 2.7.

Message Authentication Codes. A message authentication code MAC specifies the following PT algorithms: via $pp \leftarrow_{\$} \mathsf{MAC.Pg}(1^\lambda)$ one generates public parameters pp common to all users; via $K \leftarrow_{\$} \mathsf{MAC.Kg}(pp)$ a user can generate a key K; via $\tau \leftarrow \mathsf{MAC.Gen}(pp, K, m)$ a user can generate a message authentication tag τ for message $m \in \mathsf{MAC.MSp}(pp)$; via $d \leftarrow \mathsf{MAC.Verify}(pp, K, m, \tau)$ a user can deterministically produce a decision $d \in \{\mathsf{true}, \mathsf{false}\}$ regarding whether τ is a valid message authentication tag of m. Correctness requires that $\mathsf{MAC.Verify}(pp, K, m, \mathsf{MAC.Gen}(pp, K, m)) = \mathsf{true}$ for all all $\lambda \in \mathbb{N}$, all $pp \in [\mathsf{MAC.Pg}(1^\lambda)]$, all $K \in [\mathsf{MAC.Kg}(pp)]$, and all $m \in \mathsf{MAC.MSp}(pp)$.

We say that a message authentication code MAC is *one-time strongly unforgeable* if $\mathbf{Adv}^{\mathrm{ot\text{-}suf}}_{\mathsf{MAC},A}(\cdot)$ is negligible for all PT adversaries A, where $\mathbf{Adv}^{\mathrm{ot\text{-}suf}}_{\mathsf{MAC},A}(\lambda) = \Pr[\mathrm{OT\text{-}SUF}^A_{\mathsf{MAC}}(\lambda)]$ and game OT-SUF is in Figure 2.8.

Signcryption Schemes. A signcryption scheme combines the functionality of signature and encryption schemes, allowing users to achieve message confidentiality and origin authentication through one operation. Signcryption can be defined so that each user has two keypairs, one for when the user plays the role of sender and the other for for when he plays the role of receiver, however our work con-

```
main OT-SUF$^A_{\mathsf{MAC}}(\lambda)$
$m^* \leftarrow \perp$ ; $\tau^* \leftarrow \perp$
$pp \leftarrow\$ \mathsf{MAC.Pg}(1^\lambda)$
$K \leftarrow\$ \mathsf{MAC.Kg}(pp)$
$(m, \tau) \leftarrow\$ A(pp)$
Return $(\mathsf{MAC.Verify}(pp, K, m, \tau) \wedge ((m, \tau) \neq (m^*, \tau^*)))$

proc Gen($m$)
If $(\tau^* \neq \perp)$ then Return $\perp$
$m^* \leftarrow m$
$\tau^* \leftarrow \mathsf{MAC.Gen}(pp, K, m^*)$
Return $\tau^*$
```

Figure 2.8: Game OT-SUF defining one-time strong unforgeability of message authentication scheme MAC.

cerns the case where a user has a single keypair used for both sending and receiving messages, so we present here a single keypair definition. A signcryption scheme SC specifies the following PT algorithms: via $pp \leftarrow\$ \mathsf{SC.Pg}(1^\lambda)$ one generates public parameters pp common to all users; via $(sk, pk) \leftarrow\$ \mathsf{SC.Kg}(pp)$ a user can generate a secret key sk and corresponding public key pk (to clarify which role a keypair is used in, we attach the subscript r when used as a receiver keypair, i.e. (pk_r, sk_r), and attach the subscript s when used as a sender keypair, i.e. (pk_s, sk_s)); via $c \leftarrow\$ \mathsf{SC.Sc}(pp, sk_s, pk_s, pk_r, m)$ the sender can generate a ciphertext c encrypting message m in the message space $\mathsf{SC.MSp}(pp)$ under the receiver's public key pk_r; via $m \leftarrow \mathsf{SC.Usc}(pp, sk_r, pk_r, pk_s, c)$ the receiver can deterministically decrypt ciphertext c to get a value $m \in \mathsf{SC.MSp}(pp) \cup \{\perp\}$ Correctness requires that $\mathsf{SC.Usc}(pp, sk_r, pk_r, pk_s, \mathsf{SC.Sc}(pp, sk_s, pk_s, pk_r, m)) = m$ for all $\lambda \in \mathbb{N}$, all $pp \in [\mathsf{SC.Pg}(1^\lambda)]$, all $(sk_s, pk_s) \in [\mathsf{SC.Kg}(pp)]$, all $(sk_r, pk_r) \in [\mathsf{SC.Kg}(pp)]$, and all $m \in \mathsf{SC.MSp}(pp)$.

A number of security models capturing different levels of security have been proposed for signcryption (e.g. see [89] for an overview). The main differences between these models concern whether or not the adversary is considered to be an insider with the knowledge of the secret key material of the challenge sender and challenge receiver in the definition of confidentiality and unforgeability, respectively, and to what extent the adversary is allowed to maliciously generate the keys of the users in the system. We focus on the strongest security model of these, which captures the notions of insider confidentiality and insider unforgeability in the multi-user setting.

Multi-user indistinguishability under insider chosen-ciphertext attack (MU-IND-

```
main MU-IND-iCCA$^A_{\mathsf{SC}}(\lambda)$
$b \leftarrow\!\$\ \{0,1\}$ ; $c^* \leftarrow \perp$ ; $pk_s^* \leftarrow \perp$
$pp \leftarrow\!\$\ \mathsf{SC.Pg}(1^\lambda)$ ; $(sk, pk) \leftarrow\!\$\ \mathsf{SC.Kg}(pp)$
$b' \leftarrow\!\$\ A^{\text{Sc},\text{Usc},\text{LR}}(pp, pk)$
Return $(b = b')$

proc Sc$(pk_r, m)$
Return $m \leftarrow \mathsf{SC.Sc}(pp, sk, pk, pk_r, m)$

proc Usc$(pk_s, c)$
If $((c = c^*) \wedge (pk_s = pk_s^*))$ then Return $\perp$
Return $\mathsf{SC.Usc}(pp, sk, pk, pk_s, c)$

proc LR$(sk_s, pk_s, m_0, m_1)$
If $(c^* \neq \perp)$ then Return $\perp$
If $(|m_0| \neq |m_1|)$ then Return $\perp$
$pk_s^* \leftarrow pk_s$
$c^* \leftarrow\!\$\ \mathsf{SC.Sc}(pp, sk_s, pk_s, pk, m_b)$
Return $c^*$
```

Figure 2.9: Game MU-IND-iCCA defining multi-user indistinguishability of signcryption scheme SC under insider chosen-ciphertext attack.

iCCA) captures the property that an adversary cannot distinguish between the signcryption of two different messages for a challenge receiver, even though all other keys in the system are maliciously generated, and the adversary is given access to signcryption and unsigncryption oracles. We say that SC is MU-IND-iCCA-secure if $\mathbf{Adv}^{\text{mu-ind-icca}}_{\mathsf{SC},A}(\cdot)$ is negligible for all PT adversaries A, where $\mathbf{Adv}^{\text{mu-ind-icca}}_{\mathsf{SC},A}(\lambda) = 2\Pr[\text{MU-IND-iCCA}^A_{\mathsf{SC}}(\lambda)] - 1$ and game MU-IND-iCCA is in Figure 2.9. Since this is a CCA notion the adversary has access to a Usc oracle unsigncrypting ciphertexts under the challenge keypair, i.e. having the challenge keypair in the receiver role. Since the same keypair is used in both sender and receiver roles, the adversary here also has access to a Sc oracle where the challenge keypair is used in the sender role.

Multi-user existential unforgeability under insider chosen-message attack (MU-EUF-iCMA) captures the property that an adversary cannot create a valid ciphertext from a challenge sender which contains a new message, even if all other keys in the system are maliciously generated and the adversary is given access to signcryption and unsigncryption oracles. We say that SC is MU-EUF-iCMA-secure if $\mathbf{Adv}^{\text{mu-euf-icma}}_{\mathsf{SC},A}(\cdot)$ is negligible for all PT adversaries A, where $\mathbf{Adv}^{\text{mu-euf-icma}}_{\mathsf{SC},A}(\lambda) = \Pr[\text{MU-EUF-iCMA}^A_{\mathsf{SC}}(\lambda)]$ and game MU-EUF-iCMA is in Figure 2.10. Since this is a CMA notion the adversary has access to a Sc oracle encrypting messages under the challenge keypair, i.e. having the challenge keypair in the sender role. Since the

```
main MU-EUF-iCMA$_{\mathsf{SC}}^{A}(\lambda)$
$Q \leftarrow \emptyset$
$pp \leftarrow_{\$} \mathsf{SC.Pg}(1^\lambda)$ ; $(sk, pk) \leftarrow_{\$} \mathsf{SC.Kg}(pp)$
$(sk_r, pk_r, c) \leftarrow_{\$} A^{\text{Sc},\text{Usc}}(pp, pk)$
$m \leftarrow \mathsf{SC.Usc}(pp, sk_r, pk_r, pk, c)$
Return $((m \in \mathsf{SC.MSp}(pp)) \wedge ((pk_r, m) \notin Q))$

proc Sc$(pk_r, m)$
$c \leftarrow \mathsf{SC.Sc}(pp, sk, pk, pk_r, m)$
$Q \leftarrow Q \cup \{(pk_r, m)\}$
Return $c$

proc Usc$(pk_s, c)$
Return $\mathsf{SC.Usc}(pp, sk, pk, pk_s, c)$
```

Figure 2.10: Game MU-EUF-iCMA defining multi-user existential unforgeability of signcryption scheme SC under insider chosen-message attack.

same keypair is used in both sender and receiver roles, the adversary here also has access to a Usc oracle where the challenge keypair is used in the receiver role.

NIZK systems. Suppose R: $\{0,1\}^* \times \{0,1\}^* \to \{\mathsf{true}, \mathsf{false}\}$. For $x \in \{0,1\}^*$ we let $\mathsf{R}(x) = \{\ w\ :\ \mathsf{R}(x,w) = \mathsf{true}\ \}$ be the *witness set* of x. We say that R is an **NP**-relation if it is computable in time polynomial in the length of its first input and there is a function ℓ such that $\mathsf{R}(x) \subseteq \{0,1\}^{\ell(|x|)}$ for all $x \in \{0,1\}^*$. We let $L(\mathsf{R}) = \{\ x\ :\ \mathsf{R}(x) \neq \emptyset\ \}$ be the *language* associated to R. The fact that R is an **NP**-relation means that $L(\mathsf{R}) \in \mathbf{NP}$.

A non-interactive (NI) system NIZK for relation R specifies the following PT algorithms: via $crs \leftarrow_{\$} \mathsf{NIZK.Pg}(1^\lambda)$ one generates a common reference string crs; via $\pi \leftarrow_{\$} \mathsf{NIZK.P}(crs, x, w)$ the prover given x and $w \in \mathsf{R}(x)$ generates a proof π that $x \in L(\mathsf{R})$; via $d \leftarrow \mathsf{NIZK.V}(crs, x, \pi)$ a verifier can produce a decision $d \in \{\mathsf{true}, \mathsf{false}\}$ regarding whether π is a valid proof that $x \in L(\mathsf{R})$. We require completeness, namely $\mathsf{NIZK.V}(crs, x, \mathsf{NIZK.P}(crs, x, w)) = \mathsf{true}$ for all $\lambda \in \mathbb{N}$, all $crs \in [\mathsf{NIZK.Pg}(1^\lambda)]$, all $x \in \{0,1\}^*$ and all $w \in \mathsf{R}(x)$. We say that NIZK is zero-knowledge (ZK) if it specifies additional PT algorithms $\mathsf{NIZK.SimPg}$ and $\mathsf{NIZK.SimP}$ such that $\mathbf{Adv}^{\mathrm{zk}}_{\mathsf{NIZK},\mathsf{R},A}(\cdot)$ is negligible for every PT adversary A, where $\mathbf{Adv}^{\mathrm{zk}}_{\mathsf{NIZK},\mathsf{R},A}(\lambda) = 2\Pr[\mathrm{ZK}^{A}_{\mathsf{NIZK},\mathsf{R}}(\lambda)] - 1$ and game ZK is specified on the left-hand side of Figure 2.11. This definition is based on [28, 50]. We say that NIZK is simulation-extractable (SE) if it specifies an additional PT algorithm $\mathsf{NIZK.Ext}$ such that $\mathbf{Adv}^{\mathrm{se}}_{\mathsf{NIZK},\mathsf{R},A}(\cdot)$ is negligible for every PT adversary A, where $\mathbf{Adv}^{\mathrm{se}}_{\mathsf{NIZK},\mathsf{R},A}(\lambda) = \Pr[\mathrm{SE}^{A}_{\mathsf{NIZK},\mathsf{R}}(\lambda)]$ and game SE is specified

```
main $\mathrm{ZK}^{A}_{\mathsf{NIZK},\mathsf{R}}(\lambda)$
$b \leftarrow_{\$} \{0,1\}$
$crs_1 \leftarrow_{\$} \mathsf{NIZK.Pg}(1^{\lambda})$
$(crs_0, std, xtd) \leftarrow_{\$} \mathsf{NIZK.SimPg}(1^{\lambda})$
$b' \leftarrow_{\$} A^{\textsc{Prove}}(crs_b)$
Return $(b = b')$

$\textsc{Prove}(x, w)$
If $(!\mathsf{R}(x, w))$ then Return $\perp$
If $(b = 1)$ then $\pi \leftarrow_{\$} \mathsf{NIZK.P}(crs_1, x, w)$
Else $\pi \leftarrow_{\$} \mathsf{NIZK.SimP}(crs_0, std, x)$
Return $\pi$
```

```
main $\mathrm{SE}^{A}_{\mathsf{NIZK},\mathsf{R}}(\lambda)$
$Q \leftarrow \emptyset$
$(crs, std, xtd) \leftarrow_{\$} \mathsf{NIZK.SimPg}(1^{\lambda})$
$(x, \pi) \leftarrow_{\$} A^{\textsc{Prove}}(crs)$
If $(x \notin L(\mathsf{R}))$ then Return $\mathsf{false}$
If $(!\mathsf{NIZK.V}(crs, x, \pi))$ then Return $\mathsf{false}$
If $((x, \pi) \in Q)$ then Return $\mathsf{false}$
$w \leftarrow_{\$} \mathsf{NIZK.Ext}(crs, xtd, x, \pi)$
Return $!\mathsf{R}(x, w)$

$\textsc{Prove}(x, w)$
If $(!\mathsf{R}(x, w))$ then Return $\perp$
$\pi \leftarrow_{\$} \mathsf{NIZK.SimP}(crs, std, x)$
$Q \leftarrow Q \cup \{(x, \pi)\}$
Return $\pi$
```

Figure 2.11: Left: Game ZK defining zero knowledge property of NIZK system NIZK. Right: Game SE defining simulation extractability.

on the right-hand side of Figure 2.11. This definition is based on [50, 70, 71, 56].

The first construction of SE NIZKs (using a stronger notion of simulation extractability) was given in [70], but for a fairly restricted language related to sets of pairing product equations in bilinear groups. In [56] (and further formalised in [73]), the authors provide a generic construction of SE NIZKs from a (regular) NIZK, an IND-CCA-secure encryption scheme, and a one-time signature, which establishes that SE NIZKs exist for all **NP**.

2.3 Constructions

The CHK transform [32] constructs a PKE scheme from an IBE scheme IBE and a signature scheme DS. Figure 2.12 shows the algorithms of the PKE scheme CHK[IBE, DS]. The resulting PKE scheme is IND-CCA secure though the IBE scheme need only be IND-CPA secure.

Theorem 2.3.1 ([32]) *Let* IBE *be a selectively-secure IBE scheme and* DS *be a one-time strongly unforgeable signature scheme. Then* CHK[IBE, DS] *is IND-CCA secure.*

```
$\underline{\mathsf{CHK[IBE,DS].Pg}(1^\lambda):}$
$pp_{\mathsf{IBE}} \leftarrow\!\!\$\ \mathsf{IBE.Pg}(1^\lambda)$
$pp_{\mathsf{DS}} \leftarrow\!\!\$\ \mathsf{DS.Pg}(1^\lambda)$
Return $(pp_{\mathsf{IBE}}, pp_{\mathsf{DS}})$

$\underline{\mathsf{CHK[IBE,DS].Enc}(pp, pk, m):}$
$(pp_{\mathsf{IBE}}, pp_{\mathsf{DS}}) \leftarrow pp$ ; $mpk \leftarrow pk$
$(sk_{\mathsf{DS}}, pk_{\mathsf{DS}}) \leftarrow\!\!\$\ \mathsf{DS.Kg}(pp_{\mathsf{DS}})$
$u \leftarrow pk_{\mathsf{DS}}$
$c_{\mathsf{IBE}} \leftarrow\!\!\$\ \mathsf{IBE.Enc}(pp_{\mathsf{IBE}}, mpk, u, m)$
$\sigma_{\mathsf{DS}} \leftarrow\!\!\$\ \mathsf{DS.Sign}(pp_{\mathsf{DS}}, sk_{\mathsf{DS}}, c_{\mathsf{IBE}})$
Return $(pk_{\mathsf{DS}}, c_{\mathsf{IBE}}, \sigma_{\mathsf{DS}})$

$\underline{\mathsf{CHK[IBE,DS].Kg}(pp):}$
$(pp_{\mathsf{IBE}}, pp_{\mathsf{DS}}) \leftarrow pp$
$(msk, mpk) \leftarrow\!\!\$\ \mathsf{IBE.MKg}(pp_{\mathsf{IBE}})$
Return $(msk, mpk)$

$\underline{\mathsf{CHK[IBE,DS].Dec}(pp, sk, c):}$
$(pk_{\mathsf{DS}}, c_{\mathsf{IBE}}, \sigma_{\mathsf{DS}}) \leftarrow c$ ; $msk \leftarrow sk$
If $(!\mathsf{DS.Verify}(pp_{\mathsf{DS}}, pk_{\mathsf{DS}}, c_{\mathsf{IBE}}, \sigma_{\mathsf{DS}}))$
    then Return $\bot$
$u \leftarrow pk_{\mathsf{DS}}$
$usk \leftarrow\!\!\$\ \mathsf{IBE.UKg}(pp_{\mathsf{IBE}}, msk, u)$
Return $\mathsf{IBE.Dec}(pp_{\mathsf{IBE}}, usk, c_{\mathsf{IBE}})$
```

Figure 2.12: PKE scheme CHK[IBE, DS].

```
$\underline{\mathsf{HPKE[KEM,DEM].Pg}(1^\lambda):}$
$pp_{\mathsf{KEM}} \leftarrow\!\!\$\ \mathsf{KEM.Pg}(1^\lambda)$
$pp_{\mathsf{DEM}} \leftarrow\!\!\$\ \mathsf{DEM.Pg}(1^\lambda)$
Return $(pp_{\mathsf{KEM}}, pp_{\mathsf{DEM}})$

$\underline{\mathsf{HPKE[KEM,DEM].Kg}(pp):}$
$(pp_{\mathsf{KEM}}, pp_{\mathsf{DEM}}) \leftarrow pp$
$(sk, pk) \leftarrow\!\!\$\ \mathsf{KEM.Kg}(pp_{\mathsf{KEM}})$
Return $(sk, pk)$

$\underline{\mathsf{HPKE[KEM,DEM].Enc}(pp, pk, m):}$
$(pp_{\mathsf{KEM}}, pp_{\mathsf{DEM}}) \leftarrow pp$
$(c_1, K) \leftarrow\!\!\$\ \mathsf{KEM.Enc}(pp_{\mathsf{KEM}}, pk)$
$c_2 \leftarrow \mathsf{DEM.Enc}(K, m)$
Return $(c_1, c_2)$

$\underline{\mathsf{HPKE[KEM,DEM].Dec}(pp, sk, c):}$
$(pp_{\mathsf{KEM}}, pp_{\mathsf{DEM}}) \leftarrow pp$ ; $(c_1, c_2) \leftarrow c$
$K \leftarrow \mathsf{KEM.Dec}(pp_{\mathsf{KEM}}, sk, c_1)$
Return $\mathsf{DEM.Dec}(pp_{\mathsf{DEM}}, K, c_2)$
```

Figure 2.13: Hybrid PKE scheme HPKE[KEM, DEM].

The KEM-DEM paradigm of [49] constructs a *hybrid encryption* scheme from a KEM KEM and a DEM DEM. Figure 2.13 shows the algorithms of the PKE scheme HPKE[KEM, DEM].

Theorem 2.3.2 ([49]) *Let* KEM *be an IND-CCA-secure KEM and* DEM *be a OT-IND-CCA-secure DEM. Then the hybrid encryption scheme* HPKE[KEM, DEM] *is IND-CCA secure.*

2.4 Bilinear Pairings

Let $\mathbb{G}_1 = \langle g_1 \rangle$, $\mathbb{G}_2 = \langle g_2 \rangle$, $\mathbb{G}_T$ be groups of prime order p. A pairing is a map $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$ that satisfies the following properties:

(1) Bilinear: For all $a, b \in \mathbb{Z}, e(g_1^a, g_2^b) = e(g_1, g_2)^{ab}$.

(2) Non-degenerate: $e(g_1, g_2) \neq 1$.

(3) Computable: There is an efficient algorithm to compute the map e.

We assume the existence of a group generator G with an efficient algorithm $\mathsf{G.Pg}(\cdot)$ that outputs $(p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e)$ as described above, appropriately for the security parameter taken as input.

When $\mathbb{G}_1 = \mathbb{G}_2$ we write both groups as $\mathbb{G}$; this is called a symmetric pairing. At higher security levels (128 bits and above), asymmetric pairings are far more efficient both in terms of computation and in terms of the size of group elements [61]. As a concrete example, using BN curves [12] and sextic twists, we can attain the 128-bit security level with elements of $\mathbb{G}_1$ being represented by 256 bits and elements of $\mathbb{G}_2$ needing 512 bits. By exploiting compression techniques [95], elements of $\mathbb{G}_T$ in this case can be represented using 1024 bits. For further details on parameter selection for pairings, see [60].

We will make use of the following problems, presented here in either the symmetric or asymmetric setting as needed.

Bilinear Diffie-Hellman problem. We say the BDH problem is hard for G if $\mathbf{Adv}^{\mathrm{bdh}}_{\mathsf{G},A}(\cdot)$ is negligible for all PT adversaries A, where $\mathbf{Adv}^{\mathrm{bdh}}_{\mathsf{G},A}(\lambda) = \Pr[\mathrm{BDH}^A_{\mathsf{G}}(\lambda)]$ and game BDH is on the left-hand side of Figure 2.14.

Decisional Bilinear Diffie-Hellman problem. We say the DBDH problem is hard for G if $\mathbf{Adv}^{\mathrm{dbdh}}_{\mathsf{G},A}(\cdot)$ is negligible for all PT adversaries A, where $\mathbf{Adv}^{\mathrm{dbdh}}_{\mathsf{G},A}(\lambda) = 2\Pr[\mathrm{BDH}^A_{\mathsf{G}}(\lambda)] - 1$ and game DBDH is on the right-hand side of Figure 2.14.

Decisional Bilinear Diffie-Hellman Inversion problem. We say the DBDHI problem is hard for G if $\mathbf{Adv}^{\mathrm{dbdhi}}_{\mathsf{G},q(\cdot),A}(\cdot)$ is negligible for all PT adversaries A and every polynomially bounded function $q : \mathbb{N} \mapsto \mathbb{N}$, where $\mathbf{Adv}^{\mathrm{dbdhi}}_{\mathsf{G},q(\lambda),A}(\lambda) = 2\Pr[\mathrm{DBDHI}^A_{\mathsf{G},q(\lambda)}(\lambda)] - 1$ and game DBDHI is in Figure 2.15.

```
main $\mathrm{BDH}^A_{\mathsf{G}}(\lambda)$
$(p, \mathbb{G}, \mathbb{G}_T, e) \leftarrow_{\$} \mathsf{G.Pg}(1^\lambda)$
$\alpha, \beta, \gamma \leftarrow_{\$} \mathbb{Z}_p^*$
$g \leftarrow_{\$} \mathbb{G}^*$
$T \leftarrow_{\$} A(p, \mathbb{G}, \mathbb{G}_T, e, g, g^\alpha, g^\beta, g^\gamma)$
Return $(T = e(g,g)^{\alpha\beta\gamma})$
```

```
main $\mathrm{DBDH}^A_{\mathsf{G}}(\lambda)$
$(p, \mathbb{G}, \mathbb{G}_T, e) \leftarrow_{\$} \mathsf{G.Pg}(1^\lambda)$
$b \leftarrow_{\$} \{0,1\}$
$\alpha, \beta, \gamma \leftarrow_{\$} \mathbb{Z}_p^*$
$g \leftarrow_{\$} \mathbb{G}^*$
If $(b = 1)$ then $T \leftarrow e(g,g)^{\alpha\beta\gamma}$
Else $T \leftarrow_{\$} \mathbb{G}_T$
$b' \leftarrow_{\$} A(p, \mathbb{G}, \mathbb{G}_T, e, g, g^\alpha, g^\beta, g^\gamma, T)$
Return $(b = b')$
```

Figure 2.14: Left: Game BDH defining the Bilinear Diffie-Hellman problem for a group generator G. Right: Game DBDH defining the Decisional Bilinear Diffie-Hellman problem.

```
main $\mathrm{DBDHI}^A_{\mathsf{G},q(\lambda)}(\lambda)$
$q \leftarrow q(\lambda)$
$(p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e) \leftarrow_{\$} \mathsf{G.Pg}(1^\lambda)$
$b \leftarrow_{\$} \{0,1\}$
$\alpha \leftarrow_{\$} \mathbb{Z}_p^*$
$g_1 \leftarrow_{\$} \mathbb{G}_1^*$ ; $g_2 \leftarrow_{\$} \mathbb{G}_2^*$
If $(b = 1)$ then $T \leftarrow e(g_1, g_2)^{1/\alpha}$
Else $T \leftarrow_{\$} \mathbb{G}_T$
$b' \leftarrow_{\$} A^{\textsc{Chal}}(p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e, g_1, g_1^\alpha, g_2, g_2^\alpha, g_2^{\alpha^2}, \ldots, g_2^{\alpha^q}, T)$
Return $(b = b')$
```

Figure 2.15: Game DBDHI defining the Decisional Bilinear Diffie-Hellman Inversion problem for a group generator G.

Strong Diffie-Hellman problem. We say the SDH problem is hard for G if $\mathbf{Adv}^{\mathrm{sdh}}_{\mathsf{G},q(\cdot),A}(\cdot)$ is negligible for all PT adversaries A and every polynomially bounded function $q : \mathbb{N} \mapsto \mathbb{N}$, where $\mathbf{Adv}^{\mathrm{sdh}}_{\mathsf{G},q(\lambda),A}(\lambda) = \Pr[\mathrm{SDH}^A_{\mathsf{G},q(\lambda)}(\lambda)]$ and game SDH is in Figure 2.16.

```
main $\mathrm{SDH}^{A}_{\mathsf{G},q(\lambda)}(\lambda)$
$q \leftarrow q(\lambda)$
$(p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e) \leftarrow\!\!\$\ \mathsf{G.Pg}(1^\lambda)$
$\alpha \leftarrow\!\!\$\ \mathbb{Z}_p^*$
$g_1 \leftarrow\!\!\$\ \mathbb{G}_1^*$ ; $g_2 \leftarrow\!\!\$\ \mathbb{G}_2^*$
$(c, g) \leftarrow\!\!\$\ A^{\textsc{Chal}}(p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e, g_1, g_1^\alpha, g_2, g_2^\alpha, g_2^{\alpha^2}, \ldots, g_2^{\alpha^q})$
Return $(g = g_2^{1/\alpha+c})$
```

Figure 2.16: Game SDH defining the Strong Diffie-Hellman problem for a group generator G.

Chapter 3

Joint Encryption and Signature

Contents

In this chapter we revisit the topic of joint encryption and signature schemes, where a single keypair is used for both encryption and signature in a secure manner. While breaking the principle of key separation, such schemes have attractive properties and are sometimes used in practice. We give a general construction for a joint encryption and signature scheme that uses IBE as a component, and that is secure in the standard model. We then provide a more efficient direct construction, also secure in the standard model. Finally, we show how these results relate to signcryption.

3.1 Introduction

The folklore principle of key separation dictates using different keys for different cryptographic operations. While this is well-motivated by real-world, security engineering concerns, there are still situations where it is desirable to use the same key for multiple operations [72]. In the context of public-key cryptography, using the same keypair for both encryption and signature primitives can reduce storage requirements (for certificates as well as keys), reduce the cost of key certification and the time taken to verify certificates, and reduce the footprint of cryptographic code. These savings may be critical in embedded systems and low-end smart card applications. As a prime example, the globally-deployed EMV standard for authenticating credit and debit card transactions allows the same keypair to be reused for encryption and signatures for precisely these reasons [58].

However, this approach of reusing keys is not without its problems. For example, there is the issue that encryption and signature keypairs may have different lifetimes, or that the private keys may require different levels of protection [72]. Most importantly of all, there is the question of whether it is *secure* to use the same keypair in two (or more) different primitives – perhaps the two uses will interact with one another badly, in such a way as to undermine the security of one or both of the primitives. In the case of textbook RSA, it is obvious that using the same keypair for decryption and signing is dangerous, since the signing and decryption functions are so closely related in this case. Security issues may still arise even if some standardised padding is used prior to encryption and signing [79]. Later we will provide another example in the context of encryption and signature primitives, where the individual components are secure (according to the usual notions of security for encryption and signature) but become completely insecure as soon as they are used in combination with one another. At the protocol level, Kelsey, Schneier and Wagner [77] gave examples of protocols that are individually secure, but that interact badly when a keypair is shared between them.

The formal study of the security of key reuse was initiated by Haber and Pinkas [72]. They introduced the concept of a *combined public key scheme*. Here, an encryption scheme and signature scheme are combined: the existing algorithms to encrypt, decrypt, sign and verify are preserved, but the two key generation algorithms are modified to produce a single algorithm. This algorithm outputs two keypairs, one for the encryption scheme and one for the signature scheme, with the keypairs no longer necessarily being independent. Indeed, under certain conditions, the two keypairs may be identical, in which case the savings described above may be realised. In other cases, the keypairs are not identical but can have some shared components, leading to more modest savings. Haber and Pinkas also introduced the natural security model for combined public key schemes, where the adversary against the encryption part of the scheme is equipped with a signature oracle in addition to the usual decryption oracle, and where the adversary against the signature part of the scheme is given a decryption oracle in addition to the usual signature oracle. In this setting, we talk about the *joint security* of the combined scheme.

While Haber and Pinkas considered various well-known concrete schemes and conditions under which their keys could be partially shared, none of their examples having provable security in the standard model lead to *identical* keypairs for both encryption and signature. Indeed, while the approach of Haber and Pinkas can be made to work in the random oracle model by careful oracle programming and domain separation, their approach does not naturally extend to the standard model. More specifically, in their approach, to be able to simulate the signing oracle in the IND-CCA security game, the public key of the combined scheme cannot be exactly the same as the public key of the underlying encryption scheme (otherwise, successful simulation would lead to a signature forgery). This makes it hard to achieve full effective overlap between the public keys for signing and encryption. For the (standard model) schemes considered by Haber and Pinkas this results in the requirements that part of the public key be specific to the encryption scheme and that another part of it be specific to the signature scheme. Furthermore, at the time of publication of [72] only a few secure (IND-CCA, resp. EUF-CMA) and efficient standard-model schemes were known. Consequently, no "compatible" signature and encryption schemes were identified in [72] for the standard model.

The special case of combined public key schemes built from trapdoor permutations was considered in [47, 81]. Here, both sets of authors considered the use of various message padding schemes in conjunction with an arbitrary trapdoor permutation to build schemes having joint security. Specifically, Coron et al. [47] considered the case of PSS-R encoding, while Komano and Ohta [81] considered the cases of OAEP+ and REACT encodings. All of the results in these two papers are in the random oracle model.

In further related, but distinct, work, Dodis et al. [55] (see also [54]) considered the use of message padding schemes and trapdoor permutations to build *signcryption* schemes. These offer the combined security properties of encryption and signature in a single cryptographic transform (as opposed to the notion of a combined public key scheme which offers these security properties separately, but with a common key generation algorithm). Dodis et al. showed, again in the random oracle model, how to build efficient, secure signcryption schemes in which each user's keypair, specifying a permutation and its trapdoor, is used for both encryption and signing purposes. They were careful to point out that the previous results of [47, 81] concerning combined public key schemes do not immediately imply similar results in the more complicated signcryption setting, nor can they be immediately applied to construct secure signcryption schemes via the generic composition methods of Dodis et al. [3].

Further work on combined public key schemes in the random oracle model, for both the normal public-key setting and the identity-based setting can be found in [68]. In particular, it is proved that the identity-based encryption scheme of Boneh and Franklin [35] and the identity-based signature scheme of Hess [74] can be used safely together.

3.2 Joint Encryption and Signature Schemes

We consider the case of a combined public key scheme where the keypairs are identical, and call this a *joint encryption and signature (JES) scheme.* A JES scheme JES specifies the following PT algorithms: via $pp \leftarrow_{\$} \mathsf{JES.Pg}(1^\lambda)$ one generates public parameters pp common to all users; via $(sk, pk) \leftarrow_{\$} \mathsf{JES.Kg}(pp)$ a user can generate a decryption and signing key sk and corresponding encryption and verification key pk; via $c \leftarrow_{\$} \mathsf{JES.Enc}(pp, pk, m)$ anyone can generate a ciphertext c encrypting message m in the encryption message space $\mathsf{JES.EMSp}(pp)$ under pk; via $m \leftarrow \mathsf{JES.Dec}(pp, sk, c)$ the user can deterministically decrypt ciphertext c to get a value $m \in \mathsf{JES.EMSp}(pp) \cup \{\perp\}$; via $\sigma \leftarrow_{\$} \mathsf{JES.Sign}(pp, sk, m)$ the signer can generate a signature σ on a message m in the signature message space $\mathsf{JES.SMSp}(pp)$; via $d \leftarrow \mathsf{JES.Verify}(pp, pk, m, \sigma)$ a verifier can deterministically produce a decision $d \in \{\mathsf{true}, \mathsf{false}\}$ regarding whether σ is a valid signature of m under pk. Correctness requires that $\mathsf{JES.Dec}(pp, sk, \mathsf{JES.Enc}(pp, pk, m)) = m$ for all $\lambda \in \mathbb{N}$, all $pp \in [\mathsf{JES.Pg}(1^\lambda)]$, all $(sk, pk) \in [\mathsf{JES.Kg}(pp)]$, and all $m \in \mathsf{JES.EMSp}(pp)$, and that $\mathsf{JES.Verify}(pp, pk, m, \mathsf{JES.Sign}(pp, sk, m)) = \mathsf{true}$ for all $\lambda \in \mathbb{N}$, all $pp \in [\mathsf{JES.Pg}(1^\lambda)]$, all $(sk, pk) \in [\mathsf{JES.Kg}(pp)]$, and all $m \in \mathsf{JES.SMSp}(pp)$.

Since the encryption and signature schemes share a keypair the standard notions of IND-CCA and EUF-CMA security need to be extended to reflect an adversary's ability to request both decryptions and signatures under the challenge public key. When defining a security game against a component of the scheme the nature of any additional oracles depends on the required security of the other components. For example, if EUF-CMA security of the signature component of a JES scheme is required, then it is necessary to provide the adversary with unrestricted access to a signature oracle when proving IND-CCA security of the encryption component of the scheme. The security definitions given implicitly in [47], considering IND-CCMA and EUF-CCMA security of a JES scheme, are stated formally here.

We say that JES is IND-CCMA secure if $\mathbf{Adv}^{\text{ind-ccma}}_{\mathsf{JES},A}(\cdot)$ is negligible for all PT adver-

```
main IND-CCMA$^A_{\mathsf{JES}}(\lambda)$
$b \leftarrow_\$ \{0,1\}$ ; $c^* \leftarrow \perp$
$pp \leftarrow_\$ \mathsf{JES.Pg}(1^\lambda)$
$(sk, pk) \leftarrow_\$ \mathsf{JES.Kg}(pp)$
$b' \leftarrow_\$ A^{\textsc{Dec},\textsc{Sign},\mathrm{LR}}(pp, pk)$
Return $(b = b')$

proc Dec($c$)
If $(c = c^*)$ then Return $\perp$
Return $m \leftarrow \mathsf{JES.Dec}(pp, sk, c)$

proc Sign($m$)
Return $\sigma \leftarrow_\$ \mathsf{JES.Sign}(pp, sk, m)$

proc LR($m_0, m_1$)
If $(c^* \neq \perp)$ then Return $\perp$
If $(|m_0| \neq |m_1|)$ then Return $\perp$
$c^* \leftarrow_\$ \mathsf{JES.Enc}(pp, pk, m_b)$
Return $c^*$
```

```
main EUF-CCMA$^A_{\mathsf{JES}}(\lambda)$
$Q \leftarrow \emptyset$
$pp \leftarrow_\$ \mathsf{JES.Pg}(1^\lambda)$
$(sk, pk) \leftarrow_\$ \mathsf{JES.Kg}(pp)$
$(m, \sigma) \leftarrow_\$ A^{\textsc{Sign},\textsc{Dec}}(pp, pk)$
Return $(\mathsf{JES.Verify}(pp, pk, m, \sigma) \wedge m \notin Q)$

proc Sign($m$)
$\sigma \leftarrow_\$ \mathsf{JES.Sign}(pp, sk, m)$
$Q \leftarrow Q \cup \{m\}$
Return $\sigma$

proc Dec($c$)
Return $m \leftarrow \mathsf{JES.Dec}(pp, sk, c)$
```

Figure 3.1: Left: Game IND-CCMA defining indistinguishability of joint encryption and signature scheme JES under chosen-ciphertext attack in the presence of a signing oracle. Right: Game EUF-CCMA defining existential unforgeability under chosen-message attack in the presence of a decryption oracle.

saries A, where $\mathbf{Adv}^{\text{ind-ccma}}_{\mathsf{JES},A}(\lambda) = 2\Pr[\text{IND-CCMA}^A_{\mathsf{JES}}(\lambda)] - 1$ and game IND-CCMA is on the left-hand side of Figure 3.1. This represents indistinguishability under chosen-ciphertext attack in the presence of a signing oracle.

We say that JES is EUF-CCMA secure if $\mathbf{Adv}^{\text{euf-ccma}}_{\mathsf{JES},A}(\cdot)$ is negligible for all PT adversaries A, where $\mathbf{Adv}^{\text{euf-ccma}}_{\mathsf{JES},A}(\lambda) = \Pr[\text{EUF-CCMA}^A_{\mathsf{JES}}(\lambda)]$ and game EUF-CCMA is on the right-hand side of Figure 3.1. This represents existential unforgeability under chosen-message attack in the presence of a decryption oracle.

Informally, we say that a JES scheme is *jointly secure* if it is both IND-CCMA secure and EUF-CCMA secure.

3.3 A Cartesian Product Construction

A trivial way of obtaining a system satisfying the above security properties is to concatenate the keys of an encryption scheme and signature scheme, then use the appropriate component of the compound key for each operation. This gives a JES scheme where the encryption and signature operations are essentially independent. Consequently their respective security properties are retained in the presence of the additional oracle. This simple construction sets a benchmark in terms of key size and other performance measures that any bespoke construction should best in one or more metrics.

Formally, let PKE be an encryption scheme, and let DS be a signature scheme. Then the Cartesian product JES scheme $\mathsf{JES}_{cart}[\mathsf{PKE},\mathsf{DS}]$ is constructed as follows:

- $\underline{\mathsf{JES}_{cart}[\mathsf{PKE},\mathsf{DS}].\mathsf{Pg}(1^\lambda)}$: $pp_{\mathsf{PKE}} \leftarrow\$ \mathsf{PKE}.\mathsf{Pg}(1^\lambda)$; $pp_{\mathsf{DS}} \leftarrow\$ \mathsf{DS}.\mathsf{Pg}(1^\lambda)$;
  Return $(pp_{\mathsf{PKE}}, pp_{\mathsf{DS}})$.
- $\underline{\mathsf{JES}_{cart}[\mathsf{PKE},\mathsf{DS}].\mathsf{Kg}(1^\lambda)}$: $(sk_{\mathsf{PKE}}, pk_{\mathsf{PKE}}) \leftarrow\$ \mathsf{PKE}.\mathsf{Kg}(pp_{\mathsf{PKE}})$;
  $(sk_{\mathsf{DS}}, pk_{\mathsf{DS}}) \leftarrow\$ \mathsf{DS}.\mathsf{Kg}(pp_{\mathsf{DS}})$; Return $((sk_{\mathsf{PKE}}, sk_{\mathsf{DS}}), (pk_{\mathsf{PKE}}, pk_{\mathsf{DS}}))$.
- $\underline{\mathsf{JES}_{cart}[\mathsf{PKE},\mathsf{DS}].\mathsf{Enc}(pp, pk, m)}$: Return $\mathsf{PKE}.\mathsf{Enc}(pp_{\mathsf{PKE}}, pk_{\mathsf{PKE}}, m)$.
- $\underline{\mathsf{JES}_{cart}[\mathsf{PKE},\mathsf{DS}].\mathsf{Dec}(pp, sk, c)}$: Return $\mathsf{PKE}.\mathsf{Dec}(pp_{\mathsf{PKE}}, sk_{\mathsf{PKE}}, c)$.
- $\underline{\mathsf{JES}_{cart}[\mathsf{PKE},\mathsf{DS}].\mathsf{Sign}(pp, sk, m)}$: Return $\mathsf{DS}.\mathsf{Sign}(pp_{\mathsf{DS}}, sk_{\mathsf{DS}}, m)$.
- $\underline{\mathsf{JES}_{cart}[\mathsf{PKE},\mathsf{DS}].\mathsf{Verify}(pp, pk, m, \sigma)}$: Return $\mathsf{DS}.\mathsf{Verify}(pp_{\mathsf{DS}}, pk_{\mathsf{DS}}, m, \sigma)$.

We omit the straightforward proof that this scheme is jointly secure if PKE is IND-CCA secure and DS is EUF-CMA secure.

3.4 An Insecure JES Scheme whose Components are Secure

To show that the definitions are not trivially satisfied, we give a pathological example demonstrating that an encryption scheme and a signature scheme that are individually secure may not be secure when used in combination. Let PKE be an IND-CCA-secure encryption scheme, and let DS be an EUF-CMA-secure signature scheme. Then a JES scheme $\mathsf{JES}_{bad}[\mathsf{PKE},\mathsf{DS}]$ can be constructed as follows:

- $\underline{\mathsf{JES}_{bad}[\mathsf{PKE},\mathsf{DS}].\mathsf{Pg}(1^\lambda)}$: $pp_{\mathsf{PKE}} \leftarrow\$ \mathsf{PKE}.\mathsf{Pg}(1^\lambda)$; $pp_{\mathsf{DS}} \leftarrow\$ \mathsf{DS}.\mathsf{Pg}(1^\lambda)$;

Return $(pp_{\mathsf{PKE}}, pp_{\mathsf{DS}})$.

- $\underline{\mathsf{JES}_{bad}[\mathsf{PKE},\mathsf{DS}].\mathsf{Kg}(1^\lambda)}$: $(sk_{\mathsf{PKE}}, pk_{\mathsf{PKE}}) \leftarrow\$ \mathsf{PKE.Kg}(pp_{\mathsf{PKE}})$; $(sk_{\mathsf{DS}}, pk_{\mathsf{DS}}) \leftarrow\$ \mathsf{DS.Kg}(pp_{\mathsf{DS}})$; Return $((sk_{\mathsf{PKE}}, sk_{\mathsf{DS}}), (pk_{\mathsf{PKE}}, pk_{\mathsf{DS}}))$.
- $\underline{\mathsf{JES}_{bad}[\mathsf{PKE},\mathsf{DS}].\mathsf{Enc}(pp, pk, m)}$: Return $\mathsf{PKE.Enc}(pp_{\mathsf{PKE}}, pk_{\mathsf{PKE}}, m)$.
- $\underline{\mathsf{JES}_{bad}[\mathsf{PKE},\mathsf{DS}].\mathsf{Dec}(pp, sk, c)}$: $m \leftarrow \mathsf{PKE.Dec}(pp_{\mathsf{PKE}}, sk_{\mathsf{PKE}}, c)$; If $(m = \perp)$ then Return sk_{DS} Else Return m.
- $\underline{\mathsf{JES}_{bad}[\mathsf{PKE},\mathsf{DS}].\mathsf{Sign}(pp, sk, m)}$: $\sigma_{\mathsf{DS}} \leftarrow\$ \mathsf{DS.Sign}(pp_{\mathsf{DS}}, sk_{\mathsf{DS}}, m)$; Return $\sigma_{\mathsf{DS}}||sk_{\mathsf{PKE}}$.
- $\underline{\mathsf{JES}_{bad}[\mathsf{PKE},\mathsf{DS}].\mathsf{Verify}(pp, pk, m, \sigma)}$: $\sigma_{\mathsf{DS}}||sk_{\mathsf{PKE}} \leftarrow \sigma$; Return $\mathsf{DS.Verify}(pp_{\mathsf{DS}}, pk_{\mathsf{DS}}, m, \sigma_{\mathsf{DS}})$.

From the security of the base schemes it is easy to see that the encryption scheme given by the algorithms $\mathsf{Pg}, \mathsf{Kg}, \mathsf{Enc}, \mathsf{Dec}$ of $\mathsf{JES}_{bad}[\mathsf{PKE},\mathsf{DS}]$ is IND-CCA secure, and the signature scheme with algorithms $\mathsf{Pg}, \mathsf{Kg}, \mathsf{Sign}, \mathsf{Verify}$ of $\mathsf{JES}_{bad}[\mathsf{PKE},\mathsf{DS}]$ is EUF-CMA secure. However when key generation is shared a single signature reveals the encryption scheme's private key, and the decryption of a badly formed ciphertext reveals the private key of the signature scheme. Thus $\mathsf{JES}_{bad}[\mathsf{PKE},\mathsf{DS}]$ is totally insecure, even though its component schemes are secure.

3.5 A Generic Construction from IBE

We show how to build a JES scheme from an IBE scheme IBE, making use of a one-time strongly unforgeable signature scheme DS. The construction is particularly simple: the encryption scheme component is constructed through a tag-based version of the CHK transform [32], and the signature scheme component through the Naor transform [34]. Since in the Naor construction signatures are just private keys from the IBE scheme, and these private keys could be used to decrypt ciphertexts in the encryption scheme resulting from the CHK transform, we use a bit prefix in the identity space to provide domain separation between the private keys and signatures.

We assume IBE has identity space $\{0,1\}^{n+1}$, and that DS has public key space $\{0,1\}^n$. The encryption component of $\mathsf{JES}[\mathsf{IBE},\mathsf{DS}]$ has message space $\mathsf{IBE.EMSp}(pp)$. The signature scheme component of $\mathsf{JES}[\mathsf{IBE},\mathsf{DS}]$ has message space $\mathsf{IBE.SMSp}(pp) = \{0,1\}^n$ but can be extended to messages of arbitrary length through the use of a collision-resistant hash function $\mathsf{H} : \{0,1\}^* \to \{0,1\}^n$. The algorithms of the joint encryption and signature scheme $\mathsf{JES}[\mathsf{IBE},\mathsf{DS}]$ are given in Figure 3.2.

```
$\underline{\mathsf{JES}[\mathsf{IBE},\mathsf{DS}].\mathsf{Pg}(1^\lambda):}$
$pp_{\mathsf{IBE}} \leftarrow\!\!\$\ \mathsf{IBE.Pg}(1^\lambda)$
$pp_{\mathsf{DS}} \leftarrow\!\!\$\ \mathsf{DS.Pg}(1^\lambda)$
Return $(pp_{\mathsf{IBE}}, pp_{\mathsf{DS}})$

$\underline{\mathsf{JES}[\mathsf{IBE},\mathsf{DS}].\mathsf{Kg}(pp):}$
$(pp_{\mathsf{IBE}}, pp_{\mathsf{DS}}) \leftarrow pp$
$(msk, mpk) \leftarrow\!\!\$\ \mathsf{IBE.MKg}(pp_{\mathsf{IBE}})$
Return $(msk, mpk)$

$\underline{\mathsf{JES}[\mathsf{IBE},\mathsf{DS}].\mathsf{Enc}(pp, pk, m):}$
$(pp_{\mathsf{IBE}}, pp_{\mathsf{DS}}) \leftarrow pp$ ; $mpk \leftarrow pk$
$(sk_{\mathsf{DS}}, pk_{\mathsf{DS}}) \leftarrow\!\!\$\ \mathsf{DS.Kg}(pp_{\mathsf{DS}})$
$u \leftarrow 1 \| pk_{\mathsf{DS}}$
$c_{\mathsf{IBE}} \leftarrow\!\!\$\ \mathsf{IBE.Enc}(pp_{\mathsf{IBE}}, mpk, u, m)$
$\sigma_{\mathsf{DS}} \leftarrow\!\!\$\ \mathsf{DS.Sign}(pp_{\mathsf{DS}}, sk_{\mathsf{DS}}, c_{\mathsf{IBE}})$
Return $(pk_{\mathsf{DS}}, c_{\mathsf{IBE}}, \sigma_{\mathsf{DS}})$

$\underline{\mathsf{JES}[\mathsf{IBE},\mathsf{DS}].\mathsf{Dec}(pp, sk, c):}$
$(pk_{\mathsf{DS}}, c_{\mathsf{IBE}}, \sigma_{\mathsf{DS}}) \leftarrow c$ ; $msk \leftarrow sk$
If $(!\mathsf{DS.Verify}(pp_{\mathsf{DS}}, pk_{\mathsf{DS}}, c_{\mathsf{IBE}}, \sigma_{\mathsf{DS}}))$
    then Return $\perp$
$u \leftarrow 1 \| pk_{\mathsf{DS}}$
$usk \leftarrow\!\!\$\ \mathsf{IBE.UKg}(pp_{\mathsf{IBE}}, msk, u)$
Return $\mathsf{IBE.Dec}(pp_{\mathsf{IBE}}, usk, c_{\mathsf{IBE}})$

$\underline{\mathsf{JES}[\mathsf{IBE},\mathsf{DS}].\mathsf{Sign}(pp, sk, m):}$
$(pp_{\mathsf{IBE}}, pp_{\mathsf{DS}}) \leftarrow pp$ ; $msk \leftarrow sk$
$u \leftarrow 0 \| m$
Return $\mathsf{IBE.UKg}(pp_{\mathsf{IBE}}, msk, u)$

$\underline{\mathsf{JES}[\mathsf{IBE},\mathsf{DS}].\mathsf{Verify}(pp, pk, m, \sigma):}$
$(pp_{\mathsf{IBE}}, pp_{\mathsf{DS}}) \leftarrow pp$ ; $mpk \leftarrow pk$ ; $usk \leftarrow \sigma$
$u \leftarrow 0 \| m$
$x \leftarrow\!\!\$\ \mathsf{IBE.EMSp}(pp_{\mathsf{IBE}})$
$c_{\mathsf{IBE}} \leftarrow\!\!\$\ \mathsf{IBE.Enc}(pp_{\mathsf{IBE}}, mpk, u, x)$
Return $(\mathsf{IBE.Dec}(pp_{\mathsf{IBE}}, usk, c_{\mathsf{IBE}}) = x)$
```

Figure 3.2: JES scheme $\mathsf{JES}[\mathsf{IBE},\mathsf{DS}]$.

Theorem 3.5.1 *Let* IBE *be a selectively-secure IBE scheme and* DS *be a one-time strongly unforgeable signature scheme. Then* $\mathsf{JES}[\mathsf{IBE},\mathsf{DS}]$ *is IND-CCMA secure.*

Proof: Let A be a PT adversary playing game IND-CCMA. We build PT adversaries A_1, A_2 such that

$$\mathbf{Adv}^{\text{ind-ccma}}_{\mathsf{JES}[\mathsf{IBE},\mathsf{DS}],A}(\lambda) \le 2\mathbf{Adv}^{\text{ot-suf-cma}}_{\mathsf{DS},A_1}(\lambda) + \mathbf{Adv}^{\text{ind-sid}}_{\mathsf{IBE},A_2}(\lambda)$$

for all $\lambda \in \mathbb{N}$, from which the theorem follows.

The proof uses the games in Figure 3.3. Game G_0 is as IND-CCMA but returns $\perp$ in response to a decryption query for a ciphertext containing a valid signature under the same pk_{DS} as the challenge ciphertext. We will build A_1, A_2 so that for all $\lambda \in \mathbb{N}$ we have

$$\Pr[\mathrm{G}_0^A(\lambda) \text{ sets } \mathsf{bad}] \le \mathbf{Adv}^{\text{ot-suf-cma}}_{\mathsf{DS},A_1}(\lambda) \tag{3.1}$$

$$2\Pr[\mathrm{G}_0^A(\lambda)] - 1 \le \mathbf{Adv}^{\text{ind-sid}}_{\mathsf{IBE},A_2}(\lambda)\ . \tag{3.2}$$

Games IND-CCMA$_{\mathsf{JES}[\mathsf{IBE},\mathsf{DS}]}$ and G_0 are identical until bad, so by the Fundamental

```
$\underline{\text{MAIN IND-CCMA}^A_{\mathsf{JES[IBE,DS]}}(\lambda)\ /\ \mathrm{G}^A_0(\lambda)}$
$b \leftarrow_\$ \{0,1\}$ ; $c^* \leftarrow \perp$ ; $pk^*_{\mathsf{DS}} \leftarrow \perp$
$pp_{\mathsf{IBE}} \leftarrow_\$ \mathsf{IBE.Pg}(1^\lambda)$ ; $pp_{\mathsf{DS}} \leftarrow_\$ \mathsf{DS.Pg}(1^\lambda)$
$(msk, mpk) \leftarrow_\$ \mathsf{IBE.MKg}(pp_{\mathsf{IBE}})$
$b' \leftarrow_\$ A^{\text{DEC,SIGN,LR}}((pp_{\mathsf{IBE}}, pp_{\mathsf{DS}}), mpk)$
Return $(b = b')$

$\underline{\text{proc DEC}(c)}$ // IND-CCMA$^A_{\mathsf{JES[IBE,DS]}}(\lambda)$ / $\boxed{\mathrm{G}^A_0(\lambda)}$
If $(c = c^*)$ then Return $\perp$
$(pk_{\mathsf{DS}}, c_{\mathsf{IBE}}, \sigma_{\mathsf{DS}}) \leftarrow c$
If $(!\mathsf{DS.Verify}(pp_{\mathsf{DS}}, pk_{\mathsf{DS}}, c_{\mathsf{IBE}}, \sigma_{\mathsf{DS}}))$ then Return $\perp$
If $(pk_{\mathsf{DS}} = pk^*_{\mathsf{DS}})$ then $\mathsf{bad} \leftarrow \mathsf{true}$ ; $\boxed{\text{Return } \perp}$
$u \leftarrow 1||pk_{\mathsf{DS}}$
$usk \leftarrow_\$ \mathsf{IBE.UKg}(pp_{\mathsf{IBE}}, msk, u)$
Return $\mathsf{IBE.Dec}(pp_{\mathsf{IBE}}, usk, c_{\mathsf{IBE}})$

$\underline{\text{proc SIGN}(m)}$ // IND-CCMA$^A_{\mathsf{JES[IBE,DS]}}(\lambda)$ / $\mathrm{G}^A_0(\lambda)$
$u \leftarrow 0||m$
Return $\mathsf{IBE.UKg}(pp_{\mathsf{IBE}}, msk, u)$

$\underline{\text{proc LR}(m_0, m_1)}$ // IND-CCMA$^A_{\mathsf{JES[IBE,DS]}}(\lambda)$ / $\mathrm{G}^A_0(\lambda)$
If $(c^* \neq \perp)$ then Return $\perp$
If $(|m_0| \neq |m_1|)$ then Return $\perp$
$(sk^*_{\mathsf{DS}}, pk^*_{\mathsf{DS}}) \leftarrow_\$ \mathsf{DS.Kg}(pp_{\mathsf{DS}})$
$u \leftarrow 1||pk^*_{\mathsf{DS}}$
$c^*_{\mathsf{IBE}} \leftarrow_\$ \mathsf{IBE.Enc}(pp_{\mathsf{IBE}}, mpk, u, m)$
$\sigma^*_{\mathsf{DS}} \leftarrow_\$ \mathsf{DS.Sign}(pp_{\mathsf{DS}}, sk^*_{\mathsf{DS}}, c^*_{\mathsf{IBE}})$
$c^* \leftarrow (pk^*_{\mathsf{DS}}, c^*_{\mathsf{IBE}}, \sigma^*_{\mathsf{DS}})$
Return $c^*$
```

Figure 3.3: Games used in the proof of Theorem 3.5.1.

Lemma of Game-Playing [23] and the above, for all $\lambda \in \mathbb{N}$ we have:

$$\begin{aligned}
\mathbf{Adv}^{\text{ind-ccma}}_{\mathsf{JES[IBE,DS]},A}(\lambda) &= 2\Pr[\text{IND-CCMA}^{A}_{\mathsf{JES[IBE,DS]}}(\lambda)] - 1 \\
&= 2(\Pr[\text{IND-CCMA}^{A}_{\mathsf{JES[IBE,DS]}}(\lambda)] - \Pr[\mathrm{G}_0^A(\lambda)]) \\
&\qquad + 2\Pr[\mathrm{G}_0^A(\lambda)] - 1 \\
&\le 2\Pr[\mathrm{G}_0^A(\lambda) \text{ sets } \mathsf{bad}] + 2\Pr[\mathrm{G}_0^A(\lambda)] - 1 \\
&\le 2\mathbf{Adv}^{\text{ot-suf-cma}}_{\mathsf{DS},A_1}(\lambda) + \mathbf{Adv}^{\text{ind-sid}}_{\mathsf{IBE},A_2}(\lambda)
\end{aligned}$$

as desired. We proceed to the constructions of A_1, A_2. Adversary A_1 against the strong unforgeability of DS behaves as follows:

```
$A_1^{\text{Sign}}(pp_{\mathsf{DS}}, pk^*_{\mathsf{DS}})$
$(m^*, \sigma^*) \leftarrow \perp$
$b \leftarrow\$ \{0,1\}$ ; $c^* \leftarrow \perp$
$pp_{\mathsf{IBE}} \leftarrow\$ \mathsf{IBE.Pg}(1^\lambda)$
$(msk, mpk) \leftarrow\$ \mathsf{IBE.MKg}(pp_{\mathsf{IBE}})$
$b' \leftarrow\$ A^{\text{Dec},\text{SignSim},\text{LR}}((pp_{\mathsf{IBE}}, pp_{\mathsf{DS}}), mpk)$
Return $(m^*, \sigma^*)$

$\text{Dec}(c)$
If $(c = c^*)$ then Return $\perp$
$(pk_{\mathsf{DS}}, c_{\mathsf{IBE}}, \sigma_{\mathsf{DS}}) \leftarrow c$
If $(!\mathsf{DS.Verify}(pp_{\mathsf{DS}}, pk_{\mathsf{DS}}, c_{\mathsf{IBE}}, \sigma_{\mathsf{DS}}))$
   then Return $\perp$
If $(pk_{\mathsf{DS}} = pk^*_{\mathsf{DS}})$ then
   $(m^*, \sigma^*) \leftarrow (c_{\mathsf{IBE}}, \sigma_{\mathsf{DS}})$ ; Return $\perp$
$u \leftarrow 1||pk_{\mathsf{DS}}$
$usk \leftarrow\$ \mathsf{IBE.UKg}(pp_{\mathsf{IBE}}, msk, u)$
Return $\mathsf{IBE.Dec}(pp_{\mathsf{IBE}}, usk, c_{\mathsf{IBE}})$

$\text{SignSim}(m)$
$u \leftarrow 0||m$
Return $\mathsf{IBE.UKg}(pp_{\mathsf{IBE}}, msk, u)$

$\text{LR}(m_0, m_1)$
If $(c^* \ne \perp)$ then Return $\perp$
If $(|m_0| \ne |m_1|)$ then Return $\perp$
$u \leftarrow 1||pk^*_{\mathsf{DS}}$
$c^*_{\mathsf{IBE}} \leftarrow\$ \mathsf{IBE.Enc}(pp_{\mathsf{IBE}}, mpk, u, m_b)$
$\sigma^*_{\mathsf{DS}} \leftarrow\$ \text{Sign}(c^*_{\mathsf{IBE}})$
$c^* \leftarrow (pk^*_{\mathsf{DS}}, c^*_{\mathsf{IBE}}, \sigma^*_{\mathsf{DS}})$
Return $c^*$
```

Adversary A_1 simulates game G_0 for A, using its challenge public key in creating the challenge ciphertext for A. The Sign oracle called in the LR procedure is A_1's own. Since Sign is only called when A calls LR and $c^* = \perp$, and this situation can occur only on A's first call to LR, A_1 makes at most one call to its Sign oracle. The forgery that A_1 outputs is valid exactly when the flag bad is set in game G_0. The three "If" statements before the forgery (m^*, σ^*) is set ensure this. The second and third "If" statements ensure the signature is valid under challenge public key pk^*_{DS}, while the first and third ensure the pair (m^*, σ^*) differs from A_1's Sign oracle response and query pair $(c^*_{\mathsf{IBE}}, \sigma^*_{\mathsf{DS}})$. This establishes Equation (3.1).

Adversary A_2 against the selective security of IBE behaves as follows:

```
$\underline{A_2^{\text{sID,KD,LR}}(pp_{\mathsf{IBE}})}$
$c^* \leftarrow \perp$
$pp_{\mathsf{DS}} \leftarrow\!\!\$\ \mathsf{DS.Pg}(1^\lambda)$
$(sk^*_{\mathsf{DS}}, pk^*_{\mathsf{DS}}) \leftarrow\!\!\$\ \mathsf{DS.Kg}(pp_{\mathsf{DS}})$
$u^* \leftarrow 1||pk^*_{\mathsf{DS}}$
$mpk \leftarrow \text{sID}(u^*)$
$b' \leftarrow\!\!\$\ A^{\text{Dec,Sign,LRSim}}((pp_{\mathsf{IBE}}, pp_{\mathsf{DS}}), mpk)$
Return $b'$

$\underline{\text{Dec}(c)}$
If $(c = c^*)$ then Return $\perp$
$(pk_{\mathsf{DS}}, c_{\mathsf{IBE}}, \sigma_{\mathsf{DS}}) \leftarrow c$
If $(!\mathsf{DS.Verify}(pp_{\mathsf{DS}}, pk_{\mathsf{DS}}, c_{\mathsf{IBE}}, \sigma_{\mathsf{DS}}))$
   then Return $\perp$
If $(pk_{\mathsf{DS}} = pk^*_{\mathsf{DS}})$ then Return $\perp$
$u \leftarrow 1||pk_{\mathsf{DS}}$
$usk \leftarrow\!\!\$\ \text{KD}(u)$
Return $\mathsf{IBE.Dec}(pp_{\mathsf{IBE}}, usk, c_{\mathsf{IBE}})$

$\underline{\text{Sign}(m)}$
$u \leftarrow 0||m$
Return KD$(u)$

$\underline{\text{LRSim}(m_0, m_1)}$
If $(c^* \neq \perp)$ then Return $\perp$
If $(|m_0| \neq |m_1|)$ then Return $\perp$
$c^*_{\mathsf{IBE}} \leftarrow\!\!\$\ \text{LR}(m_0, m_1)$
$\sigma^*_{\mathsf{DS}} \leftarrow\!\!\$\ \mathsf{DS.Sign}(pp_{\mathsf{DS}}, sk^*_{\mathsf{DS}}, c^*_{\mathsf{IBE}})$
$c^* \leftarrow (pk^*_{\mathsf{DS}}, c^*_{\mathsf{IBE}}, \sigma^*_{\mathsf{DS}})$
Return $c^*$
```

Adversary A_2 simulates game G_0 for A, choosing up-front the public verification key that will determine the identity used in the challenge ciphertext. This identity will never be queried to the KD oracle by Dec as the third "If" statement will cause $\perp$ to be returned before the KD oracle is called on it, and will not be queried by Sign as the challenge identity begins with 1 and all Sign's KD queries are for identities beginning with 0. Adversary A_2 outputs A's guess as its own, winning whenever A does, establishing Equation (3.2) ▌

Theorem 3.5.2 *Let* IBE *be a one-way IBE scheme. Then* $\mathsf{JES}[\mathsf{IBE}, \mathsf{DS}]$ *is EUF-CCMA secure.*

Proof: Let A be a PT adversary playing game EUF-CCMA. We build a PT adversary A_1 such that

$$\mathbf{Adv}^{\text{euf-ccma}}_{\mathsf{JES}[\mathsf{IBE},\mathsf{DS}],A}(\lambda) \leq \mathbf{Adv}^{\text{ow-aid}}_{\mathsf{IBE},A_1}(\lambda) \qquad (3.3)$$

for all $\lambda \in \mathbb{N}$, from which the theorem follows. Adversary A_1 behaves as follows:

```
$\underline{A_1^{\mathrm{KD,C{\scriptsize HAL}}}(pp_{\mathsf{IBE}})}$
$pp_{\mathsf{DS}} \leftarrow_{\$} \mathsf{DS.Pg}(1^\lambda)$
$(m, \sigma) \leftarrow_{\$} A^{\mathrm{S{\scriptsize IGN},D{\scriptsize EC}}}((pp_{\mathsf{IBE}}, pp_{\mathsf{DS}}), mpk)$
$usk \leftarrow \sigma$
$u \leftarrow 0||m$
$c^*_{\mathsf{IBE}} \leftarrow \mathrm{C{\scriptsize HAL}}(u)$
Return $\mathsf{IBE.Dec}(pp_{\mathsf{IBE}}, usk, c^*_{\mathsf{IBE}})$
```

```
$\underline{\mathrm{S{\scriptsize IGN}}(m)}$
$u \leftarrow 0||m$
Return $\mathrm{KD}(u)$

$\underline{\mathrm{D{\scriptsize EC}}(c)}$
$(pk_{\mathsf{DS}}, c_{\mathsf{IBE}}, \sigma_{\mathsf{DS}}) \leftarrow c$
If $(!\mathsf{DS.Verify}(pp_{\mathsf{DS}}, pk_{\mathsf{DS}}, c_{\mathsf{IBE}}, \sigma_{\mathsf{DS}}))$
    then Return $\perp$
$u \leftarrow 1||pk_{\mathsf{DS}}$
$usk \leftarrow_{\$} \mathrm{KD}(u)$
Return $\mathsf{IBE.Dec}(pp_{\mathsf{IBE}}, usk, c_{\mathsf{IBE}})$
```

Adversary A_1 simulates game EUF-CCMA for A, using its KD oracle to answer DEC and SIGN queries. When A halts outputting a message m and signature σ, A_1 submits $0||m$ as its challenge identity. If A did not submit m to its SIGN oracle then this identity has not been queried to the KD oracle so the challenge oracle will respond with a challenge ciphertext. If A output a valid signature on m then the challenge ciphertext will decrypt successfully using the signature as a user-level key, giving $\Pr[\text{EUF-CCMA}^A_{\mathsf{JES[IBE,DS]}}(\lambda)] \leq \Pr[\text{OW-aID}^{A_1}_{\mathsf{IBE}}(\lambda)]$, establishing Equation (3.3). ∎

IBE schemes meeting the standard model security requirements include those of Gentry [65] and Waters [98]. The latter results in a large public key ($n + 3$ group elements), though this could be reduced in practice by generating most of the elements from a seed in a pseudo-random manner. We focus on the instantiation of our construction using Gentry's scheme. This scheme was originally presented in the setting of symmetric pairings. When we translate it to the asymmetric setting and apply our construction at the 128-bit security level using BN curves with sextic twists, we obtain a JES scheme in which the public key consists of two elements of $\mathbb{G}_1$ and two elements of $\mathbb{G}_2$, giving a public key size of 1536 bits. Ciphertexts encrypt elements of $\mathbb{G}_T$ and consist of an element of $\mathbb{G}_1$, two elements of $\mathbb{G}_T$, and a verification key and signature from DS, so are 2304 bits plus the bit length of a verification key and signature in DS. Signatures consist of an element of $\mathbb{Z}_p$ and an element of $\mathbb{G}_2$, so are 768 bits in size. Here we assume that descriptions of groups and pairings are domain parameters that are omitted from our key size calculations. The security of this scheme depends on the hardness of a problem closely related to the Decisional Augmented Bilinear Diffie-Hellman Exponent problem. The full details are in [91].

3.6 A More Efficient Construction

The JES scheme JES1 in Figure 3.4 is based on the signature scheme by Boneh and Boyen [30] and a KEM obtained by applying the techniques of Boyen, Mei and Waters [39] to the second IBE scheme of Boneh and Boyen in [29]. The schemes make use of a bilinear pairing, and the KEM furthermore makes use of a second-preimage resistant hash function $\mathsf{H} : \mathbb{G}_1 \to \{0,1\}^{n-1}$ where $2^n < p$. To obtain a full encryption scheme, the KEM is combined with a DEM, and we assume for simplicity that the key space of the DEM is $\mathcal{K} = \mathbb{G}_T$. Where a binary string is treated as a member of $\mathbb{Z}_p$ it is implicitly converted in the natural manner. The signature scheme supports messages in $\{0,1\}^{n-1}$, but can be extended to support messages in $\{0,1\}^*$ by using a collision resistant hash function, while the encryption scheme has the same message space as the DEM. Note that to minimise the public key size and ciphertext overhead in the scheme, the elements of the public key are placed in the group $\mathbb{G}_1$. However, this implies that signatures contain an element of group $\mathbb{G}_2$, having larger bit representations of elements.

Theorem 3.6.1 *Let* H *be a second-preimage resistant hash function, and* DEM *be a OT-IND-CCA-secure DEM. Assume the DBDHI problem is hard for* G. *Then* JES1 *is IND-CCMA secure.*

Proof: Let A be a PT adversary playing game IND-CCMA and q be a polynomially-bounded function such that A makes $q(\lambda)$ SIGN queries for all $\lambda \in \mathbb{N}$. We build PT adversaries A_1, A_2, A_3 and give a negligible function ν such that

$$\mathbf{Adv}^{\text{ind-ccma}}_{\mathsf{JES1},A}(\lambda) \leq 2\mathbf{Adv}^{\text{sec}}_{\mathsf{H},A_1}(\lambda) + 2\nu(\lambda) + 2\mathbf{Adv}^{\text{dbdhi}}_{\mathsf{G},q(\lambda),A_2}(\lambda) + \mathbf{Adv}^{\text{ot-ind-cca}}_{\mathsf{DEM},A_3}(\lambda)$$

for all $\lambda \in \mathbb{N}$, from which the theorem follows.

The proof uses the games in Figure 3.5 and Figure 3.6. Game G_0 is as IND-CCMA but constructs the message-independent parts of the challenge ciphertext up-front, a change invisible from the adversary's point of view. Game G_1 is as G_0 but returns $\perp$ in response to decryption queries where the first component is not equal to that of the challenge ciphertext, but its hash is. Game G_2 is as G_1 but generates the value x as a function of y, s and a randomly chosen value u, and chooses the randomness in responses to SIGN queries from a more restricted set to accommodate this change. Game G_3 is as G_2 but generates the third component of the challenge ciphertext by encrypting the message under a random DEM key K^*, and responds to decryption

```
$\underline{\mathsf{JES1.Pg}(1^\lambda):}$
$pp_{\mathsf{G}} \leftarrow_{\$} \mathsf{G.Pg}(1^\lambda)$
$pp_{\mathsf{DEM}} \leftarrow_{\$} \mathsf{DEM.Pg}(1^\lambda)$
$pp_{\mathsf{H}} \leftarrow_{\$} \mathsf{H.Pg}(1^\lambda)$
Return $(pp_{\mathsf{G}}, pp_{\mathsf{DEM}}, pp_{\mathsf{H}})$

$\underline{\mathsf{JES1.Kg}(pp):}$
$((p, \mathbb{G}_1, \mathbb{G}_2, e), pp_{\mathsf{DEM}}, pp_{\mathsf{H}}) \leftarrow pp$
$g_1 \leftarrow_{\$} \mathbb{G}_1$ ; $g_2 \leftarrow_{\$} \mathbb{G}_2$
$x, y \leftarrow_{\$} \mathbb{Z}_p^*$
$X \leftarrow g_1^x, Y \leftarrow g_1^y$
$sk \leftarrow (g_1, g_2, x, y)$
$pk \leftarrow (g_1, g_2, X, Y)$
Return $(sk, pk)$

$\underline{\mathsf{JES1.Enc}(pp, pk, m):}$
$((p, \mathbb{G}_1, \mathbb{G}_2, e), pp_{\mathsf{DEM}}, pp_{\mathsf{H}}) \leftarrow pp$
$(g_1, g_2, X, Y) \leftarrow pk$
$s \leftarrow_{\$} \mathbb{Z}_p^*$
$c_1 \leftarrow Y^s$
$h \leftarrow \mathsf{H.Eval}(pp_{\mathsf{H}}, c_1)$
$h' \leftarrow 1||h$
$c_2 \leftarrow X^s \cdot g_1^{s \cdot h'}$
$K \leftarrow e(g_1, g_2)^s$
$c_3 \leftarrow \mathsf{DEM.Enc}(pp_{\mathsf{DEM}}, K, m)$
Return $(c_1, c_2, c_3)$

$\underline{\mathsf{JES1.Dec}(pp, sk, c):}$
$((p, \mathbb{G}_1, \mathbb{G}_2, e), pp_{\mathsf{DEM}}, pp_{\mathsf{H}}) \leftarrow pp$
$(g_1, g_2, x, y) \leftarrow sk$
$h \leftarrow \mathsf{H.Eval}(pp_{\mathsf{H}}, c_1)$
$h' \leftarrow 1||h$
If $(c_1^{(x+h')/y} \neq c_2)$ then Return $\perp$
$K \leftarrow e(c_1, g_2^{1/y})$
Return $\mathsf{DEM.Dec}(pp_{\mathsf{DEM}}, K, c_3)$

$\underline{\mathsf{JES1.Sign}(pp, sk, m):}$
$((p, \mathbb{G}_1, \mathbb{G}_2, e), pp_{\mathsf{DEM}}, pp_{\mathsf{H}}) \leftarrow pp$
$(g_1, g_2, x, y) \leftarrow sk$
$m' \leftarrow 0||m$
$r \leftarrow_{\$} \mathbb{Z}_p \setminus \{-\frac{x+m'}{y}\}$
$\sigma' \leftarrow g_2^{\frac{1}{x+ry+m'}}$
Return $(\sigma', r)$

$\underline{\mathsf{JES1.Verify}(pp, pk, m, \sigma):}$
$((p, \mathbb{G}_1, \mathbb{G}_2, e), pp_{\mathsf{DEM}}, pp_{\mathsf{H}}) \leftarrow pp$
$(g_1, g_2, X, Y) \leftarrow pk$ ; $(\sigma', r) \leftarrow \sigma$
$m' \leftarrow_{\$} 0||m$
Return $(e(X \cdot g_1^{m'} \cdot Y^r, \sigma') = e(g_1, g_2))$
```

Figure 3.4: JES scheme $\mathsf{JES1}$.

```
MAIN $\text{IND-CCMA}^A_{\mathsf{JES1}}(\lambda)$ / $\boxed{G^A_0(\lambda)\ /\ G^A_1(\lambda)}$
$b \leftarrow_\$ \{0,1\}$ ; $c^* \leftarrow \perp$ ; $pp_{\mathsf{G}} \leftarrow_\$ \mathsf{G.Pg}(1^\lambda)$ ; $pp_{\mathsf{DEM}} \leftarrow_\$ \mathsf{DEM.Pg}(1^\lambda)$ ; $pp_{\mathsf{H}} \leftarrow_\$ \mathsf{H.Pg}(1^\lambda)$
$g_1 \leftarrow_\$ \mathbb{G}_1$ ; $g_2 \leftarrow_\$ \mathbb{G}_2$ ; $x, y \leftarrow_\$ \mathbb{Z}^*_p$ ; $X \leftarrow g_1^x, Y \leftarrow g_1^y$ ; $pk \leftarrow (g_1, g_2, X, Y)$
$\boxed{s \leftarrow_\$ \mathbb{Z}^*_p ;\ c^*_1 \leftarrow Y^s ;\ h^* \leftarrow \mathsf{H.Eval}(pp_{\mathsf{H}}, c^*_1) ;\ h' \leftarrow 1||h^* ;\ c^*_2 \leftarrow X^s \cdot g_1^{s \cdot h'} ;\ K^* \leftarrow e(g_1, g_2)^s}$
$b' \leftarrow_\$ A^{\text{DEC},\text{SIGN},\text{LR}}((pp_{\mathsf{G}}, pp_{\mathsf{DEM}}, pp_{\mathsf{H}}), pk)$
Return $(b = b')$

proc DEC($c$)   // $\text{IND-CCMA}^A_{\mathsf{JES1}}(\lambda)$ / $G^A_0(\lambda)$ / $\boxed{G^A_1(\lambda)\ /\ G^A_2(\lambda)\ /\ G^A_3(\lambda)}$
If $(c = c^*)$ then Return $\perp$
$h \leftarrow \mathsf{H.Eval}(pp_{\mathsf{H}}, c_1)$
If $((h = h^*) \wedge (c_1 \neq c^*_1))$ then $\mathsf{bad} \leftarrow \mathsf{true}$ ; $\boxed{\text{Return } \perp}$
$h' \leftarrow 1||h$
If $(c_1^{(x+h')/y} \neq c_2)$ then Return $\perp$
$K \leftarrow e(c_1, g_2^{1/y})$
Return $\mathsf{DEM.Dec}(pp_{\mathsf{DEM}}, K, c_3)$

proc SIGN($m$)   // $\text{IND-CCMA}^A_{\mathsf{JES1}}(\lambda)$ / $G^A_0(\lambda)$ / $G^A_1(\lambda)$ / $\boxed{G^A_2(\lambda)\ /\ G^A_3(\lambda)}$
$m' \leftarrow 0||m$ ; $r \leftarrow_\$ \mathbb{Z}_p \setminus \{-\frac{x+m'}{y}\}$ ; $\boxed{r \leftarrow_\$ \mathbb{Z}_p \setminus \{-\frac{x+m'}{y}, u\} ;}$ $\sigma' \leftarrow g_2^{\frac{1}{x+ry+m'}}$
Return $(\sigma', r)$

proc LR($m_0, m_1$)   // $\text{IND-CCMA}^A_{\mathsf{JES1}}(\lambda)$
If $(c^* \neq \perp)$ then Return $\perp$
If $(|m_0| \neq |m_1|)$ then Return $\perp$
$s \leftarrow_\$ \mathbb{Z}^*_p$ ; $c^*_1 \leftarrow Y^s$ ; $h^* \leftarrow \mathsf{H.Eval}(pp_{\mathsf{H}}, c^*_1)$ ; $h' \leftarrow 1||h^*$ ; $c^*_2 \leftarrow X^s \cdot g_1^{s \cdot h'}$
$K^* \leftarrow e(g_1, g_2)^s$ ; $c^*_3 \leftarrow \mathsf{DEM.Enc}(pp_{\mathsf{DEM}}, K^*, m_b)$ ; $c^* \leftarrow (c^*_1, c^*_2, c^*_3)$
Return $c^*$

proc LR($m_0, m_1$)   // $G^A_0(\lambda)$ / $G^A_1(\lambda)$ / $G^A_2(\lambda)$ / $G^A_3(\lambda)$
If $(c^* \neq \perp)$ then Return $\perp$
If $(|m_0| \neq |m_1|)$ then Return $\perp$
$c^*_3 \leftarrow \mathsf{DEM.Enc}(pp_{\mathsf{DEM}}, K^*, m_b)$ ; $c^* \leftarrow (c^*_1, c^*_2, c^*_3)$
Return $c^*$
```

Figure 3.5: Games used in proof of Theorem 3.6.1.

```
MAIN $\mathrm{G}_2^A(\lambda)$ / $\boxed{\mathrm{G}_3^A(\lambda)}$
$b \leftarrow\!\!\$\ \{0,1\}$ ; $c^* \leftarrow \bot$ ; $pp_{\mathsf{G}} \leftarrow\!\!\$\ \mathsf{G.Pg}(1^\lambda)$ ; $pp_{\mathsf{DEM}} \leftarrow\!\!\$\ \mathsf{DEM.Pg}(1^\lambda)$ ; $pp_{\mathsf{H}} \leftarrow\!\!\$\ \mathsf{H.Pg}(1^\lambda)$
$g_1 \leftarrow\!\!\$\ \mathbb{G}_1$ ; $g_2 \leftarrow\!\!\$\ \mathbb{G}_2$ ; $y, u \leftarrow\!\!\$\ \mathbb{Z}_p^*$ ; $Y \leftarrow g_1^y$
$s \leftarrow\!\!\$\ \mathbb{Z}_p^*$ ; $c_1^* \leftarrow Y^s$ ; $h^* \leftarrow \mathsf{H.Eval}(pp_{\mathsf{H}}, c_1^*)$ ; $h' \leftarrow 1||h^*$
$v \leftarrow h'/u$ ; $x \leftarrow -u(y+v)$ ; $X \leftarrow g_1^x$ ; $pk \leftarrow (g_1, g_2, X, Y)$ ; $c_2^* \leftarrow X^s \cdot g_1^{s \cdot h'}$
$K^* \leftarrow e(g_1, g_2)^s$ ; $\boxed{K^* \leftarrow\!\!\$\ \mathbb{G}_T}$
$b' \leftarrow\!\!\$\ A^{\mathrm{DEC,SIGN,LR}}((pp_{\mathsf{G}}, pp_{\mathsf{DEM}}, pp_{\mathsf{H}}), pk)$
Return $(b = b')$

proc DEC($c$)   // $\mathrm{G}_3^A(\lambda)$
If $(c = c^*)$ then Return $\bot$
$h \leftarrow \mathsf{H.Eval}(pp_{\mathsf{H}}, c_1)$
If $((h = h^*) \wedge (c_1 \neq c_1^*))$ then $\mathsf{bad} \leftarrow \mathsf{true}$ ; Return $\bot$
$h' \leftarrow 1||h$
If $(c_1^{(x+h')/y} \neq c_2)$ then Return $\bot$
$K \leftarrow e(c_1, g_2^{1/y})$
If $((c_1, c_2) = (c_1^*, c_2^*))$ then $K \leftarrow K^*$
Return $\mathsf{DEM.Dec}(pp_{\mathsf{DEM}}, K, c_3)$
```

Figure 3.6: Games used in proof of Theorem 3.6.1, continued.

queries where the first two components of the ciphertext are equal to those of the challenge ciphertext with the decryption of the third component under K^*. We will build A_1, A_2, A_3 and give a negligible function $\nu(\cdot)$ so that for all $\lambda \in \mathbb{N}$ we have

$$\Pr[\mathrm{G}_1^A(\lambda) \text{ sets } \mathsf{bad}] \leq \mathbf{Adv}^{\mathrm{sec}}_{\mathsf{H},A_1}(\lambda) \tag{3.4}$$

$$\Pr[\mathrm{G}_1^A(\lambda)] - \Pr[\mathrm{G}_2^A(\lambda)] \leq \nu(\lambda) \tag{3.5}$$

$$\Pr[\mathrm{G}_2^A(\lambda)] - \Pr[\mathrm{G}_3^A(\lambda)] \leq \mathbf{Adv}^{\mathrm{dbdhi}}_{\mathsf{G},q(\lambda),A_2}(\lambda) \tag{3.6}$$

$$2\Pr[\mathrm{G}_3^A(\lambda)] - 1 \leq \mathbf{Adv}^{\text{ot-ind-cca}}_{\mathsf{DEM},A_3}(\lambda)\ . \tag{3.7}$$

Games G_0 and G_1 are identical until bad, so by the Fundamental Lemma of Game-Playing [23] and the above, for all $\lambda \in \mathbb{N}$ we have:

$$\begin{aligned}
\mathbf{Adv}^{\text{ind-ccma}}_{\mathsf{JES1},A}(\lambda) &= 2\Pr[\text{IND-CCMA}^A_{\mathsf{JES1}}(\lambda)] - 1 \\
&= 2(\Pr[\mathrm{G}_0^A(\lambda)] + (\Pr[\mathrm{G}_1^A(\lambda)] - \Pr[\mathrm{G}_1^A(\lambda)]) \\
&\qquad + (\Pr[\mathrm{G}_2^A(\lambda)] - \Pr[\mathrm{G}_2^A(\lambda)]) + (\Pr[\mathrm{G}_3^A(\lambda)] - \Pr[\mathrm{G}_3^A(\lambda)])) - 1 \\
&= 2\Pr[\mathrm{G}_1^A(\lambda) \text{ sets } \mathsf{bad}] + 2(\Pr[\mathrm{G}_1^A(\lambda)] - \Pr[\mathrm{G}_2^A(\lambda)]) \\
&\qquad + 2(\Pr[\mathrm{G}_2^A(\lambda)] - \Pr[\mathrm{G}_3^A(\lambda)]) + 2\Pr[\mathrm{G}_3^A(\lambda)] - 1 \\
&\leq 2\mathbf{Adv}^{\mathrm{sec}}_{\mathsf{H},A_1}(\lambda) + 2\nu(\lambda) + 2\mathbf{Adv}^{\mathrm{dbdhi}}_{\mathsf{G},q(\lambda),A_2}(\lambda) + \mathbf{Adv}^{\text{ot-ind-cca}}_{\mathsf{DEM},A_3}(\lambda)
\end{aligned}$$

as desired. We proceed to the constructions of A_1, A_2, A_3 and ν. Adversary A_1 against the second-preimage resistance of H behaves as follows:

```
$\underline{A_1(pp_{\mathsf{H}}, z)}$
$z' \leftarrow \perp$ ; $b \leftarrow_{\$} \{0,1\}$ ; $c^* \leftarrow \perp$
$pp_{\mathsf{G}} \leftarrow_{\$} \mathsf{G.Pg}(1^\lambda)$
$pp_{\mathsf{DEM}} \leftarrow_{\$} \mathsf{DEM.Pg}(1^\lambda)$
$((g_1, g_2, x, y), pk) \leftarrow_{\$} \mathsf{JES1.Kg}(pp)$
$c_1^* \leftarrow z$ ; $h^* \leftarrow \mathsf{H.Eval}(pp_{\mathsf{H}}, c_1^*)$
$h' \leftarrow 1||h^*$ ; $c_2^* \leftarrow z^{(x+h')/y}$
$K^* \leftarrow e(z^{1/y}, g_2)$
$b' \leftarrow_{\$} A^{\text{Dec,Sign,LR}}((pp_{\mathsf{G}}, pp_{\mathsf{DEM}}, pp_{\mathsf{H}}), pk)$
Return $z'$

$\underline{\text{Dec}(c)}$
If $(c = c^*)$ then Return $\perp$
$h \leftarrow \mathsf{H.Eval}(pp_{\mathsf{H}}, c_1)$
If $((h = h^*) \wedge (c_1 \neq c_1^*))$ then
    $z' \leftarrow c_1$ ; Return $\perp$
$h' \leftarrow 1||h$
If $(c_1^{(x+h')/y} \neq c_2)$ then Return $\perp$
$K \leftarrow e(c_1, g_2^{1/y})$
Return $\mathsf{DEM.Dec}(pp_{\mathsf{DEM}}, K, c_3)$

$\underline{\text{Sign}(m)}$
$m' \leftarrow 0||m$ ;
$r \leftarrow_{\$} \mathbb{Z}_p \setminus \{-\frac{x+m'}{y}\}$
$\sigma' \leftarrow g_2^{\frac{1}{x+ry+m'}}$
Return $(\sigma', r)$

$\underline{\text{LR}(m_0, m_1)}$
If $(c^* \neq \perp)$ then Return $\perp$
If $(|m_0| \neq |m_1|)$ then Return $\perp$
$c_3^* \leftarrow \mathsf{DEM.Enc}(pp_{\mathsf{DEM}}, K^*, m_b)$
$c^* \leftarrow (c_1^*, c_2^*, c_3^*)$
Return $c^*$
```

Adversary A_1 embeds the sec challenge z as the first component of the challenge ciphertext and constructs c_2^* and K^* accordingly. Since z is a random element of $\mathbb{G}_1$, these values are distributed as in game G_1. It is clear that adversary A_1 succeeds whenever A sets bad. This establishes Equation (3.4).

The public key components X, Y computed in G_2 are distributed as random elements in $\mathbb{G}_1$. Hence, the only difference between G_1 and G_2 from the adversary's point of view is that the randomness used in the construction of signatures is picked from $\mathbb{Z}_p \setminus \{-\frac{x+m'}{y}, u\}$ instead of $\mathbb{Z}_p \setminus \{-\frac{x+m'}{y}\}$, where u is a random element of $\mathbb{Z}_p^*$. This implies that a signature created in G_2 is distributed with a statistical difference of $1/p$ from the distribution of signatures in G_1. Since A asks for q signatures, the signatures created in G_2 will be jointly distributed with a statistical difference of at most q/p from the signatures constructed in G_1. This is a negligible quantity in λ, so establishing Equation (3.5).

Adversary A_2 against the DBDHI problem for G behaves as follows:

```
$\underline{A_2(p, \mathbb{G}_1, \mathbb{G}_2, e, g_1, g_1^\alpha, g_2, g_2^\alpha, g_2^{\alpha^2}, \ldots, g_2^{\alpha^q}, T)}$
$b \leftarrow_\$ \{0,1\}$ ; $c^* \leftarrow \perp$ ; $j \leftarrow 0$
$pp_\mathsf{G} \leftarrow (p, \mathbb{G}_1, \mathbb{G}_2, e)$
$pp_\mathsf{DEM} \leftarrow_\$ \mathsf{DEM.Pg}(1^\lambda)$ ; $pp_\mathsf{H} \leftarrow_\$ \mathsf{H.Pg}(1^\lambda)$
$w_1, \ldots, w_q, w \leftarrow_\$ \mathbb{Z}_p^*$
$f \leftarrow w \prod_{i=1}^q (z + w_i)$
$d_0, \ldots, d_q \leftarrow \mathsf{Coeffs}(f, z)$
$g_2' \leftarrow \prod_{i=0}^q (g_2^{\alpha^i})^{d_i}$
If $(g_2' = 1)$ then
    For $i = 1$ to $q$
        If $(g_2^{w_i} g_2^\alpha = 1_{\mathbb{G}_2})$ then
            If $(e(g_1, g_2)^{-1/w_i} = T)$
                then Return 1
            Else Return 0
$u, l \leftarrow_\$ \mathbb{Z}_p^*$ ; $c_1^* \leftarrow g_1^l$
$h^* \leftarrow \mathsf{H.Eval}(pp_\mathsf{H}, c_1^*)$ ; $h'^* \leftarrow 1||h^*$
$c_2^* \leftarrow g_1^{-ul}$
$v \leftarrow h'^*/u$
$K^* \leftarrow T^{ld_0} \prod_{i=1}^q e(g_1, g_2^{\alpha^{i-1}})^{ld_i}$
$X \leftarrow (g_1^\alpha)^{-u} g_1^{-uv}$ ; $Y \leftarrow g_1^\alpha$
$X' \leftarrow (g_2^\alpha)^{-u} g_2^{-uv}$ ; $Y' \leftarrow g_2^\alpha$
$pk \leftarrow (g_1, g_2', X, Y)$
$b' \leftarrow_\$ A^{\textsc{Dec},\textsc{Sign},\mathrm{LR}}((pp_\mathsf{G}, pp_\mathsf{DEM}, pp_\mathsf{H}), pk)$
If $(b' = b)$ then Return 1
Return 0
```

```
$\underline{\textsc{Dec}(c)}$
If $(c = c^*)$ then Return $\perp$
$h \leftarrow \mathsf{H.Eval}(pp_\mathsf{H}, c_1)$ ; $h' \leftarrow 1||h$
If $((h = h^*) \wedge (c_1 \neq c_1^*))$
    then Return $\perp$
If $(e(c_1, X' g_2^{h'}) \neq e(c_2, Y'))$
    then Return $\perp$
If $((c_1, c_2) = (c_1^*, c_2^*))$
    then $K \leftarrow K^*$
Else $K \leftarrow e((c_2 c_1^u)^{1/(h' - h'^*)}, g_2')$
Return $\mathsf{DEM.Dec}(pp_\mathsf{DEM}, K, c_3)$

$\underline{\textsc{Sign}(m)}$
$j \leftarrow j + 1$
$m' \leftarrow 0||m$ ; $r \leftarrow u + \frac{m' - uv}{w_j}$
$f_j(z) \leftarrow f(z)/(z + w_j)$
$\overline{d}_0, \ldots, \overline{d}_q \leftarrow \mathsf{Coeffs}(f_j, z)$
$\sigma_j \leftarrow \prod_{i=0}^{q-1} (g_2^{\alpha^i})^{\overline{d}_i}$ ; $\sigma' \leftarrow \sigma_j^{1/(r-u)}$
Return $(\sigma', r)$

$\underline{\mathrm{LR}(m_0, m_1)}$
If $(c^* \neq \perp)$ then Return $\perp$
If $(|m_0| \neq |m_1|)$ then Return $\perp$
$c_3^* \leftarrow \mathsf{DEM.Enc}(pp_\mathsf{DEM}, K^*, m_b)$
$c^* \leftarrow (c_1^*, c_2^*, c_3^*)$
Return $c^*$
```

Adversary A_2 constructs a generator g_2' of $\mathbb{G}_2$ for which it knows a set of q values of the form $(w_i, (g_2')^{1/(w_i+\alpha)})$. This is done as follows: A_2 picks random $w_1, \ldots, w_q, w \in \mathbb{Z}_p^*$, defines the polynomial $f(z) = w \prod_{i=1}^q (z + w_i)$. A_2 then expands $f(z) = \sum_{i=0}^q d_i z^i$ to obtain the coefficients $d_0, \ldots, d_q \in \mathbb{Z}_p$, an operation denoted above by $\mathsf{Coeffs}(f, z)$. Then A_2 computes the generator

$$g_2' = \prod_{i=0}^q \left(g_2^{\alpha^i}\right)^{d_i} = g_2^{f(\alpha)} \in \mathbb{G}_2.$$

A_2 checks whether $g_2' = 1_{\mathbb{G}_2}$ (the identity element in $\mathbb{G}_2$), and if this is the case, we must have that $w_i = -\alpha$ for some i, so A_2 can solve the DBDHI problem without interacting with A. Observe that A_2 can compute $(g_2')^{1/(w_i+\alpha)}$ for any of the above chosen w_i by expanding the polynomial $f_i(z) = f(z)/(z + w_i) = \sum_{i=0}^{q-1} \overline{d}_i z^i$ and computing $g_2^{f_i(\alpha)} = \prod_{i=0}^{q-1} (g_2^{\alpha^i})^{\overline{d}_i} = (g_2')^{1/(w_i+\alpha)}$.

A_2 proceeds by picking random $u, l \in \mathbb{Z}_p^*$ and computing challenge ciphertext components $c_1^* = g_1^l$ and $c_2^* = g_1^{-ul}$, as well as the value $v = h'^*/u$. Furthermore, A_2 computes the challenge key $K^* = T^{ld_0} \prod_{i=1}^q e(g_1, g_2^{\alpha^{i-1}})^{ld_i}$.

Lastly, A_2 computes the public key components $X = (g_1^\alpha)^{-u} g_1^{-uv}$ and $Y = g_1^\alpha$. This will implicitly define the private key as $(x, y) = (-u(\alpha + v), \alpha)$. Define $s = l/\alpha$. Observe that

$$\begin{aligned} c_1^* &= g_1^l = Y^{l/\alpha} = Y^s \\ c_2^* &= g_1^{-ul} = g_1^{-u\alpha(l/\alpha)} = g_1^{(x+uv)(l/\alpha)} = X^s g_1^{s \cdot h'^*} \ , \end{aligned}$$

so the ciphertext components are correctly formed for public key (g_1, g_2', X, Y). Furthermore, if $T = e(g_1, g_2)^{1/\alpha}$, then $K^* = e(g_1, g_2)^{lf(\alpha)/\alpha} = e(g_1, g_2')^{l/\alpha} = e(g_1, g_2')^s$, so A_2 simulates G_2. On the other hand, if T is random in $\mathbb{G}_T$, then so is K^*, and A_2 simulates G_3.

In handling decryption queries, since A_2 doesn't know the values x, y to perform the check $c_2 \neq c_1^{(x+h')/y}$, it instead checks the equivalent statement $(e(c_1, X' g_2^{h'}) \neq e(c_2, Y'))$. If the "Else" branch of the final "If" statement is executed, we must have that c_1, c_2 are of the form $c_1 = Y^s$, $c_2 = X^s g_1^{h' \cdot s}$ for some $s \in \mathbb{Z}_p$ and that $h' \neq h'^*$. Note that $c_2 c_1^u = X^s g_1^{s \cdot h'} Y^{su} = g_1^{s(x+h'+uy)} = g_1^{s(-u(\alpha+v)+h'+u\alpha)} = g_1^{s(h'-h'_*)}$. Hence, A_2 is able to compute the DEM key as

$$K = e((c_2 c_1^u)^{1/(h'-h'^*)}, g_2') = e(g_1^s, g_2') = e(g_1, g_2')^s \ .$$

In responding to signature queries, A_2 makes use of its ability to compute q values of the form $(w_j, (g_2')^{1/(w_j+\alpha)})$. It sets $r \leftarrow u + \frac{m'-uv}{w_j}$. Note that $r \neq a$ since, due to a different prefix, $m' \neq h'^* = uv$, and that r is uniform in $\mathbb{Z}_p \setminus \{-\frac{x+m'}{y}, u\}$ since w_j is uniform in $\mathbb{Z}_p \setminus \{0, -\alpha\}$. Furthermore observe that (σ', r) is a valid signature since

$$\sigma_j^{1/(r-u)} = (g_2')^{1/(r-u)(\alpha+w_j)} = (g_2')^{1/(r\alpha+m'-uv-u\alpha)} = (g_2')^{1/(ry+m'+x)} \ .$$

We have then that

$$\begin{aligned} \mathbf{Adv}_{A_2}^{\mathrm{dbdhi}}(\lambda) &= 2\Pr[\mathrm{DBDHI}_{\mathsf{G}}^{A_2}(\lambda)] - 1 \\ &\geq 2(\frac{1}{2}\Pr[\mathrm{G}_2^A(\lambda)] + \frac{1}{2}(1 - \Pr[\mathrm{G}_3^A(\lambda)])) - 1 \\ &\geq \Pr[\mathrm{G}_2^A(\lambda)] - \Pr[\mathrm{G}_3^A(\lambda)] \ , \end{aligned}$$

by a standard transformation, establishing Equation (3.6).

Adversary A_3 against the OT-IND-CCA security of DEM behaves as follows:

```
$A_3^{\text{Dec,LR}}(pp_{\mathsf{DEM}})$
$c^* \leftarrow \perp$
$pp_{\mathsf{G}} \leftarrow_{\$} \mathsf{G.Pg}(1^\lambda)$
$pp_{\mathsf{H}} \leftarrow_{\$} \mathsf{H.Pg}(1^\lambda)$
$y, u \leftarrow_{\$} \mathbb{Z}_p^*$ ; $Y \leftarrow g_1^y$
$s \leftarrow_{\$} \mathbb{Z}_p^*$ ; $c_1^* \leftarrow Y^s$
$h^* \leftarrow \mathsf{H.Eval}(pp_{\mathsf{H}}, c_1^*)$ ; $h' \leftarrow 1||h^*$
$v \leftarrow h'/u$ ; $x \leftarrow -u(y+v)$
$X \leftarrow g_1^x$ ; $pk \leftarrow (g_1, g_2, X, Y)$ ; $c_2^* \leftarrow X^s \cdot g_1^{s \cdot h'}$
$b' \leftarrow_{\$} A^{\text{DecSim,Sign,LRSim}}((pp_{\mathsf{G}}, pp_{\mathsf{DEM}}, pp_{\mathsf{H}}), pk)$
Return $b'$

$\text{DecSim}(c)$
If $(c = c^*)$ then Return $\perp$
$h \leftarrow \mathsf{H.Eval}(pp_{\mathsf{H}}, c_1)$
If $((h = h^*) \wedge (c_1 \neq c_1^*))$ then Return $\perp$
$h' \leftarrow 1||h$
If $(c_1^{(x+h')/y} \neq c_2)$ then Return $\perp$
If $((c_1, c_2) = (c_1^*, c_2^*))$ then Return $\text{Dec}(c_3)$
$K \leftarrow e(c_1, g_2^{1/y})$
Return $\mathsf{DEM.Dec}(pp_{\mathsf{DEM}}, K, c_3)$

$\text{Sign}(m)$
$m' \leftarrow 0||m$ ;
$r \leftarrow_{\$} \mathbb{Z}_p \setminus \{-\frac{x+m'}{y}, u\}$
$\sigma' \leftarrow g_2^{\frac{1}{x+ry+m'}}$
Return $(\sigma', r)$

$\text{LRSim}(m_0, m_1)$
If $(c^* \neq \perp)$ then Return $\perp$
If $(|m_0| \neq |m_1|)$ then Return $\perp$
$c_3^* \leftarrow \text{LR}(m_0, m_1)$
$c^* \leftarrow (c_1^*, c_2^*, c_3^*)$
Return $c^*$
```

Adversary A_3 constructs the public key and challenge ciphertext components c_1^*, c_2^* according to game G_3. It handles signature queries by using its knowledge of the secret key to compute signatures following the randomness restriction. Decryption queries are handled as outlined in game G_3, with an additional test for equality between (c_1, c_2) and (c_1^*, c_2^*), passing the third component to its own decryption oracle in this case. When a LRSim query is made, adversary A_3 forwards the messages to its own LR oracle. It is clear that A_3 wins whenever A wins, establishing Equation (3.7). ▌

Theorem 3.6.2 *Assume SDH is hard for* G. *Then* $\mathsf{JES1}$ *is EUF-CCMA secure.*

The proof closely follows that of [30]. Rather than reduce directly to the SDH problem, we reduce to the weak chosen-message attack security of the related signature scheme BB in Figure 3.7, which in turn reduces to the SDH problem. The following lemma appears in [30].

```
BB.Pg($1^\lambda$) :
    $pp \leftarrow_\$ \mathsf{G.Pg}(1^\lambda)$
    Return $pp$

BB.Kg($pp$) :
    $(p, \mathbb{G}_1, \mathbb{G}_2, e) \leftarrow pp$
    $g_1 \leftarrow_\$ \mathbb{G}_1$ ; $g_2 \leftarrow_\$ \mathbb{G}_2$
    $z \leftarrow_\$ \mathbb{Z}_p^*$
    $Z \leftarrow g_1^z$
    $sk \leftarrow (g_1, g_2, z)$
    $pk \leftarrow (g_1, g_2, Z)$
    Return $(sk, pk)$

BB.Sign($pp, sk, m$) :
    $(p, \mathbb{G}_1, \mathbb{G}_2, e) \leftarrow pp$
    $(g_1, g_2, z) \leftarrow sk$
    If $(z + m \equiv 0 \mod p)$ then Return $1_{\mathbb{G}_2}$
    $\sigma \leftarrow g_2^{\frac{1}{z+m}}$
    Return $\sigma$

BB.Verify($pp, pk, m, \sigma$) :
    $(p, \mathbb{G}_1, \mathbb{G}_2, e) \leftarrow pp$
    $(g_1, g_2, Z) \leftarrow pk$
    If $((\sigma = 1_{\mathbb{G}_2}) \wedge (Z \cdot g_1^m = 1_{\mathbb{G}_1}))$ then
        Return true
    Return $(e(Z \cdot g_1^m, \sigma) = e(g_1, g_2))$
```

Figure 3.7: Signature scheme BB.

Lemma 3.6.3 *Assume SDH is hard for* G. *Then* BB *is unforgeable under weak chosen-message attack.*

The following lemma combines with the preceding lemma to give Theorem 3.6.2.

Lemma 3.6.4 *Let* BB *be unforgeable under weak chosen-message attack. Then* JES1 *is EUF-CCMA secure.*

We give some intuition for the proof. Let A be a PT adversary playing game EUF-CCMA and q be a polynomially-bounded function such that A makes $q(\lambda)$ Sign queries for all $\lambda \in \mathbb{N}$. Adversary A_1 against the weak chosen-message security of BB obtains $q(\lambda)$ message signature pairs (w_i, σ_i) on random messages $w_i \in \mathbb{Z}_p$ and a public key containing $Z = g_1^z$. We can embed the BB challenge element Z as the X component in the public key we give to A, and choose random $y \in \mathbb{Z}_p$ to calculate the Y component. When A makes its ith signature query on a message m, A_1 chooses $r \in \mathbb{Z}_p$ such that $0||m + yr = w_i$, and returns (σ_i, r) as valid signature on m. Decryption queries can be handled without knowing z by using the pairing and any BB message signature pair to check they are formed correctly. A halts and outputs a forgery (m^*, σ^*, r^*), and if this verifies then $(0||m^* + yr^*, \sigma^*)$ verifies in the BB scheme.

The problem here is that $0||m^* + yr^*$ might be in $w_1, \ldots, w_q$, and so outputting $(0||m^* + yr^*, \sigma^*)$ would not win the BB game. However should this happen, the value y can be computed from A's forgery, which would be helpful had we embedded the challenge Z as the Y value instead.

We split the possible adversaries into two types. Let $m_1, \ldots, m_q$ be the messages A adaptively queries, and let (σ_i, r_i) be the signature returned in response to the ith query. Furthermore, let $w_i \leftarrow 0||m_i + yr_i$ for each i, and let (m^*, σ^*, r^*) be the forgery output by A.

A type 1 adversary

- makes a decryption query on a ciphertext $c = (c_1, c_2, c_3)$ such that $1||h = -x$, where $h = \mathsf{H.Eval}(pp_{\mathsf{H}}, c_1)$, or
- makes a signature query on a message m such that $0||m = -x$, or
- outputs a forgery on a message m^* such that $0||m^* + yr^* \notin \{w_1, \ldots, w_q\}$,

while a type 2 adversary

- never makes a decryption query on a ciphertext $c = (c_1, c_2, c_3)$ such that $1||h = -x$, where $h = \mathsf{H.Eval}(pp_{\mathsf{H}}, c_1)$, and
- never makes a signature query on a message m such that $0||m = -x$, and
- outputs a forgery on a message m^* such that $0||m^* + yr^* \in \{w_1, \ldots, w_q\}$.

This partition covers all cases so a successful adversary must be either type 1 or type 2. We build adversaries A_1, A_2 to handle each case.

Proof: Let A be a PT type 1 adversary playing game EUF-CCMA. We build a PT adversary A_1 such that

$$\mathbf{Adv}^{\text{euf-ccma}}_{\mathsf{JES1},A}(\lambda) \leq \mathbf{Adv}^{\text{euf-wma}}_{\mathsf{BB},A_1}(\lambda) \tag{3.8}$$

for all $\lambda \in \mathbb{N}$.

Adversary A_1 behaves as follows:

```
$\underline{A_1(pp)}$
$w_1, \ldots, w_q \leftarrow_\$ \mathbb{Z}_p$
$(pk, \sigma_1, \ldots, \sigma_q) \leftarrow \textsc{Sign}(w_1, \ldots, w_q)$
$(g_1, g_2, Z) \leftarrow pk$
For $i = 1$ to $q$
    If $(\sigma_i = 1_{\mathbb{G}_2})$ then Return $\textsc{Forge}(-w_i)$
$l \leftarrow 0$
$y \leftarrow_\$ \mathbb{Z}_p^*$ ; , $Y \leftarrow g_1^y$
$(m^*, \sigma^*, r^*) \leftarrow_\$ A^{\textsc{SignSim},\textsc{Dec}}(pp, g_1, g_2, Z, Y)$
Return $(0||m^* + yr^*, \sigma^*)$

$\underline{\textsc{Forge}(z)}$
$m \leftarrow_\$ \mathbb{Z}_p \setminus \{-z, w_1, \ldots, w_q\}$
$\sigma \leftarrow g_2^{\frac{1}{z+m}}$
Return $(m, \sigma)$
```

```
$\underline{\textsc{SignSim}(m)}$
$m' \leftarrow_\$ 0||m$
If $(g_1^{-m'} = Z)$
    then Exit $(\textsc{Forge}(-m'))$
$l \leftarrow l + 1$
$r_l \leftarrow (w_l - m')/y$
Return $(\sigma_l, r_l)$

$\underline{\textsc{Dec}(c)}$
$(c_1, c_2, c_3) \leftarrow c$
$h \leftarrow \mathsf{H.Eval}(pp_\mathsf{H}, c_1)$
$h' \leftarrow 1||h$
If $(g_1^{-h'} = Z)$
    then Exit $(\textsc{Forge}(-h'))$
If $(e(c_2 c_1^{(w_1-h')/y}, \sigma_1) \neq e(c_1, g_2^{1/y}))$
    then Return $\perp$
$K \leftarrow e(c_1, g_2^{1/y})$
Return $\mathsf{DEM.Dec}(pp_\mathsf{DEM}, K, c_3)$
```

A_1 starts by requesting signatures on q random messages in $\mathbb{Z}_p$. Upon receiving the corresponding signatures, A_1 checks if any $\sigma_i = 1_{\mathbb{G}_2}$, revealing $-w_i$ as the secret key z which the Forge procedure uses to generate a valid signature on a random message. A similar check is performed during Sign and Dec queries. The call "Exit $(\cdot)$" exits the current procedure and halts execution of A_1, outputting the result of evaluating whatever is in the brackets. It is clear that if the Forge procedure is executed then A_1 wins the game. We now consider the execution of A_1 if the Forge procedure is not executed.

In response to the lth Sign query, A_1 computes r_l as $(w_l - m')/y$ and returns (σ_l, r_l) as the signature on the queried message m. This is a valid signature as

$$e(Z \cdot g_1^{m'} \cdot Y^{r_l}, \sigma_l) = e(Z \cdot g_1^{m'+r_l y}, \sigma_l) = e(Z \cdot g_1^{w_l}, \sigma_l) = e(g_1, g_2),$$

by the validity of σ_l on w_l in the BB scheme. The signatures are properly distributed as w_l is uniform in $\mathbb{Z}_p \setminus \{-z\}$, so r_l is uniform in $\mathbb{Z}_p \setminus \{-\frac{z-m'}{y}\}$.

When responding to decryption queries, A_1 must perform the check $(c_1^{(z+h')/y} \neq c_2)$ without knowing z. A_1 instead checks if the equation

$$e(c_2 c_1^{(w_1-h')/y}, \sigma_1) = e(c_1, g_2^{1/y})$$

holds. If this is the case, then, due to the properties of the pairing and that $\sigma_1 = g_2^{1/(z+w_1)}$, we must have that

$$\left(c_2 c_1^{(w_1-h')/y}\right)^{1/(z+w_1)} = c_1^{1/y}$$

which implies that $c_2 = c_1^{(z+h')/y}$.

Since A is a type 1 adversary, its output is such that $0||m^* + yr^* \notin \{w_1, \dots, w_q\}$, so A_1 wins whenever A does, or whenever the Forge procedure is called, establishing Equation (3.8).

Let A be a PT type 2 adversary playing game EUF-CCMA. We build a PT adversary A_2 such that

$$\mathbf{Adv}^{\text{euf-ccma}}_{\mathsf{JES1},A}(\lambda) \le \mathbf{Adv}^{\text{euf-wma}}_{\mathsf{BB},A_2}(\lambda) + q(\lambda)/p \tag{3.9}$$

for all $\lambda \in \mathbb{N}$.

Adversary A_2 behaves as follows:

```
$\underline{A_2(pp)}$
$w_1, \dots, w_q \leftarrow_\$ \mathbb{Z}_p^*$
$(pk, \sigma_1, \dots, \sigma_q) \leftarrow \text{Sign}(w_1, \dots, w_q)$
$(g_1, g_2, Z) \leftarrow pk$
For $i = 1$ to $q$
    If $(\sigma_i = 1_{\mathbb{G}_2})$ then Return $\text{Forge}(-w_i)$
$l \leftarrow 0$ ; $L \leftarrow \perp$
$x \leftarrow_\$ \mathbb{Z}_p^*$ ; , $X \leftarrow g_1^x$
$(m^*, \sigma^*, r^*) \leftarrow_\$ A^{\text{Sign,Dec}}(pp, g_1, g_2, X, Z)$
$(m, r) \leftarrow T[g_1^{0||m^*} Z^{r^*}]$
$z \leftarrow (0||m^* - 0||m)/(r - r^*)$
$(m', \sigma') \leftarrow \text{Forge}(z)$
Return $(m', \sigma')$

$\underline{\text{Forge}(z)}$
$m \leftarrow_\$ \mathbb{Z}_p \setminus \{-z, w_1, \dots, w_q\}$
$\sigma \leftarrow g_2^{\frac{1}{z+m}}$
Return $(m, \sigma)$
```

```
$\underline{\text{Sign}(m)}$
$m' \leftarrow_\$ 0||m$
$l \leftarrow l + 1$
$r_l \leftarrow (x + m')/w_l$
$T[g_1^{m'} Z^{r_l}] \leftarrow (m, r_l)$
Return $(\sigma_l^{1/r_l}, r_l)$

$\underline{\text{Dec}(c)}$
$(c_1, c_2, c_3) \leftarrow c$
$h \leftarrow \mathsf{H.Eval}(pp_{\mathsf{H}}, c_1)$
$h' \leftarrow 1||h$
If $(e(c_1^{(x+h')} c_2^{w_1}, \sigma_1) \neq e(c_2, g_2))$
    then Return $\perp$
$K \leftarrow e(c_2, g_2^{1/(x+h')})$
Return $\mathsf{DEM.Dec}(pp_{\mathsf{DEM}}, K, c_3)$
```

A_2 starts by requesting signatures on $q(\lambda)$ random messages in $\mathbb{Z}_p^*$. Upon receiving the corresponding signatures, A_2 checks if any $\sigma_i = 1_{\mathbb{G}_2}$, revealing $-w_i$ as the secret key z which the FORGE procedure uses to generate a valid signature on a random message. If this does not occur the execution of A_2 continues.

In response to the lth SIGN query, A_2 computes r_l as $(x + m')/w_l$ and returns $(\sigma_l^{1/r_l}, r_l)$ as the signature on the queried message m. This is a valid signature as

$$e(Xg_1^{m'}Z^{r_l}, \sigma_l^{1/r_l}) = e(g_1^{x+m'}Z^{r_l}, \sigma_l^{1/r_l}) = e(g_1^{(x+m')/r_l}Z, \sigma_l) = e(g_1^{w_l}Z, \sigma_l) = e(g_1, g_2)$$

by the validity of σ_l on w_l in the BB scheme. A_2 stores the value (m, r_l) in a hash table T under the key $g_1^{m'}Z^{r_l}$. A_2 will later use T in computing z from A's forgery. Note that $r_l \neq 0$ since a type 2 adversary will only submit messages with $m' \neq -x$, and that r_l is uniform in $\mathbb{Z}_p^* \setminus \{-\frac{x+m'}{z}\}$ since w_l is uniform in $\mathbb{Z}_p^* \setminus \{-z\}$. However, since r_l would be distributed uniformly in $\mathbb{Z}_p \setminus \{-\frac{x+m'}{z}\}$ in a real interaction, there is a statistical distance of $1/p$ from the correct distribution. Hence, A_2's simulation of all signature queries will at most have a statistical distance of $q(\lambda)/p$ from the correct distribution.

When responding to decryption queries, A_2 must perform the check $(c_1^{(x+h')/z} \neq c_2)$ without knowing z. A_2 instead checks if the equation

$$e(c_1^{(x+h')}c_2^{w_1}, \sigma_1) = e(c_2, g_2)$$

holds. If this is the case, then, due to the properties of the pairing and that $\sigma_1 = g_2^{1/(z+w_1)}$, we must have

$$\left(c_1^{(x+h')}c_2^{w_1}\right)^{1/(z+w_1)} = c_2$$

which implies that $c_2 = c_1^{(x+h')/z}$. From the same equality, A_2 can compute the DEM decryption key $e(c_1, g_2^{1/z})$ as $e(c_2, g_2^{1/(x+h')})$. Note that a type 2 adversary will never submit a ciphertext with $h' = -x$.

Since A is a type 2 adversary we have that $0||m^* + yr^* \in \{w_1, \ldots, w_q\}$. A_2 computes $g_1^{0||m^*}Z^{r^*}$ and obtains the corresponding value (m, r) from the hash table T. We then have that $g_1^{0||m^*}Z^{r^*} = g_1^{0||m}Z^r$. If A is successful then $(m, r) \neq (m^*, r^*)$, otherwise the forgery would be identical to a previously given signature on m. Since $g_1^{0||m^*}Z^{r^*} = g_1^{0||m}Z^r$ it follows that $0||m \neq 0||m^*$ and $r \neq r^*$, therefore A_2 can compute z as $(0||m^* - 0||m)/(r - r^*)$ and forge a BB signature on a message of its choosing. A_2 succeeds when A does, establishing Equation (3.9).

Since we don't know in advance whether A will be a type 1 or type 2 adversary, we run either A_1 or A_2 with equal probability, resulting in an adversary that succeeds in breaking BB with probability at least $(\Pr[\text{EUF-CCMA}^{A}_{\mathsf{JES1}}(\lambda)] - q(\lambda)/p)/2$, and the theorem holds. ▌

The scheme $\mathsf{JES1}$ provides public keys consisting of three group elements of $\mathbb{G}_1$ and one group element of $\mathbb{G}_2$. If the scheme is instantiated using BN curves with sextic twists, this translates into a public key size of 1280 bits for a 128 bit security level. Furthermore, assuming that the DEM is redundancy-free (which can be achieved if the DEM is a strong pseudorandom permutation [92]), the total ciphertext overhead is just two group elements of $\mathbb{G}_1$ which translates into 512 bits. Signatures consist of a single group element of $\mathbb{G}_2$ and an element of $\mathbb{Z}_p$, and will be 768 bits.

3.7 Comparison of Schemes

In this section, we provide a comparison of the schemes arising from our IBE-based construction $\mathsf{JES}[\mathsf{IBE}, \mathsf{DS}]$, our more efficient construction $\mathsf{JES1}$ and the Cartesian product construction $\mathsf{JES}_{cart}[\mathsf{PKE}, \mathsf{DS}]$. In our comparison we will limit ourselves to other discrete-log/pairing-based schemes since provably secure (standard model) lattice-based schemes with short public keys are still unavailable and factoring-based schemes do not scale very well (for 128-bit security, the modulus would need to be > 3000 bits which is not competitive). We will include group generators in public key size calculations as the required number depends on the scheme, but we allow sharing of generators between signature and encryption component in Cartesian product instantiations to improve these constructions. Note that it is possible to reduce the private key of any scheme to a single short random seed by making the following simple modification to the scheme: to generate a public/private keypair, pick a random seed, generate the randomness required by the key generation algorithm by applying a pseudorandom generator to the seed, and generate the public/private keypair using this randomness, but store only the seed as the private key. Whenever the original private key is needed, re-compute this by applying the pseudorandom generator to the seed and re-run the key generation algorithm with the resulting randomness. This observation essentially makes the difference in private key sizes irrelevant, and we will not include this aspect in our comparison. We consider several instantiations of the Cartesian product construction with standard model secure encryption and signature schemes and give the results in Figure 3.8.

We will focus on Cartesian product instantiations using the scheme by Boneh and Boyen [30] as a signature component. This scheme is among the most efficient

Signature Scheme	PKE Scheme	Public Key Size	Signature Size	Ciphertext Overhead
BB[30]	BB[29] + BMW[39]	1792	768	512
BB[30]	KD[82]	2048	768	640
BB[30]	Kiltz[78]	1792	768	512
JES[Gentry[65], DS]		1536	768	$1280 + \lvert pk_{\mathsf{DS}} \rvert + \lvert \sigma_{\mathsf{DS}} \rvert$
JES1		1280	768	512

Figure 3.8: Comparison of JES schemes at the 128-bit security level.

signature schemes and additionally has a short public key. To reduce the public key size even further, we can remove the redundant element $v = e(g_1, g_2)$ and place as many elements as possible in the group $\mathbb{G}_1$ of the pairing. The latter implies that signatures will be elements of $\mathbb{G}_2 \times \mathbb{Z}_p$ which results in an increase in signature size. However, since the Cartesian product constructions should compete with the JES schemes in terms of public key size, this trade-off is desirable. While other signature schemes could be considered, we were not able to find a scheme providing shorter public keys without a significant disadvantage elsewhere. For instance, hash-based signature schemes give extremely short public keys (the hash function description plus the root digest), but result in signatures with length logarithmic in the number of messages to be signed. The signature scheme of Hofheinz and Kiltz [76] has shorter signatures than the Boneh-Boyen scheme and a public key consisting of a few group elements plus a hash key, but here the hash key will be long to achieve provable programmability.

For the encryption component, a relevant option is a DEM combined with the KEM obtained by applying the techniques by Boyen, Mei and Waters [39] to the second IBE scheme of Boneh and Boyen in [29], which also forms the basis of our concrete scheme. Combined with the Boneh-Boyen signature scheme, and assuming the group generators in the two schemes are shared, this yields a very efficient instantiation of the Cartesian product construction in which public keys consist of five group elements of $\mathbb{G}_1$, one group element of $\mathbb{G}_2$ (and a key defining a target collision resistant hash function). This is larger by two elements of $\mathbb{G}_1$ than the public key in our concrete construction JES1, which translates to a difference of 512 bits. Note that signature size, ciphertext overhead and computation costs are the same for the Cartesian product scheme and our construction.

Another encryption scheme to consider is that of Kurosawa and Desmedt [82]. Instantiating the Cartesian product construction with the Kurosawa-Desmedt scheme and the Boneh-Boyen signature scheme yields a scheme with a public key consisting of six elements of $\mathbb{G}_1$, one element of $\mathbb{G}_2$ (and a key defining a target collision resistant hash), assuming that the Kurosawa-Desmedt scheme is implemented in $\mathbb{G}_1$. Hence, the public key will be larger by three group elements of $\mathbb{G}_1$ compared to our concrete construction, which equates to a difference of 768 bits at the 128-bit security level. Signature size and signing and verification costs will be the same as in our construction, while the ciphertext overhead will be slightly larger (an extra 128 bits) due to the requirement that the symmetric encryption scheme used in the Kurosawa-Desmedt scheme is authenticated. However, decryption costs will be lower since no pairing computations are required.

Lastly, the encryption scheme of Kiltz [78] might be considered. Again, combining this with the Boneh-Boyen signature scheme, and assuming group generators are shared, will yield a Cartesian product scheme with public keys consisting of five elements of $\mathbb{G}_1$ and one element of $\mathbb{G}_2$. This is two group elements of $\mathbb{G}_1$ larger than the public key of our concrete construction, which equates to an increase of 512 bits at the 128-bit security level. Signature size and ciphertext overhead will be the same while decryption in the Cartesian product scheme will be more efficient, since no pairing computations are required.

In summary, our concrete construction $\mathsf{JES1}$ of a JES scheme admits shorter public keys than any instantiation of the Cartesian product construction $\mathsf{JES}_{cart}[\mathsf{PKE}, \mathsf{DS}]$ with known standard model secure encryption and signature schemes, and furthermore enjoys compact ciphertexts and signatures.

3.8 Signcryption from Joint Encryption and Signature

A signcryption scheme combines the functionality of signatures and encryption, allowing users to achieve message confidentiality and origin authentication through one operation. However, with the exception of a few random oracle model schemes [85, 84, 88, 83], most signcryption schemes define separate key generation algorithms for senders and receivers, or essentially define a public/private keypair to consist of the concatenation of separate sender and receiver keypairs. Hence, a user playing the role of both sender and receiver will have to generate the equivalent of two keypairs. Extending the ideas of a joint encryption and signature scheme, we show how to construct a signcryption scheme which enables a user to use a single keypair for both sender and receiver roles, and which furthermore allows efficient instantiations with short public keys in the standard model.

Note that any signcryption scheme can be redefined to use a single key generation algorithm. More specifically, a signcryption scheme using separate key generation algorithms for senders and receivers, $\mathsf{SC.SKg}(pp)$ and $\mathsf{SC.RKg}(pp)$, can be redefined to use a single key generation algorithm which simply runs $(sk_s, pk_s) \leftarrow \mathsf{SC.SKg}(pp)$ and $(sk_r, pk_r) \leftarrow \mathsf{SC.RKg}(pp)$, and returns the private key $sk = (sk_s, sk_r)$ and public key $pk = (pk_s, pk_r)$. When using (sk, pk) as either a sender or receiver keypair, only the relevant parts of the keys are used. This trivial construction can be used as a benchmark when judging the public key size and other performance measures of concrete signcryption schemes using a single keypair for both sender and receiver roles.

Construction. We will now show how a joint encryption and signature scheme can be used to construct a signcryption, scheme. Our construction is based on the "sign then tag-based encrypt" (StTE) construction of [89]. In this construction, signcryption involves generating a signature on the message and receiver public key, then using the sender public key as a tag in encrypting the message and signature. This binds the ciphertext to a specific sender/receiver keypair, which is required to achieve security in the multi-user setting. We therefore need a joint tag-based encryption and signature scheme, which we can construct from a joint encryption and signature scheme through the standard technique of appending the tag to the message before encryption, and then on decryption checking for equality between tags. The signcryption scheme $\mathsf{SC}[\mathsf{JES}]$ is shown in Figure 3.9. The next two theorems establish the security of $\mathsf{SC}[\mathsf{JES}]$.

Theorem 3.8.1 *Let* JES *be an IND-CCMA-secure JES scheme. Then* $\mathsf{SC}[\mathsf{JES}]$ *is MU-IND-iCCA secure.*

Proof: Let A be a PT adversary playing game MU-IND-iCCA. We build a PT adversary A_1 such that

$$\mathbf{Adv}^{\text{mu-ind-icca}}_{\mathsf{SC}[\mathsf{JES}],A}(\lambda) \leq \mathbf{Adv}^{\text{ind-ccma}}_{\mathsf{JES},A_1}(\lambda) \tag{3.10}$$

for all $\lambda \in \mathbb{N}$, from which the theorem follows. Adversary A_1 behaves as follows:

```
$\underline{\mathsf{SC[JES].Pg}(1^\lambda):}$
$pp \leftarrow\!\$\ \mathsf{JES.Pg}(1^\lambda)$
Return $pp$

$\underline{\mathsf{SC[JES].Kg}(pp):}$
$(sk, pk) \leftarrow\!\$\ \mathsf{JES.Kg}(pp)$
Return $(sk, pk)$

$\underline{\mathsf{SC[JES].Sc}(pp, sk_s, pk_s, pk_r, m):}$
$t \leftarrow pk_s$
$\sigma \leftarrow\!\$\ \mathsf{JES.Sign}(pp, sk_s, pk_r||m)$
$c \leftarrow\!\$\ \mathsf{JES.Enc}(pp, pk_r, m||t||\sigma)$
Return $c$

$\underline{\mathsf{SC[JES].Usc}(pp, sk_r, pk_r, pk_s, c):}$
$t \leftarrow pk_s$
$m' \leftarrow \mathsf{JES.Dec}(pp, sk_r, c)$
If $(m = \perp)$ then Return $\perp$
$m||t'||\sigma \leftarrow m'$
If $(t' \neq t)$ then Return $\perp$
If $(!\mathsf{JES.Verify}(pp, pk_s, pk_r||m, \sigma))$
    then Return $\perp$
Return $m$
```

Figure 3.9: Signcryption scheme SC[JES].

```
$\underline{A_1^{\text{DEC,SIGN,LR}}(pp)}$
$c^* \leftarrow \perp$ ; $pk_s^* \leftarrow \perp$
$pp \leftarrow\!\$\ \mathsf{JES.Pg}(1^\lambda)$
$(sk, pk) \leftarrow\!\$\ \mathsf{JES.Kg}(pp)$
$b' \leftarrow\!\$\ A^{\text{SC,USC,LRSIM}}(pp, pk)$
Return $b'$

$\underline{\text{SC}(pk_r, m)}$
$t \leftarrow pk$
$\sigma \leftarrow\!\$\ \text{SIGN}(pk_r||m)$
Return $\mathsf{JES.Enc}(pp, pk_r, m||t||\sigma)$

$\underline{\text{USC}(pk_s, c)}$
If $((c = c^*) \wedge (pk_s = pk_s^*))$
    then Return $\perp$
$t \leftarrow pk_s$
$m' \leftarrow \text{DEC}(c)$
If $(m' = \perp)$ then Return $\perp$
$m||t'||\sigma \leftarrow m'$
If $(t \neq t')$ then Return $\perp$
If $(!\mathsf{JES.Verify}(pp, pk_s, pk||m, \sigma))$
    then Return $\perp$
Return $m$

$\underline{\text{LRSIM}(sk_s, pk_s, m_0, m_1)}$
If $(c^* \neq \perp)$ then Return $\perp$
If $(|m_0| \neq |m_1|)$ then Return $\perp$
$pk_s^* \leftarrow pk_s$
$t \leftarrow pk_s$
$\sigma_0 \leftarrow\!\$\ \text{SIGN}(pk||m_0)$
$\sigma_1 \leftarrow\!\$\ \text{SIGN}(pk||m_1)$
$c^* \leftarrow\!\$\ \text{LR}(m_0||t||\sigma_0, m_1||t||\sigma_1)$
Return $c^*$
```

It is possible that the call to A_1's DEC oracle in USC will be for the challenge ciphertext c^*. This occurs if A submits to USC the challenge ciphertext, but with $pk_s \neq pk_s^*$. In this case DEC (and hence USC) will return $\perp$, however this is the appropriate response as the third "If" statement in USC should cause $\perp$ to be returned since the value t encrypted in the challenge ciphertext is pk_s^* which is not equal to pk_s. Adversary A_1 simulates game MU-IND-iCCA for A, establishing Equation (3.10). ▮

Theorem 3.8.2 *Let* JES *be an EUF-CCMA-secure JES scheme. Then* SC[JES] *is MU-EUF-iCMA secure.*

Proof: Let A be a PT adversary playing game MU-EUF-iCMA. We build a PT adversary A_1 such that

$$\mathbf{Adv}^{\text{mu-euf-icma}}_{\mathsf{SC[JES]},A}(\lambda) \leq \mathbf{Adv}^{\text{euf-ccma}}_{\mathsf{JES},A_1}(\lambda) \tag{3.11}$$

for all $\lambda \in \mathbb{N}$, from which the theorem follows. Adversary A_1 behaves as follows:

```
$\underline{A_1^{\text{DEC},\text{SIGN}}(pp)}$
$pp \leftarrow\!\$\ \mathsf{JES.Pg}(1^\lambda)$
$(sk, pk) \leftarrow\!\$\ \mathsf{JES.Kg}(pp)$
$(sk_r, pk_r, c) \leftarrow\!\$\ A^{\text{SC},\text{USC}}(pp, pk)$
$t \leftarrow pk$
$m' \leftarrow \mathsf{JES.Dec}(pp, sk_r, c)$
$m||t'||\sigma \leftarrow m'$
Return $(pk_r||m, \sigma)$
```

```
$\underline{\text{SC}(pk_r, m)}$
$t \leftarrow pk$
$\sigma \leftarrow\!\$\ \text{SIGN}(pk_r||m)$
Return $\mathsf{JES.Enc}(pp, pk_r, m||t||\sigma)$

$\underline{\text{USC}(pk_s, c)}$
$t \leftarrow pk_s$
$m' \leftarrow \text{DEC}(c)$
If $(m' = \perp)$ then Return $\perp$
$m||t'||\sigma \leftarrow m'$
If $(t \neq t')$ then Return $\perp$
If $(!\mathsf{JES.Verify}(pp, pk_s, pk||m, \sigma))$
   then Return $\perp$
Return $m$
```

If A outputs a winning tuple (sk_r, pk_r, c), then $m||t||\sigma \leftarrow \mathsf{JES.Dec}(pp, sk_r, c)$ is such that $\mathsf{JES.Verify}(pp, pk, pk_r||m, \sigma)$ returns true, and A never queried pk_r, m to SC, so A_1 never queried $pk_r||m$ to its SIGN oracle. Adversary A_1 wins whenever A does, establishing Equation (3.11). ▮

JOINT SIGNCRYPTION, ENCRYPTION, AND SIGNATURE. While efficient signcryption schemes using a single short keypair for both sender and receiver roles are interesting in their own right, in [91] we consider an extended primitive which additionally allows users to use their keypair as part of an ordinary encryption and signature scheme, i.e. we consider a scheme implementing the functionality of signcryption, encryption and signature using a single keypair. This type of scheme consists of algorithms $\mathsf{Pg}, \mathsf{Kg}, \mathsf{Sc}, \mathsf{Usc}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{Sign}, \mathsf{Verify}$ such that $\mathsf{Pg}, \mathsf{Kg}, \mathsf{Sc}, \mathsf{Usc}$ form a signcryption scheme, $\mathsf{Pg}, \mathsf{Kg}, \mathsf{Enc}, \mathsf{Dec}$ form an encryption scheme and $\mathsf{Pg}, \mathsf{Kg}, \mathsf{Sign}, \mathsf{Verify}$ form a signature scheme.

As in the case of joint encryption and signature, a joint signcryption, encryption, and signature scheme which is jointly secure can trivially be constructed by concatenating the keys of independent signcryption, encryption and signature schemes. However, as with JES, we are interested in schemes that are more efficient (in terms of public key size and other measures) than this type of trivial construction. Note also that while an encryption or signature scheme can be constructed from a signcryption scheme by attaching an honestly generated "dummy" keypair to the public parameters to be used in combination with the signcrypt and unsigncrypt algorithms, this construction will not be jointly secure if a single keypair is used for the signcryption, signature and encryption components.

Security of such a scheme is defined in the natural manner, by including additional oracles to the signcryption and JES security games as necessary. In [91] we construct a joint signcryption, encryption, and signature scheme from a JES scheme JES, using the $\mathsf{SC}[\mathsf{JES}]$ construction in parallel with JES, with a similar domain separation trick to that in the JES scheme construction of Section 3.5 to ensure security. The construction uses a tag-based JES scheme, the algorithms and security of which are defined in [91], together with tag-based versions of the JES scheme constructions of Section 3.5 and Section 3.6.

3.9 Conclusion

We revisited the topic of joint encryption and signature, focussing on the construction of schemes in the standard model, an issue not fully addressed in prior work. We gave a general construction of a joint encryption and signature scheme from IBE, as well as a more efficient concrete construction based on pairings. Using BN curves, these can be efficiently instantiated at high security levels and have performance that is competitive with the best schemes arising from the Cartesian product construction. Our results fill the gap left open in the original work of Haber and Pinkas [72], of constructing standard model secure joint encryption and signature schemes in which the encryption and signature components share an identical keypair. An interesting open problem is to construct efficient joint encryption and signature schemes in the standard model without using pairings. For example, is it possible to obtain joint security in the discrete log or in the RSA setting, in the standard model?

We also considered the construction of signcryption schemes from joint encryption and signature schemes, giving a method to produce a triple of schemes (signcryption, encryption, and signature) that are jointly secure in an appropriate and strong security model, from any jointly secure tag-based JES scheme. This leads to efficient standard model signcryption schemes in which a single short keypair can be used for both sender and receiver functions.

Our work points the way to an interesting new research area in cryptography, which closely relates to and generalises the topic of cryptographic agility [2]. The general question can be posed as follows: under what conditions is it safe to use the same key (or keypair) across multiple instantiations of the *same* or *different* cryptographic primitives?

Chapter 4

Related-Key Attack Security for IBE

Contents

This chapter concerns security under related-key attacks (RKA), where an adversary can modify a stored secret key and observe the outputs of the system as it operates under this new key. We provide a framework enabling the construction of RKA-secure identity-based encryption schemes, and show how specific instantiations of the framework yield IBE schemes secure against adversaries deriving new keys through affine and polynomial transformations of the master secret key.

4.1 Introduction

A related-key attack (RKA) allows an adversary to modify a stored secret key and observe outputs of the system under the new key. RKAs were first conceived as tools for the cryptanalysis of blockciphers [80, 24]. However, the ability of attackers to modify keys stored in memory via tampering [33, 25] 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.

Furthermore, RKA security is finding applications beyond providing protection against tampering-based side-channel attacks [64], including instantiating random oracles in higher-level protocols and improving efficiency [8, 5].

The theoretical foundations of RKA security were laid by Bellare and Kohno [18], who treated the case of PRFs and PRPs. They model the transformations the adversary can apply to the target key as functions ϕ, and parameterise the definition by the class Φ of such functions. This parameterisation is necessary as without restrictions on the functions an adversary can apply, security is unachievable. For example for PRFs, a constant function $\phi(K) = c$ sets the key to a known value c that the adversary can then use to compute the function himself, and compare the output with that of his PRF oracle.

Provably achieving security against RKAs, however, has proven extremely challenging. We aim 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.

The primitive we target in this chapter is IBE. RKA security for this primitive was defined by Bellare, Cash, and Miller [14]. For future reference we define a few relevant classes of functions over the space MSKSp of master keys. The set $\Phi^c = \{\phi_c\}_{c \in \mathsf{MSKSp}}$ with $\phi_c(msk) = c$ is the set of constant functions. If MSKSp is a group under an operation $*$ then $\Phi^{\text{lin}} = \{\phi_a\}_{a \in \mathsf{MSKSp}}$ with $\phi_a(msk) = a * msk$ is the class of linear functions. (Here $*$ could be multiplication or addition.) If MSKSp is a field we let $\Phi^{\text{aff}} = \{\phi_{a,b}\}_{a,b \in \mathsf{MSKSp}}$ with $\phi_{a,b}(msk) = a \cdot msk + b$ be the class of affine functions and $\Phi^{\text{poly}(d)} = \{\phi_q\}_{q \in \mathsf{MSKSp}_d[x]}$ with $\phi_q(msk) = q(msk)$ the class of polynomial functions, where q ranges over the set $\mathsf{MSKSp}_d[x]$ of polynomials over MSKSp of degree at most d. RKA security increases and is a more ambitious target as we move from Φ^{lin} to Φ^{aff} to $\Phi^{\text{poly}(d)}$.

We begin by presenting attacks showing that existing IBE schemes such as those of Boneh-Franklin [34] and Waters [98] are not RKA secure, even for Φ^{lin}. This means we must seek new designs.

We then present a framework for constructing RKA-secure IBE schemes. It is an adaptation of the framework of Bellare and Cash [13] 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 $\Phi = \Phi^{\mathrm{lin}}$ but even for $\Phi = \Phi^{\mathrm{aff}}$. We additionally show that these schemes support efficient collision-resistant identity renaming. As a consequence we obtain Φ^{aff}-RKA-secure IBE schemes based on the same assumptions used to prove standard IBE security of the base IBE schemes.

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

However, we can go further, providing a $\Phi^{\mathrm{poly}(d)}$-RKA-secure IBE scheme. Our scheme is an extension of Waters' scheme [98]. 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.

4.2 Preliminaries

RKD FUNCTIONS AND CLASSES. We say that ϕ is a related-key deriving (RKD) function over a set MSKSp if $\phi \in \mathsf{Fun}(\mathsf{MSKSp}, \mathsf{MSKSp})$. We say that Φ is a class of RKD functions over MSKSp if $\Phi \subseteq \mathsf{Fun}(\mathsf{MSKSp}, \mathsf{MSKSp})$ and $\mathsf{id} \in \Phi$ where id is the identity function on MSKSp. In our constructs, MSKSp will have an algebraic structure, such as being a group, ring or field. In the last case, for $a, b \in \mathsf{MSKSp}$ we define $\phi_b^+, \phi_a^*, \phi_{a,b}^{\mathrm{aff}} \in \mathsf{Fun}(\mathsf{MSKSp}, \mathsf{MSKSp})$ via $\phi_b^+(msk) = msk{+}b$, $\phi_a^*(msk) = a{\cdot}msk$, and $\phi_{a,b}^{\mathrm{aff}}(msk) = a \cdot msk + b$ for all $msk \in \mathsf{MSKSp}$. For a polynomial q over field MSKSp, we define $\phi_q^{\mathrm{poly}}(msk) = q(msk)$ for all $msk \in \mathsf{MSKSp}$. We let $\Phi^+ = \{ \phi_b^+ : b \in \mathsf{MSKSp} \}$ be the class of additive RKD functions, $\Phi^* = \{ \phi_a^* : a \in \mathsf{MSKSp} \}$ the class of multiplicative RKD functions, $\Phi^{\text{aff}} = \{\phi^{\text{aff}}_{a,b} : a, b \in \mathsf{MSKSp}\}$ the class of affine RKD functions, and for any fixed positive integer d, $\Phi^{\text{poly}(d)} = \{\, \phi^{\text{poly}}_q \,:\, \deg q \leq d \,\}$ 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 *msk* such that $\phi(msk) \neq \phi'(msk)$, but there could also be keys *msk* on which $\phi(msk) = \phi'(msk)$. 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 $\phi(msk) \neq \phi'(msk)$ for *all* $msk \in \mathsf{MSKSp}$. With the exception of [69], all previous constructions of Φ-RKA-secure primitives with proofs of security have been for claw-free classes [18, 86, 66, 13, 14, 99]. In particular, key fingerprints are defined in [13] in such a way that their assumption of a Φ-key fingerprint automatically implies that Φ is claw-free. We provide the first proofs of achievability of RKA security for non-trivial classes with claws for a primitive other than CPA-secure symmetric encryption.

IBE SYNTAX. The usual IBE syntax specifies a single master key generation algorithm that produces *msk*, *mpk* 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. Let us say that IBE is canonical if the operation $(msk, mpk) \leftarrow_{\$} \mathsf{IBE.MKg}(pp)$ picks *msk* at random from a finite, non-empty set we denote $\mathsf{IBE.MSKSp}(pp)$, and then applies to (pp, msk) a PT deterministic master public-key derivation function we denote $\mathsf{IBE.MPK}$ to get *mpk*. 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 all IBE schemes are canonical.

We make an important distinction between parameters and the master public key, namely that the former may not depend on *msk* while the latter might. Parameters will be groups, group generators, pairings and the like. They will be fixed and available to all algorithms. The RKD function set Φ will depend on these parameters, as they define the master secret key space. For example if the master secret key space is a group the parameters will define its order. The value *pp* is not explicitly input to an RKD function ϕ but is available to it.

In order to define a satisfiable notion of security, it is necessary to limit the material an adversary can modify. The adversary can modify only user-specific inputs to algorithms, that is key material and values derived from it. One can imagine a scenario where shared parameters are hard-coded into a device and therefore cannot

```
main IND-RKA$^A_{\mathsf{IBE},\Phi}(\lambda)$
$b \leftarrow_\$ \{0,1\}$ ; $c^* \leftarrow \perp$ ; $u^* \leftarrow \perp$ ; $U \leftarrow \emptyset$
$pp \leftarrow_\$ \mathsf{IBE.Pg}(1^\lambda)$
$msk \leftarrow_\$ \mathsf{IBE.MSKSp}(pp)$
$mpk \leftarrow \mathsf{IBE.MPK}(pp, msk)$
$b' \leftarrow_\$ A^{\mathrm{KD,LR}}(pp, mpk)$
Return $(b = b')$

proc KD$(\phi, u)$
If $(\phi \notin \Phi)$ then Return $\perp$
$msk' \leftarrow \phi(msk)$
If $(msk' = msk)$ then $U \leftarrow U \cup \{u\}$
If $(u^* \in U)$ then Return $\perp$
Return $\mathsf{IBE.UKg}(pp, msk', u)$

proc LR$(u, m_0, m_1)$
If $(c^* \neq \perp)$ then Return $\perp$
If $(|m_0| \neq |m_1|)$ then Return $\perp$
$u^* \leftarrow u$
If $(u^* \in U)$ then Return $\perp$
$c^* \leftarrow_\$ \mathsf{IBE.Enc}(pp, mpk, u^*, m_b)$
Return $c^*$
```

Figure 4.1: Game IND-RKA defining Φ-RKA security of identity-based encryption scheme IBE.

be modified, with only user-specific values in modifiable memory. In the case of IBE, the adversary can then modify through the RKD functions the master secret key used in the key derivation algorithm. Note that this algorithm is not given the master public key, since as a user-specific value it would also be subject to modification by the adversary. Canonicity of the IBE scheme means an algorithm given the master secret key is able to derive from it the master public key should it need to.

RKA-secure IBE. We define RKA security of IBE schemes for a class Φ of RKD functions over $\mathsf{IBE.MSKSp}(pp)$ following [14]. We say that IBE is Φ-RKA secure if $\mathbf{Adv}^{\text{ind-rka}}_{\mathsf{IBE},\Phi,A,}(\cdot)$ is negligible for all PT adversaries A, where

$$\mathbf{Adv}^{\text{ind-rka}}_{\mathsf{IBE},\Phi,A}(\lambda) = 2\Pr[\text{IND-RKA}^A_{\mathsf{IBE},\Phi}(\lambda)] - 1$$

and game IND-RKA is in Figure 4.1. 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 *msk*, cannot check whether or not the IBE adversary succeeded. This difficulty is easily resolved if Φ is claw-free 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 PKE and signature schemes.

Notice that in this game, the RKD functions are applied only to the master secret key *msk*. One could consider definitions in which the RKD functions are instead applied to user secret keys *usk*. This may be more realistic in applications, where one might expect *msk* to be better protected against RKAs than the *usk*. One could also consider a mixed model where RKD functions are applied to both *msk* and *usk*. Since we are ultimately interested in the application of our results for IBE to the construction of PKE and signature schemes using the results of [14], we follow their original formulation of Φ-RKA security of IBE.

4.3 Existing IBE Schemes Under Related-Key Attacks

The algorithms of the Boneh-Franklin BasicIdent IBE scheme [34] are given in Figure 4.2. The parameters pp of the scheme are groups $\mathbb{G}, \mathbb{G}_T$ of prime order p, a symmetric pairing $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$, a generator g of $\mathbb{G}$ and hash functions $\mathsf{H}_1 : \{0,1\}^* \to \mathbb{G}$, $\mathsf{H}_2 : \mathbb{G}_T \to \{0,1\}^n$ which are modelled as random oracles in the security analysis. The identity space $\mathsf{USp}(pp)$ is $\{0,1\}^*$, the master secret key space $\mathsf{MSKSp}(pp)$ is $\mathbb{Z}_p^*$, and the message space $\mathsf{MSp}(pp)$ is $\{0,1\}^n$. This scheme is adaptively secure in the usual model for IBE security, under the Bilinear Diffie-Hellman (BDH) assumption.

Theorem 4.3.1 ([34], Theorem 4.1) *Let A be an adversary against the adaptive security of* BF *making q_k key derivation queries and q_{H_2} queries to random oracle* H_2. *Then there is an algorithm A_1 solving the Bilinear Diffie-Hellman problem such that*

$$\mathbf{Adv}^{\text{ind-aid}}_{\mathsf{BF},A}(\lambda) \leq \frac{e(1+q_k)q_{\mathsf{H}_2}}{2} \cdot \mathbf{Adv}^{\text{bdh}}_{\mathsf{G},A_1}(\lambda) \; .$$

The algorithms of the Waters IBE scheme [98] are also given in Figure 4.2. The parameters pp of the scheme are groups $\mathbb{G}, \mathbb{G}_T$ of prime order p, a symmetric pairing $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$, generators g, g_1 of $\mathbb{G}$ and group elements $h_0, \ldots, h_n \in \mathbb{G}$ specifying the hash function $\mathsf{H}(u) = h_0 \prod_{i \in u} h_i$, where $i \in u$ denotes the ith bit of the bitstring u is set to 1. The identity space $\mathsf{USp}(pp)$ is $\{0,1\}^n$, the master secret key space

```
$\underline{\mathsf{BF.Pg}(1^\lambda)}$:
$(p, \mathbb{G}, e) \leftarrow_\$ \mathsf{G.Pg}(1^\lambda)$
$g \leftarrow_\$ \mathbb{G}^*$
Return $(p, \mathbb{G}, e, g, \mathsf{H}_1, \mathsf{H}_2)$

$\underline{\mathsf{BF.MPK}(pp, msk)}$:
$(p, \mathbb{G}, e, g, \mathsf{H}_1, \mathsf{H}_2) \leftarrow pp$
$mpk \leftarrow g^{msk}$
Return $mpk$

$\underline{\mathsf{BF.UKg}(pp, msk, u)}$:
$(p, \mathbb{G}, e, g, \mathsf{H}_1, \mathsf{H}_2) \leftarrow pp$
$usk \leftarrow \mathsf{H}_1(u)^{msk}$
Return $usk$

$\underline{\mathsf{BF.Enc}(pp, mpk, u, m)}$:
$(p, \mathbb{G}, e, g, \mathsf{H}_1, \mathsf{H}_2) \leftarrow pp$
$t \leftarrow_\$ \mathbb{Z}_p$
$c_1 \leftarrow g^t$
$c_2 \leftarrow \mathsf{H}_2(e(mpk, \mathsf{H}_1(u))^t) \oplus m$
Return $(c_1, c_2)$

$\underline{\mathsf{BF.Dec}(pp, usk, c)}$:
$(p, \mathbb{G}, e, g, \mathsf{H}_1, \mathsf{H}_2) \leftarrow pp$
$(c_1, c_2) \leftarrow c$
$m \leftarrow c_2 \oplus \mathsf{H}_2(e(usk, c_1))$
Return $m$
```

```
$\underline{\mathsf{Wat.Pg}(1^\lambda)}$:
$(p, \mathbb{G}, e) \leftarrow_\$ \mathsf{G.Pg}(1^\lambda)$
$g, g_1, h_0, \ldots, h_n \leftarrow_\$ \mathbb{G}^*$
Return $(p, \mathbb{G}, e, g, g_1, h_0, \ldots, h_n)$

$\underline{\mathsf{Wat.MPK}(pp, msk)}$:
$(p, \mathbb{G}, e, g, g_1, h_0, \ldots, h_n) \leftarrow pp$
$mpk \leftarrow g^{msk}$
Return $mpk$

$\underline{\mathsf{Wat.UKg}(pp, msk, u)}$:
$(p, \mathbb{G}, e, g, g_1, h_0, \ldots, h_n) \leftarrow pp$
$r \leftarrow_\$ \mathbb{Z}_p$
$usk_1 \leftarrow g_1^{msk} \cdot \mathsf{H}(u)^r$ ; $usk_2 \leftarrow g^r$
Return $(usk_1, usk_2)$

$\underline{\mathsf{Wat.Enc}(pp, mpk, u, m)}$:
$(p, \mathbb{G}, e, g, g_1, h_0, \ldots, h_n) \leftarrow pp$
$t \leftarrow_\$ \mathbb{Z}_p$
$c_1 \leftarrow g^t$
$c_2 \leftarrow \mathsf{H}(u)^t$
$c_3 \leftarrow e(mpk, g_1)^t \cdot m$
Return $(c_1, c_2, c_3)$

$\underline{\mathsf{Wat.Dec}(usk, c)}$:
$(p, \mathbb{G}, e, g, g_1, h_0, \ldots, h_n) \leftarrow pp$
$(c_1, c_2, c_3) \leftarrow c$
$m \leftarrow c_3 \cdot \frac{e(usk_2, c_2)}{e(usk_1, c_1)}$
Return $m$
```

Figure 4.2: Left: Boneh-Franklin IBE scheme BF. Right: Waters IBE scheme Wat.

$\mathsf{MSKSp}(pp)$ is $\mathbb{Z}_p^*$, and the message space $\mathsf{MSp}(pp)$ is $\mathbb{G}_T$. The Waters IBE scheme is also adaptively secure in the usual model for IBE security, under the Decisional Bilinear Diffie-Hellman (DBDH) assumption.

Theorem 4.3.2 ([98], Theorem 1) *Let A be an adversary against the adaptive security of* Wat *making q_k key derivation queries. Then there is an algorithm A_1 solving the Decision Bilinear Diffie-Hellman problem such that*

$$\mathbf{Adv}^{\text{ind-aid}}_{\mathsf{Wat},A}(\lambda) \leq 32(n+1) \cdot q_k \cdot \mathbf{Adv}^{\text{dbdh}}_{\mathsf{G},A_1}(\lambda)\ .$$

The Waters IBE scheme is not RKA secure if Φ includes a function $\phi^*_a(msk) = a \cdot msk$. A call to the key derivation oracle with any such ϕ yields a user secret key $(usk_1, usk_2) = (g_1^{a \cdot msk} \cdot \mathsf{H}(u)^r, g^r)$. Raising this to $a^{-1} \mod p$ gives $(usk'_1, usk'_2) = (g_1^{msk} \cdot \mathsf{H}(u)^{ra^{-1}}, g^{ra^{-1}})$, so that (usk'_1, usk'_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.

4.4 Framework for Deriving RKA-Secure IBE Schemes

In the previous section we saw that the Boneh-Franklin and Waters IBE 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 IBE is Φ-key-malleable if there is an algorithm T, called the key simulator, which, given pp, mpk, an identity u, a decryption key $usk' \leftarrow_{\$} \mathsf{IBE.UKg}(pp, msk, u)$ for u under msk, and an RKD function $\phi \in \Phi$, outputs a decryption key usk for u under master secret key $\phi(msk)$ that is

```
MAIN $\mathrm{KMReal}^{M}_{\mathsf{IBE},\Phi}(\lambda)$
$pp \leftarrow\!\$\ \mathsf{IBE.Pg}(1^{\lambda})$
$msk \leftarrow\!\$\ \mathsf{IBE.MSKSp}(pp)$
$mpk \leftarrow\!\$\ \mathsf{IBE.MPK}(pp, msk)$
$b' \leftarrow\!\$\ A^{\mathrm{KD}}(pp, mpk)$
Return $(b' = 1)$

proc $\mathrm{KD}(\phi, u)$
If $(\phi \notin \Phi)$ then Return $\perp$
$msk' \leftarrow \phi(msk)$
Return $\mathsf{IBE.UKg}(pp, msk', u)$
```

```
MAIN $\mathrm{KMSim}^{M}_{\mathsf{IBE},\Phi,T}(\lambda)$
$pp \leftarrow\!\$\ \mathsf{IBE.Pg}(1^{\lambda})$
$msk \leftarrow\!\$\ \mathsf{IBE.MSKSp}(pp)$
$mpk \leftarrow\!\$\ \mathsf{IBE.MPK}(pp, msk)$
$b' \leftarrow\!\$\ A^{\mathrm{KD}}(pp, mpk)$
Return $(b' = 1)$

proc $\mathrm{KD}(\phi, u)$
If $(\phi \notin \Phi)$ then Return $\perp$
$usk' \leftarrow \mathsf{IBE.UKg}(pp, msk, u)$
Return $T(pp, mpk, u, usk', \phi)$
```

Figure 4.3: Games KMReal and KMSim defining key-malleability of IBE scheme IBE.

distributed identically to the output of $\mathsf{IBE.UKg}(pp, \phi(msk), u)$. The formalisation 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

$$\Pr[\mathrm{KMReal}^{M}_{\mathsf{IBE},\Phi}(\lambda)] = \Pr[\mathrm{KMSim}^{M}_{\mathsf{IBE},\Phi,T}(\lambda)]$$

for all (not necessarily computationally bounded) adversaries M, where the games are as in Figure 4.3.

Using key-malleability. 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 catch that motivates 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

$$\mathbf{Adv}^{\text{ind-rka}}_{\mathsf{IBE},\Phi,\overline{A}}(\lambda) \leq \mathbf{Adv}^{\text{ind-aid}}_{\mathsf{IBE},A}(\lambda)\ . \qquad (4.1)$$

On input pp, mpk, adversary A runs $\overline{A}(pp, mpk)$. When the latter makes a $\mathrm{KD}(\phi, u)$ query, A lets $usk \leftarrow \mathrm{KD}(u)$, where KD is A's own key derivation oracle. It then lets $\overline{usk} \leftarrow T(pp, mpk, u, usk, \phi)$ and returns $\overline{usk}$ to $\overline{A}$. Key-malleability tells us that $\overline{usk}$ 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 b', adversary A does the same.

The simulation seems perfect, so we appear to have established Equation (4.1). 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 $\phi(msk) \neq msk$; 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 $\phi(msk) \neq msk$ 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 $\mathrm{KD}(u)$ 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-derivation 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 $\mathsf{IBE} = (\mathsf{Pg}, \mathsf{MKg}, \mathsf{UKg}, \mathsf{Enc}, \mathsf{Dec})$ denote the given, base IBE scheme, with parameters pp and identity space $\mathsf{USp}(pp)$. A renaming scheme is a pair $(\mathrm{SI}, \mathrm{PI})$ of functions where SI: $\{pp\} \times \mathsf{MSKSp}(pp) \times \overline{\mathsf{USp}}(pp) \to \mathsf{USp}(pp)$ and PI: $\{pp\} \times [\mathsf{MPK}(pp, \mathsf{MSKSp}(pp))] \times \overline{\mathsf{USp}}(pp) \times \Phi \to \mathsf{USp}(pp)$ where $\overline{\mathsf{USp}}(pp)$, 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 $\mathrm{SI}(pp, \phi(msk), \overline{u}) = \mathrm{PI}(pp, mpk, \overline{u}, \phi)$ for all $msk \in \mathsf{MSKSp}(pp)$, all $mpk \in [\mathsf{MPK}(pp, msk)]$, 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 $\mathsf{IBE} = (\mathsf{Pg}, \mathsf{MKg}, \mathsf{UKg}, \mathsf{Enc}, \mathsf{Dec})$ to a new IBE scheme $\overline{\mathsf{IBE}} = (\mathsf{Pg}, \mathsf{MKg}, \overline{\mathsf{UKg}}, \overline{\mathsf{Enc}}, \mathsf{Dec})$. As the notation indicates, the parameter generation algorithm, master public key generation algorithm and decryption algorithm are unchanged. The other algorithms are defined by

$$\overline{\mathsf{UKg}}(pp, msk, \overline{u}) = \mathsf{UKg}(pp, msk, \mathrm{SI}(pp, msk, \overline{u}))$$

and

$$\overline{\mathsf{Enc}}(pp, mpk, \overline{u}, m) = \mathsf{Enc}(pp, mpk, \mathrm{PI}(pp, mpk, \overline{u}, \mathsf{id}), m) \ .$$

We clarify that algorithms of the new IBE scheme do not, and cannot, have as input the RKD functions ϕ used by an 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 msk but not mpk (recall we cannot give it mpk because otherwise it becomes subject to the RKA) and thus uses the secret renaming function. On the other hand the encryption algorithm has mpk but obviously not msk 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 $\overline{\mathsf{USp}}(pp)$ from which the renaming functions draw their identity inputs.

The above compatibility requirement implies that

$$\mathrm{SI}(pp, msk, \overline{u}) = \mathrm{PI}(pp, mpk, \overline{u}, \mathsf{id}) \ .$$

From this it follows that $\overline{\mathsf{IBE}}$ 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

$$\mathrm{SI}(pp, msk, \overline{u}) = \overline{u} = \mathrm{PI}(pp, mpk, \overline{u}, \phi) \ .$$

This satisfies the compatibility condition. However, the transformed IBE scheme $\overline{\mathsf{IBE}}$ ends up identical to the base scheme 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 $(\mathrm{SI}, \mathrm{PI})$ will be required to have a collision-resistance property. In its simplest and strongest form the requirement is that

$$(\phi(msk), \overline{u}_1) \neq (msk, \overline{u}_2) \quad \Rightarrow \quad \mathrm{SI}(pp, \phi(msk), \overline{u}_1) \neq \mathrm{SI}(pp, msk, \overline{u}_2)$$

for all $pp \in \mathsf{PSp}$, all $msk \in \mathsf{MSKSp}(pp)$, all $\overline{u}_1, \overline{u}_2 \in \overline{\mathsf{USp}}(pp)$, and all $\phi \in \Phi$. This *statistical collision-resistance* is enough to prove that $\overline{\mathsf{IBE}}$ is Φ-RKA secure if IBE is Φ-key-malleable. Theorem 4.4.1, the main result of this chapter, states this formally.

```
MAIN $\text{IND-RKA}^{\overline{A}}_{\mathsf{IBE}}(\lambda)$ / $\boxed{G^{\overline{A}}_0(\lambda) \text{ / } G^{\overline{A}}_1(\lambda) \text{ / } G^{\overline{A}}_2(\lambda)}$
$b \leftarrow_{\$} \{0,1\}$ ; $c^* \leftarrow \perp$ ; $\overline{u}^* \leftarrow \perp$ ; $\overline{U} \leftarrow \emptyset$  $\boxed{u^* \leftarrow \perp \text{ ; } U \leftarrow \emptyset}$
$pp \leftarrow_{\$} \mathsf{IBE.Pg}(1^\lambda)$
$msk \leftarrow_{\$} \mathsf{IBE.MSKSp}(pp)$
$mpk \leftarrow \mathsf{IBE.MPK}(pp, msk)$
$b' \leftarrow_{\$} \overline{A}^{\text{KD,LR}}(pp, mpk)$
Return $(b = b')$

proc KD$(\phi, \overline{u})$ // $\text{IND-RKA}^{\overline{A}}_{\mathsf{IBE}}(\lambda)$ / $\boxed{G^{\overline{A}}_0(\lambda)}$
If $(\phi \notin \Phi)$ then Return $\perp$
$msk' \leftarrow \phi(msk)$
If $(msk' = msk)$ then $\overline{U} \leftarrow \overline{U} \cup \{\overline{u}\}$
If $(\overline{u}^* \in \overline{U})$ then Return $\perp$
$u \leftarrow \text{SI}(pp, msk', \overline{u})$
$\boxed{U \leftarrow U \cup \{u\}}$
$\boxed{\text{If } (u^* \in U) \text{ then Return } \perp}$
Return $\mathsf{IBE.UKg}(pp, msk', u)$

proc KD$(\phi, \overline{u})$ // $\boxed{G^{\overline{A}}_1(\lambda)}$ / $G^{\overline{A}}_2(\lambda)$
If $(\phi \notin \Phi)$ then Return $\perp$
$u \leftarrow \text{PI}(pp, mpk, \overline{u}, \phi)$ ; $U \leftarrow U \cup \{u\}$ ; If $(u^* \in U)$ then Return $\perp$
$\boxed{\text{Return } \mathsf{IBE.UKg}(pp, \phi(msk), u)}$
$usk \leftarrow \mathsf{IBE.UKg}(pp, msk, u)$
Return $T(pp, mpk, u, usk, \phi)$

proc LR$(\overline{u}, m_0, m_1)$ // $\text{IND-RKA}^{\overline{A}}_{\mathsf{IBE}}(\lambda)$ / $\boxed{G^{\overline{A}}_0(\lambda)}$
If $(c^* \neq \perp)$ then Return $\perp$
If $(|m_0| \neq |m_1|)$ then Return $\perp$
$\overline{u}^* \leftarrow \overline{u}$
If $(\overline{u}^* \in \overline{U})$ then Return $\perp$
$u^* \leftarrow \text{PI}(pp, mpk, \overline{u}, \mathsf{id})$
$\boxed{u^* \leftarrow \text{SI}(pp, msk, \overline{u}) \text{ ; If } (u^* \in U) \text{ then Return } \perp}$
$c^* \leftarrow_{\$} \mathsf{IBE.Enc}(pp, mpk, u^*, m_b)$
Return $c^*$

proc LR$(\overline{u}, m_0, m_1)$ // $G^{\overline{A}}_1(\lambda)$ / $G^{\overline{A}}_2(\lambda)$
If $(c^* \neq \perp)$ then Return $\perp$
If $(|m_0| \neq |m_1|)$ then Return $\perp$
$u^* \leftarrow \text{PI}(pp, mpk, \overline{u}, \mathsf{id})$
If $(u^* \in U)$ then Return $\perp$
$c^* \leftarrow_{\$} \mathsf{IBE.Enc}(pp, mpk, u^*, m_b)$
Return $c^*$
```

Figure 4.4: Games used in the proof of Theorem 4.4.1.

Theorem 4.4.1 *Let* $\mathsf{IBE} = (\mathsf{Pg}, \mathsf{MKg}, \mathsf{UKg}, \mathsf{Enc}, \mathsf{Dec})$ *be an adaptively-secure* Φ*-key-malleable IBE scheme with key simulator* T*. Let* $(\mathrm{SI}, \mathrm{PI})$ *be a statistically collision-resistant renaming scheme. Let* $\overline{\mathsf{IBE}} = (\mathsf{Pg}, \mathsf{MKg}, \overline{\mathsf{UKg}}, \overline{\mathsf{Enc}}, \mathsf{Dec})$ *be obtained from* IBE *and renaming scheme* $(\mathrm{SI}, \mathrm{PI})$ *via the transform* **IRT** *described above. Then* $\overline{\mathsf{IBE}}$ *is* Φ*-RKA secure.*

Proof of Theorem 4.4.1: Let $\overline{A}$ be a Φ-RKA adversary against $\overline{\mathsf{IBE}}$. We construct an adversary A such that

$$\mathbf{Adv}^{\text{ind-rka}}_{\overline{\mathsf{IBE}},\Phi,\overline{A}}(\lambda) \leq \mathbf{Adv}^{\text{ind-aid}}_{\mathsf{IBE},A}(\lambda) \; .$$

The proof uses the games in Figure 4.4.

In answering a $\mathrm{KD}(\phi, \overline{u})$ query, game $\text{IND-RKA}_{\overline{\mathsf{IBE}},\Phi}$ must use the key-generation algorithm $\overline{\mathsf{UKg}}$ of the new scheme $\overline{\mathsf{IBE}}$ but with master secret key $msk' = \phi(msk)$. From the definition of $\overline{\mathsf{UKg}}$, it follows that not only is the key-generation done under msk', but also the identity renaming. LR, correspondingly, uses $\overline{\mathsf{Enc}}$, and thus the public renaming function PI.

The adversary A we aim to construct will not know msk. A central difficulty in the simulation is thus the dashed-boxed code of the KD procedure of $\text{IND-RKA}_{\overline{\mathsf{IBE}},\Phi}$ where the response provided to $\overline{A}$ depends on the result of a test involving msk, 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_0 moves the identity renaming up before the list of queried identities is updated and then adds the transformed identity to the list. LR is likewise modified so its test now involves the transformed (rather than original) identities. Additionally, the secret renaming function SI is used instead of PI, a modification allowed by the compatibility property of the renaming scheme. We claim this makes no difference, meaning

$$\Pr[\text{IND-RKA}^{\overline{A}}_{\overline{\mathsf{IBE}},\Phi}(\lambda)] = \Pr[\mathrm{G}_0^{\overline{A}}(\lambda)] \; .$$

Indeed, statistical collision-resistance tell us that $(msk', \overline{u}) = (msk, \overline{u}^*)$ iff $\mathrm{SI}(pp, msk', \overline{u}) = \mathrm{SI}(pp, msk, \overline{u}^*)$. This means the dashed-boxed code of $\text{IND-RKA}_{\overline{\mathsf{IBE}},\Phi}$ and the boxed code of G_0 are equivalent.

Compatibility is invoked to use PI in place of SI in both KD and in LR in G_1, so that

$$\Pr[\mathrm{G}_0^{\overline{A}}(\lambda)] = \Pr[\mathrm{G}_1^{\overline{A}}(\lambda)]\ .$$

Rather than use $\phi(msk)$ for key generation as in the boxed code of G_1, G_2 uses msk and then applies the key simulator T. We claim that key-malleability implies

$$\Pr[\mathrm{G}_1^{\overline{A}}(\lambda)] = \Pr[\mathrm{G}_2^{\overline{A}}(\lambda)]\ . \tag{4.2}$$

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

$$\Pr[\mathrm{KMReal}_{\mathsf{IBE},\Phi}^{M}(\lambda)] = \Pr[\mathrm{G}_1^{\overline{A}}(\lambda)] \qquad \text{and} \qquad \Pr[\mathrm{KMSim}_{\mathsf{IBE},\Phi,T}^{M}(\lambda)] = \Pr[\mathrm{G}_2^{\overline{A}}(\lambda)]\ .$$

Adversary M behaves as follows:

```
$M^{\mathrm{KD}}(pp, mpk)$
$b \leftarrow_\$ \{0,1\}$ ; $c^* \leftarrow \perp$ $u^* \leftarrow \perp$ ; $U \leftarrow \emptyset$
$b' \leftarrow_\$ \overline{A}^{\mathrm{KDSim},\mathrm{LR}}(pp, mpk)$
Return $(b = b')$

$\mathrm{KDSim}(\phi, \overline{u})$
$u \leftarrow \mathrm{PI}(pp, mpk, \overline{u}, \phi)$
$U \leftarrow U \cup \{u\}$
If $(u^* \in U)$ then Return $\perp$
Return $\mathrm{KD}(\phi, u)$

$\mathrm{LR}(\overline{u}, m_0, m_1)$
If $(c^* \neq \perp)$ then Return $\perp$
If $(|m_0| \neq |m_1|)$ then Return $\perp$
$u^* \leftarrow \mathrm{PI}(pp, mpk, \overline{u}, \mathsf{id})$
If $(u^* \in U)$ then Return $\perp$
$c^* \leftarrow_\$ \mathsf{IBE.Enc}(pp, mpk, u^*, m_b)$
Return $c^*$
```

If M is playing game KMReal then its KD oracle will behave as the boxed code in game G_1, while if M is playing game KMSim its KD oracle will behave as the dashed-boxed code in game G_2. If M is playing game KMReal then game G_1 is perfectly simulated, while if M is playing KMSim then game G_2 is perfectly simulated, so M returns 1 with the same probability that $\overline{A}$ wins in each case and by the key-malleability of IBE Equation (4.2) holds.

Finally, we design A so that

$$\mathbf{Adv}_{\mathsf{IBE},A}^{\mathsf{ind\text{-}aid}}(\lambda) = 2\Pr[\mathrm{G}_2^{\overline{A}}(\lambda)] - 1\ .$$

Adversary A behaves as follows:

```
$\underline{A^{\mathrm{KD,LR}}(pp, mpk)}$
$b' \leftarrow_{\$} \overline{A}^{\mathrm{KDSim,LRSim}}(pp, mpk)$
Return $b'$
```

```
$\underline{\mathrm{KDSim}(\phi, \overline{u})}$
If $(\phi \notin \Phi)$ then Return $\perp$
$u \leftarrow \mathrm{PI}(pp, mpk, \overline{u}, \phi)$
$usk \leftarrow \mathrm{KD}(u)$
$\overline{usk} \leftarrow T(pp, mpk, u, usk, \phi)$
Return $\overline{usk}$

$\underline{\mathrm{LR}(\overline{u}, m_0, m_1)}$
$u^* \leftarrow \mathrm{PI}(pp, mpk, \overline{u}, \phi)$
Return $\mathrm{LR}(u^*, m_0, m_1)$
```

A's own KD oracle will keep track of the queried identities u, so A provides a perfect simulation of G_2 for $\overline{A}$. ▮

4.5 Applying the Framework

Affine RKA-security 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 $usk' = \mathsf{H}_1(u)^{msk}$, so we can specify an algorithm T as follows:

$$T(pp, mpk, u, usk', \phi_{a,b}) : usk \leftarrow usk'^{a} \cdot \mathsf{H}_1(u)^b \text{ ; Return } usk \text{ .}$$

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

$$usk'^{a} \cdot \mathsf{H}_1(u)^b = \left(\mathsf{H}_1(u)^{msk}\right)^a \cdot \mathsf{H}_1(u)^b = \mathsf{H}_1(u)^{a \cdot msk + b} \text{ .}$$

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

Keys in the Waters IBE scheme are of the form $(usk'_1, usk'_2) = (g_1^{msk} \cdot \mathsf{H}(u)^r, g^r)$ for some r in $\mathbb{Z}_p$, so we can specify an algorithm T as follows:

```
$T(pp, mpk, u, usk', \phi_{a,b})$:
If $(a = 0)$ then $r \leftarrow_{\$} \mathbb{Z}_p$ ; $usk_1 \leftarrow g_1^b \cdot \mathsf{H}(u)^r$ ; $usk_2 \leftarrow g^r$
Else $usk_1 \leftarrow usk_1'^{a} \cdot g_1^b$ ; $usk_2 \leftarrow usk_2'^{a}$
Return $(usk_1, usk_2)$
```

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 $\phi_{a,b}(msk)$, now under randomness ra, since:

$$usk_1'^a \cdot g_1^b = \left(g_1^{msk} \cdot \mathsf{H}(u)^r\right)^a \cdot g_1^b = g_1^{a \cdot msk + b} \mathsf{H}(u)^{ra} \qquad usk_2'^a = g^{ra} \ .$$

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 $\mathsf{IBE.UKg}(pp, \phi(msk), u)$. Hence the Waters IBE scheme is key-malleable.

The same identity renaming scheme can be used for both IBE schemes. Namely, $\mathrm{SI}(pp, msk, \overline{u})$ returns $\overline{u}||g^{msk}$ and $\mathrm{PI}(pp, mpk, \overline{u}, \phi_{a,b})$ returns $\overline{u}||mpk^a \cdot g^b$. The compatibility requirement is satisfied and the renaming scheme is clearly collision-resistant since $(\overline{u}_1||g^{\phi(msk)} = \overline{u}_2||g^{msk}) \Rightarrow (\overline{u}_1 = \overline{u}_2) \wedge (\phi(msk) = msk)$. 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 4.4.1. 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 $\overline{u}||g^{msk}$ to be used in the renaming scheme.

For concreteness Figure 4.5 shows the algorithms of aff-BF and aff-Wat, the transformed versions of the Boneh-Franklin and Waters IBE schemes secure against affine related-key attacks. The following theorem is obtained by combining Theorem 4.4.1 with Theorem 4.3.1.

Theorem 4.5.1 *Assume the BDH problem is hard for* G. *Then* aff-BF *is* Φ^{aff}*-RKA secure. More concretely, let* $\overline{A}$ *be a* Φ^{aff}*-RKA adversary against* aff-BF *making* $q_k(\lambda)$ *key derivation queries and* $q_{\mathsf{H}_2}(\lambda)$ *queries to random oracle* H_2. *Then there is an algorithm* A *solving the Bilinear Diffie-Hellman problem such that*

$$\mathbf{Adv}^{\text{ind-rka}}_{\mathsf{aff\text{-}BF},\Phi^{\mathrm{aff}},\overline{A}}(\lambda) \leq \frac{e(1 + q_k(\lambda))q_{\mathsf{H}_2}(\lambda)}{2} \cdot \mathbf{Adv}^{\mathrm{bdh}}_{\mathsf{G},A}(\lambda) \ .$$

The following theorem is obtained by combining Theorem 4.4.1 with Theorem 4.3.2, and the running time of A below may be obtained in the same way. Concrete-security improvements would be obtained by using instead the analysis of Waters' scheme from [22].

Theorem 4.5.2 *Assume the DBDH problem is hard for* G. *Then* aff-Wat *is* Φ^{aff}*-RKA secure. More concretely, let* $\overline{A}$ *be a* Φ^{aff}*-RKA adversary against* aff-Wat *mak-*

```
aff-BF.Pg($1^\lambda$):
    $(p, \mathbb{G}, e) \leftarrow_\$ \mathsf{G.Pg}(1^\lambda)$
    $g \leftarrow_\$ \mathbb{G}^*$
    Return $(p, \mathbb{G}, e, g, \mathsf{H}_1, \mathsf{H}_2)$

aff-BF.MPK($pp, msk$):
    $(p, \mathbb{G}, e, g, \mathsf{H}_1, \mathsf{H}_2) \leftarrow pp$
    $mpk \leftarrow g^{msk}$
    Return $mpk$

aff-BF.$\overline{\mathsf{UKg}}$($pp, msk, \overline{u}$):
    $(p, \mathbb{G}, e, g, \mathsf{H}_1, \mathsf{H}_2) \leftarrow pp$
    $usk \leftarrow \mathsf{H}_1(\overline{u}||g^{msk})^{msk}$
    Return $usk$

aff-BF.$\overline{\mathsf{Enc}}$($pp, mpk, \overline{u}, m$):
    $(p, \mathbb{G}, e, g, \mathsf{H}_1, \mathsf{H}_2) \leftarrow pp$
    $t \leftarrow_\$ \mathbb{Z}_p$
    $c_1 \leftarrow g^t$
    $c_2 \leftarrow \mathsf{H}_2(e(mpk, \mathsf{H}_1(\overline{u}||mpk))^t) \oplus m$
    Return $(c_1, c_2)$

aff-BF.Dec($pp, usk, c$):
    $(p, \mathbb{G}, e, g, \mathsf{H}_1, \mathsf{H}_2) \leftarrow pp$
    $(c_1, c_2) \leftarrow c$
    $m \leftarrow c_2 \oplus \mathsf{H}_2(e(usk, c_1))$
    Return $m$
```

```
aff-Wat.Pg($1^\lambda$):
    $(p, \mathbb{G}, e) \leftarrow_\$ \mathsf{G.Pg}(1^\lambda)$
    $g, g_1, h_0, \ldots, h_n \leftarrow_\$ \mathbb{G}^*$
    Return $(p, \mathbb{G}, e, g, g_1, h_0, \ldots, h_n)$

aff-Wat.MPK($pp, msk$):
    $(p, \mathbb{G}, e, g, g_1, h_0, \ldots, h_n) \leftarrow pp$
    $mpk \leftarrow g^{msk}$
    Return $mpk$

aff-Wat.$\overline{\mathsf{UKg}}$($pp, msk, \overline{u}$):
    $(p, \mathbb{G}, e, g, g_1, h_0, \ldots, h_n) \leftarrow pp$
    $r \leftarrow_\$ \mathbb{Z}_p$
    $usk_1 \leftarrow g_1^{msk} \cdot \mathsf{H}(\overline{u}||g^{msk})^r$
    $usk_2 \leftarrow g^r$
    Return $(usk_1, usk_2)$

aff-Wat.$\overline{\mathsf{Enc}}$($pp, mpk, \overline{u}, m$):
    $(p, \mathbb{G}, e, g, g_1, h_0, \ldots, h_n) \leftarrow pp$
    $t \leftarrow_\$ \mathbb{Z}_p$
    $c_1 \leftarrow g^t$
    $c_2 \leftarrow \mathsf{H}(\overline{u}||mpk)^t$
    $c_3 \leftarrow e(mpk, g_1)^t \cdot m$
    Return $(c_1, c_2, c_3)$

aff-Wat.Dec($pp, usk, c$):
    $(p, \mathbb{G}, e, g, g_1, h_0, \ldots, h_n) \leftarrow pp$
    $(c_1, c_2, c_3) \leftarrow c$
    $m \leftarrow c_3 \cdot \frac{e(usk_2, c_2)}{e(usk_1, c_1)}$
    Return $m$
```

Figure 4.5: Left: Φ^{aff}-RKA-secure Boneh-Franklin IBE scheme aff-BF. Right: Φ^{aff}-RKA-secure Waters IBE scheme aff-Wat.

ing $q_k(\lambda)$ key derivation queries. Then there is an algorithm A solving the Decision Bilinear Diffie-Hellman problem such that

$$\mathbf{Adv}^{\text{ind-rka}}_{\mathsf{aff\text{-}Wat},\Phi^{\text{aff}},\overline{A}}(\lambda) \leq 32(n+1) \cdot q_k(\lambda) \cdot \mathbf{Adv}^{\text{dbdh}}_{\mathsf{G},A}(\lambda)\ .$$

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 $\Phi^{\text{poly}(d)}$, the set of all polynomials 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 $g^{msk^2}, \ldots, g^{msk^d}, {g_1}^{msk^2}, \ldots, {g_1}^{msk^d}$ alongside g^{msk}. These elements assist in achieving key-malleability for the set $\Phi^{\text{poly}(d)}$.

The master public-key generation algorithm of the extended Waters scheme is then

```
$\mathsf{MPK}(pp, msk)$:
$mpk \leftarrow \left(g^{msk}, g^{msk^2}, \ldots, g^{msk^d}, (g_1)^{msk^2}, \ldots, (g_1)^{msk^d}\right)$
Return $mpk$
```

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-aID sense for IBE) under a q-type extension of the standard DBDH assumption that we call the Extended Decisional Bilinear Diffie-Hellman (EDBDH) problem. Specifically, we say the EDBDH problem is hard for G if $\mathbf{Adv}^{\text{edbdh}}_{\mathsf{G},q(\cdot),A}(\cdot)$ is negligible for all PT adversaries A and every polynomially bounded function $q : \mathbb{N} \to \mathbb{N}$, where $\mathbf{Adv}^{\text{edbdh}}_{\mathsf{G},q(\lambda),A}(\lambda) = 2\Pr[\text{EDBDH}^{A}_{\mathsf{G},q(\lambda)}(\lambda)] - 1$ and game EDBDH is in Figure 4.6.

Theorem 4.5.3 *Let* $\mathsf{IBE} = (\mathsf{Pg}, \mathsf{MKg}, \mathsf{UKg}, \mathsf{Enc}, \mathsf{Dec})$ *be the extended Waters IBE scheme. Assume the EDBDH problem is hard for* G. *Then* IBE *is adaptively secure. More concretely, let A be an adversary against* IBE *making $q_k(\lambda)$ key derivation queries. Then there is an algorithm A_1 solving the Extended Decision Bilinear Diffie-Hellman problem such that*

$$\mathbf{Adv}^{\text{ind-aid}}_{\mathsf{IBE},A}(\lambda) \leq 32(n+1) \cdot q_k(\lambda) \cdot \mathbf{Adv}^{\text{edbdh}}_{\mathsf{G},q(\lambda),A_1}(\lambda)\ .$$

```
MAIN $\mathrm{EDBDH}^{A}_{\mathsf{G},q(\lambda)}(\lambda)$
$q \leftarrow q(\lambda)$
$(p, \mathbb{G}, \mathbb{G}_T, e) \leftarrow\$ \mathsf{G.Pg}(1^\lambda)$
$b \leftarrow\$ \{0,1\}$
$\alpha, \beta, \gamma \leftarrow\$ \mathbb{Z}_p^*$
$g \leftarrow\$ \mathbb{G}^*$
If $(b = 1)$ then $T \leftarrow e(g,g)^{\alpha\beta\gamma}$
Else $T \leftarrow\$ \mathbb{G}_T$
$b' \leftarrow\$ A^{\text{CHAL}}(p, \mathbb{G}, \mathbb{G}_T, e, g, g^\alpha, g^{\alpha^2}, \ldots, g^{\alpha^q}, g^\beta, g^{(\alpha^2)\beta}, g^{(\alpha^3)\beta}, \ldots, g^{(\alpha^q)\beta}, g^\gamma, T)$
Return $(b = b')$
```

Figure 4.6: Game EDBDH defining the Extended Decisional Bilinear Diffie-Hellman problem for a group generator G.

To see this, observe that the original proof of security for Waters' scheme [98, 22] also goes through for the extended scheme, using the elements g, g^α, g^β, T from the EDBDH problem to run the simulation as in the original proof and using the additional elements from the EDBDH problem to set up the master public key in the extended scheme.

We give evidence for the validity of the 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 [31]. In their notation, we have $P = \{1, \alpha, \alpha^2, \ldots, \alpha^q, \beta, \alpha^2\beta, \ldots, \alpha^q\beta, \gamma\}$, $Q = 1$, and $f = \alpha\beta\gamma$. It is clear by inspection that P, Q and f meet the independence requirement of the master theorem, and it then gives a lower bound on an adversary's advantage of solving the 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 $\Phi^{\text{poly}(d)}$-key malleable with algorithm T as follows:

```
$T(pp, mpk, u, usk', \phi_{a_0,a_1,\ldots,a_d})$:
If $(a_1 = 0)$ then $r \leftarrow\$ \mathbb{Z}_p$ ; $usk_1 \leftarrow g_1^{a_0} \cdot \mathsf{H}(u)^r \cdot ({g_1}^{msk^2})^{a_2} \cdots ({g_1}^{msk^d})^{a_d}$ ; $usk_2 \leftarrow g^r$
Else $usk_1 \leftarrow g_1^{a_0} \cdot {usk'_1}^{a_1} \cdot ({g_1}^{msk^2})^{a_2} \cdots ({g_1}^{msk^d})^{a_d}$ ; $usk_2 \leftarrow {usk'_2}^{a_1}$
Return $(usk_1, usk_2)$
```

The identity renaming scheme is then defined via

$$\mathrm{SI}(msk, \overline{u}) = \overline{u} || g^{msk}$$

and

$$\mathrm{PI}(mpk, \overline{u}, \phi_{a_0, a_1, \ldots, a_d}) = \overline{u} || g^{a_0} \cdot mpk^{a_1} \cdot (g^{msk^2})^{a_2} \cdots (g^{msk^d})^{a_d}$$

which clearly meets the compatibility and collision-resistance requirements.

Figure 4.7 shows IBE scheme poly-Wat, the extended Waters IBE scheme transformed to be secure against polynomial related-key attacks for polynomials of degree d. Combining Theorem 4.4.1 with Theorem 4.5.3 gives the following theorem.

Theorem 4.5.4 *Assume the EDBDH problem is hard for* G. *Then* poly-Wat *is* $\Phi^{\mathrm{poly}(q(\lambda))}$*-RKA secure. More concretely, let* $\overline{A}$ *be a* $\Phi^{\mathrm{poly}(q(\lambda))}$*-RKA adversary against* poly-Wat *making* $q_k(\lambda)$ *key derivation queries. Then there is an algorithm* A *solving the Extended Decision Bilinear Diffie-Hellman problem such that*

$$\mathbf{Adv}^{\text{ind-rka}}_{\mathsf{poly\text{-}Wat}, \Phi^{\mathrm{poly}(q(\lambda))}, \overline{A}}(\lambda) \leq 32(n+1) \cdot q_k(\lambda) \cdot \mathbf{Adv}^{\mathrm{edbdh}}_{\mathsf{G}, q(\lambda), A}(\lambda) .$$

4.6 Conclusion

We provided a framework enabling the construction of RKA-secure IBE schemes, and showed how specific instantiations of the framework yield the first IBE schemes secure against adversaries deriving new keys through affine and polynomial transformations of the master secret key. Our schemes are secure under the same assumptions as the underlying IBE schemes, and modify them in a very small and local way limited only to the way identities are hashed.

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 [14] to be an enabling primitive in the RKA domain: achieving RKA-secure IBE for any class Φ immediately yields Φ-RKA-secure PKE (CCA-secure public-key encryption) and signature schemes. These results were obtained by noting that the CHK [32] IBE-to-PKE transform and the Naor IBE-to-signatures transform both preserve RKA security. Thus, results for IBE immediately have wider impact.

An open problem here is to extend the framework to apply to the IBE scheme

```
poly-Wat.Pg($1^\lambda$):
$(p, \mathbb{G}, e) \leftarrow_\$ \mathsf{G.Pg}(1^\lambda)$
$g, g_1, h_0, \ldots, h_n \leftarrow_\$ \mathbb{G}^*$
Return $(p, \mathbb{G}, e, g, g_1, h_0, \ldots, h_n)$

poly-Wat.MSKSp($pp$):
$(p, \mathbb{G}, e, g, g_1, h_0, \ldots, h_n) \leftarrow pp$
Return $\mathbb{Z}_p^*$

poly-Wat.MPK($pp, msk$):
$(p, \mathbb{G}, e, g, g_1, h_0, \ldots, h_n) \leftarrow pp$
$mpk_0 \leftarrow g^{msk}$
$mpk \leftarrow (mpk_0, g^{msk^2}, \ldots, g^{msk^d}, g_1^{msk^2}, \ldots, g_1^{msk^d})$
Return $mpk$

poly-Wat.$\overline{\mathsf{UKg}}$($pp, msk, \overline{u}$):
$(p, \mathbb{G}, e, g, g_1, h_0, \ldots, h_n) \leftarrow pp$
$r \leftarrow_\$ \mathbb{Z}_p$
$usk_1 \leftarrow g_1^{msk} \cdot \mathsf{H}(\overline{u} || g^{msk})^r$
$usk_2 \leftarrow g^r$
Return $(usk_1, usk_2)$

poly-Wat.$\overline{\mathsf{Enc}}$($pp, mpk, \overline{u}, m$):
$(p, \mathbb{G}, e, g, g_1, h_0, \ldots, h_n) \leftarrow pp$
$(mpk_0, g^{msk^2}, \ldots, g^{msk^d}, g_1^{msk^2}, \ldots, g_1^{msk^d}) \leftarrow mpk$
$t \leftarrow_\$ \mathbb{Z}_p$
$c_1 \leftarrow g^t$
$c_2 \leftarrow \mathsf{H}(\overline{u} || mpk_0)^t$
$c_3 \leftarrow e(mpk_0, g_1)^t \cdot m$
Return $(c_1, c_2, c_3)$

poly-Wat.Dec($pp, usk, c$):
$(p, \mathbb{G}, e, g, g_1, h_0, \ldots, h_n) \leftarrow pp$
$(c_1, c_2, c_3) \leftarrow c$
$m \leftarrow c_3 \cdot \frac{e(usk_2, c_2)}{e(usk_1, c_1)}$
Return $m$
```

Figure 4.7: $\Phi^{\text{poly}(d)}$-RKA-secure Waters IBE scheme poly-Wat.

of Gentry [65], which fails to meet our notion of key-malleability, where a user-level key for user u derived from master secret key *msk* can be transformed into one derived from $\phi(msk)$. In the affine case, Gentry's IBE scheme does however have the property that a user-level key for user u derived from $\phi(msk)$ can be computed from a user-level key derived from master secret key *msk* for a *related* user $u' \leftarrow \phi^{-1}(u)$. This transformation is possible because the user space and identity space are the same, yet this same property causes issues in defining a suitable identity renaming transform, as we can no longer simply extend the message space and must instead hash into it. To achieve security, the identity renaming scheme must now have a computational rather than statistical collision-resistance property, where additionally the adversary has access to a key derivation oracle. Constructing a suitable hash function to satisfy this unnatural notion is a difficult task.

Another open problem is constructing a similar framework for constructing RKA-secure versions of selectively-secure IBE schemes such as those of Boneh and Boyen [29]. This is an interesting problem as selective security is sufficient to obtain CCA-secure PKE through the CHK transform [32], which preserves RKA security [14], and it may be possible to build schemes meeting this notion that are more efficient than ours, which meet the stronger notion of adaptive security. Since our identity renaming technique involves embedding the public key into the identity space, we cannot choose the challenge identity up front as is required in the selective security game, as we do not know at that point the public key to embed into it.

In the next chapter, we will use the RKA-secure IBE schemes of this chapter, and the techniques used in their construction, to build more RKA-secure primitives.

Chapter 5

Further RKA-Secure Primitives

Contents

In this chapter, we use the RKA-secure IBE schemes of the previous chapter, and the techniques used in their construction, to build further RKA-secure primitives. We construct RKA-secure PKE from IBE through the Boneh-Katz transform, and through applying the techniques of the previous chapter to build an RKA-secure KEM. We additionally build RKA-secure joint encryption and signature from IBE. When the base IBE scheme has a further malleability property, the PKE scheme obtained through the CHK transform can be converted into an RKA-secure CCA-SE (CCA-secure symmetric encryption) scheme. The results of this chapter combined with those of the previous chapter yield the first RKA-secure schemes for the primitives signature, PKE, and CCA-SE for non-linear RKAs, meaning beyond Φ^{lin}.

Primitive	Linear	Affine	Polynomial
IBE	[14]+[13]	✓	✓
Signatures	[14]+[13]	✓	✓
PKE	[99], [14]+[13]	✓	✓
CPA-SE	[8], [14]+[13]	[69]	[69]
CCA-SE	[14]+[13]	✓	$✓^*$
PRF	[13]	–	–

Figure 5.1: 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 and the previous chapter 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 [14]; non-adaptively secure signature schemes for non-linear RKAs were provided in [69]. 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 $\Phi^{\text{poly}(d)} \setminus \Phi^c$.

5.1 Introduction

The theoretical foundations of RKA security were laid by Bellare and Kohno [18], who treated the case of PRFs and PRPs. Research then expanded to consider other primitives [66, 8, 69, 14]. In particular, Bellare, Cash and Miller [14] provide a comprehensive treatment including strong definitions for many primitives and ways to transfer Φ-RKA security from one primitive to another.

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 [13] were able to present DDH-based PRFs secure against linear RKAs ($\Phi = \Phi^{\text{lin}}$). However, it is not clear how to take their techniques further to handle larger RKA sets Φ.

Figure 5.1 summarises 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), PKE (CCA secure public-key encryption[1]) , signatures, and IBE (CPA secure 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 in this and the previous chapter, 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 [69].

In more detail, Bellare, Cash and Miller [14] show how to transfer RKA security from PRF to any other primitive, assuming an existing standard-secure instance of the primitive. Combining this with [13] yields DDH-based schemes secure against linear RKAs for all the primitives, indicated by a "[14]+[13]" table entry. Applebaum, Harnik and Ishai [8] present LPN and LWE-based CPA-SE schemes secure against linear RKAs. Wee [99] presents PKE schemes secure against linear RKAs. Goyal, O'Neill and Rao [69] 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 [99] (based on a communication of Wichs) remarks that AMD codes [48] may be used to achieve RKA security for PKE, a method that extends to other primitives including IBE (but not PRFs), but with current constructions of these codes [48], the results continue to be restricted to linear RKAs. We note that we are interested in the stronger, adaptive definitions of RKA security as given in [14], but non-adaptively secure signature schemes for non-linear RKAs were provided in [69].

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 reach these goals with IBE as a starting point.

RKA-secure PKE and Signatures. Applying the results of [14] to the IBE schemes of the previous chapter, we obtain the first constructions of RKA-secure PKE and signature schemes for Φ^{aff} and $\Phi^{\mathrm{poly}(d)}$. Again, our schemes are efficient and our results hold in the standard model under reasonable hardness assumptions. The PKE schemes, being derived via the CHK transform [32], 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 [14], we show that the more efficient MAC-based Boneh-

[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 secret key.

Katz transform of [37, 32] 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 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 the DBDH assumption, 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 [69].

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 KEM that is inspired by our IBE framework and is based on the scheme of Boyen, Mei and Waters [39]. Our 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.

Note that RKA-secure PRFs for sets Φ^{aff} and $\Phi^{\mathrm{poly}(d)}$ cannot exist, since these sets contain constant functions, and we know that no PRF can be RKA-secure in this case [18]. Thus we are able to show stronger results for IBE, PKE and signatures than are possible for PRF. Also, although Bellare, Cash and Miller [14] showed that Φ-RKA security for PRF implies Φ-RKA security for signatures and PKE, the observation just made means we cannot use this result to get Φ^{aff} or $\Phi^{\mathrm{poly}(d)}$ RKA-secure IBE, PKE or signature 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 a combination of results from [14] and [13], our schemes are significantly more efficient.

Joint RKA Security. As a combination of the results of [14] and Chapter 3, we provide definitions for RKA security in the joint security setting, where the same keypair is used for both encryption and signature functions, and show that a Φ-RKA-secure IBE scheme can be used to build a Φ-RKA-secure joint encryption and signature scheme. This construction can be instantiated using any of our specific IBE schemes, by which we obtain the first concrete joint encryption and secure schemes for the RKA setting.

```
main IND-RKA$^A_{\mathsf{PKE},\Phi}(\lambda)$
$b \leftarrow_\$ \{0,1\}$ ; $c^* \leftarrow \perp$
$pp \leftarrow_\$ \mathsf{PKE.Pg}(1^\lambda)$ ; $(sk, pk) \leftarrow_\$ \mathsf{PKE.Kg}(pp)$
$b' \leftarrow_\$ A^{\text{Dec,LR}}(pp, pk)$
Return $(b = b')$

proc Dec$(\phi, c)$
If $(\phi \notin \Phi)$ then Return $\perp$
$sk' \leftarrow \phi(sk)$
If $((sk' = sk) \wedge (c = c^*))$ then Return $\perp$
Return $m \leftarrow \mathsf{PKE.Dec}(pp, sk', c)$

proc LR$(m_0, m_1)$
If $(c^* \neq \perp)$ then Return $\perp$
If $(|m_0| \neq |m_1|)$ then Return $\perp$
$c^* \leftarrow_\$ \mathsf{PKE.Enc}(pp, pk, m_b)$
Return $c^*$
```

Figure 5.2: Game IND-RKA defining Φ-RKA security of public key encryption scheme PKE under chosen-ciphertext attack.

RKA-Secure Symmetric Encryption. The final contribution of this chapter 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 [14] where the challenge encryption is also subject to RKA. Applying the CHK transform gives a strong Φ-RKA-secure PKE scheme which can be converted into a Φ-RKA-secure CCA-SE scheme in the natural way.

5.2 RKA-Secure PKE

We recall the definition of Φ-RKA-security for PKE from [14]. We say a public-key encryption scheme PKE is *Φ-RKA indistinguishable under chosen ciphertext attack* or *Φ-RKA secure* if $\mathbf{Adv}^{\text{ind-rka}}_{\mathsf{PKE},\Phi,A}(\cdot)$ is negligible for all PT adversaries A, where $\mathbf{Adv}^{\text{ind-rka}}_{\mathsf{PKE},\Phi,A}(\lambda) = 2\Pr[\text{IND-RKA}^A_{\mathsf{PKE},\Phi}(\lambda)] - 1$ and game IND-RKA is in Figure 5.2.

```
BK[IBE, EC, MAC].Pg(1^λ) :
$pp_{\mathsf{IBE}} \leftarrow\$ \mathsf{IBE.Pg}(1^\lambda)$
$pp_{\mathsf{EC}} \leftarrow\$ \mathsf{EC.Pg}(1^\lambda)$
$pp_{\mathsf{MAC}} \leftarrow\$ \mathsf{MAC.Pg}(1^\lambda)$
Return $(pp_{\mathsf{IBE}}, pp_{\mathsf{EC}}, pp_{\mathsf{MAC}})$

BK[IBE, EC, MAC].Kg(pp) :
$(pp_{\mathsf{IBE}}, pp_{\mathsf{EC}}, pp_{\mathsf{MAC}}) \leftarrow pp$
$(msk, mpk) \leftarrow\$ \mathsf{IBE.MKg}(pp_{\mathsf{IBE}})$
Return $(msk, mpk)$

BK[IBE, EC, MAC].Enc(pp, pk, m) :
$(pp_{\mathsf{IBE}}, pp_{\mathsf{EC}}, pp_{\mathsf{MAC}}) \leftarrow pp$ ; $mpk \leftarrow pk$
$(K, com, dec) \leftarrow\$ \mathsf{EC.Enc}(pp_{\mathsf{EC}})$
$u \leftarrow com$
$c_{\mathsf{IBE}} \leftarrow\$ \mathsf{IBE.Enc}(pp_{\mathsf{IBE}}, mpk, u, m||dec)$
$\tau \leftarrow \mathsf{MAC.Gen}(pp_{\mathsf{MAC}}, K, c_{\mathsf{IBE}})$
Return $(com, c_{\mathsf{IBE}}, \tau)$

BK[IBE, EC, MAC].Dec(pp, sk, c) :
$(pp_{\mathsf{IBE}}, pp_{\mathsf{EC}}, pp_{\mathsf{MAC}}) \leftarrow pp$ ; $msk \leftarrow sk$
$(com, c_{\mathsf{IBE}}, \tau) \leftarrow c$
$u \leftarrow com$
$usk \leftarrow\$ \mathsf{IBE.UKg}(pp_{\mathsf{IBE}}, msk, u)$
$m' \leftarrow \mathsf{IBE.Dec}(pp_{\mathsf{IBE}}, usk, c_{\mathsf{IBE}})$
$m||dec \leftarrow m'$
$K \leftarrow \mathsf{EC.Dec}(pp_{\mathsf{EC}}, com, dec)$
If $(K = \perp)$
    then Return $\perp$
If $(!\mathsf{MAC.Verify}(pp_{\mathsf{MAC}}, K, c_{\mathsf{IBE}}, \tau))$
    then Return $\perp$
Return $m$
```

Figure 5.3: PKE scheme BK[IBE, EC, MAC].

5.2.1 RKA Security of the Boneh-Katz Transform

The Boneh-Katz transform [32] constructs a PKE scheme BK[IBE, EC, MAC] from a selectively secure IBE scheme, making use of an encapsulation scheme and a message authentication code.

The transform uses an encapsulation scheme encapsulating keys of the massage authentication code, having commitment and decommitment strings *com* and *dec* of length n, where n is the bit length of identities in IBE. The algorithms of the transform are given in Figure 5.3.

We show that the Boneh-Katz transform preserves RKA security, that is that the transform can be applied to a Φ-RKA-secure IBE scheme to obtain a Φ-RKA-secure 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 that of the journal version [32], itself an improvement on the proof in the original paper [37].

```
main IND-sID-RKA$^{A}_{\mathsf{IBE},\Phi}(\lambda)$
$b \leftarrow_{\$} \{0,1\}$ ; $c^* \leftarrow \perp$ ; $u^* \leftarrow \perp$
$pp \leftarrow_{\$} \mathsf{IBE.Pg}(1^\lambda)$
$(msk, mpk) \leftarrow_{\$} \mathsf{IBE.MKg}(pp)$
$b' \leftarrow_{\$} A^{\mathrm{sID,KD,LR}}(pp)$
Return $(b = b')$

proc sID$(u)$
If $(u^* \neq \perp)$ then Return $\perp$
$u^* \leftarrow u$
Return $mpk$

proc KD$(\phi, u)$
If $(u^* = \perp)$ then Return $\perp$
If $(\phi \notin \Phi)$ then Return $\perp$
$msk' \leftarrow \phi(msk)$
If $((msk' = msk) \wedge (u = u^*))$ then Return $\perp$
Return $\mathsf{IBE.UKg}(pp, msk', u)$

proc LR$(m_0, m_1)$
If $(u^* = \perp)$ then Return $\perp$
If $(c^* \neq \perp)$ then Return $\perp$
If $(|m_0| \neq |m_1|)$ then Return $\perp$
$c^* \leftarrow_{\$} \mathsf{IBE.Enc}(pp, mpk, u^*, m_b)$
Return $c^*$
```

Figure 5.4: Game IND-sID-RKA defining selective-ID Φ-RKA indistinguishability of identity-based encryption scheme IBE.

First we extend the definition of selective-ID security of an IBE scheme to the RKA setting. We say an identity-based encryption scheme IBE is *selective-ID Φ-RKA indistinguishable under chosen plaintext attack* or *selectively Φ-RKA secure* if $\mathbf{Adv}^{\text{ind-sid-rka}}_{\mathsf{IBE},\Phi,A}(\cdot)$ is negligible for all PT adversaries A, where $\mathbf{Adv}^{\text{ind-sid-rka}}_{\mathsf{PKE},A,\Phi}(\lambda) = 2\Pr[\text{IND-sID-RKA}^{A}_{\mathsf{IBE},\Phi}(\lambda)] - 1$ and game IND-sID-RKA is given in Figure 5.4.

Theorem 5.2.1 *Let* IBE *be a selectively Φ-RKA secure IBE scheme,* EC *be a hiding and binding encapsulation scheme, and* MAC *be a one-time strongly unforgeable message authentication code. Then* $\mathsf{BK}[\mathsf{IBE},\mathsf{EC},\mathsf{MAC}]$ *is a Φ-RKA-secure PKE scheme.*

Proof: Let A be an adversary against the Φ-RKA-security of $\mathsf{BK}[\mathsf{IBE},\mathsf{EC},\mathsf{MAC}]$. making $q(\lambda)$ decryption queries. We build PT adversaries A_1, A_2, A_3, A_4, A_5 such that

$$\begin{aligned}\mathbf{Adv}^{\text{ind-rka}}_{\mathsf{BK}[\mathsf{IBE},\mathsf{EC},\mathsf{MAC}],\Phi,A}(\lambda) \leq{} & 2\mathbf{Adv}^{\text{bind}}_{\mathsf{EC},A_1}(\lambda) + 2\mathbf{Adv}^{\text{ind-sid-rka}}_{\mathsf{IBE},\Phi,A_2}(\lambda) \\ & + 2\mathbf{Adv}^{\text{ind-sid-rka}}_{\mathsf{IBE},\Phi,A_3}(\lambda) + 2\mathbf{Adv}^{\text{hide}}_{\mathsf{EC},A_4}(\lambda) \\ & + 2q(\lambda)\mathbf{Adv}^{\text{ot-suf}}_{\mathsf{MAC},A_5}(\lambda)\ ,\end{aligned}$$

for all $\lambda \in \mathbb{N}$, from which the theorem follows.

The proof uses the sequence of games in Figure 5.5. Game G_0 is as IND-RKA except that the values (K^*, com^*, dec^*) are generated in advance 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 and a ciphertext of the form $(com^*, c_{\mathsf{IBE}}, \tau)$. Game G_2 is as G_1 with a modification to the way the IBE ciphertext component of the challenge ciphertext is generated. In game G_2 it is computed as $c^*_{\mathsf{IBE}} \leftarrow_{\$} \mathsf{IBE.Enc}(pp_{\mathsf{IBE}}, mpk, com^*, 0^{|m_0|}||0^n)$, where n is the length of a decommitment string in EC. Game G_3 is as game G_2, but the tag component of the challenge ciphertext is generated under a randomly chosen key K'.

We say that a decryption query $(\phi, (com, c_{\mathsf{IBE}}, \tau))$ is valid if the ciphertext's decryption under $\phi(msk)$ does not return $\perp$. We now define a set of events that will help us bound A's probability of success.

- Let valid_i be the event that A playing game G_i makes a valid decryption query of the form $(\phi, (com^*, c_{\mathsf{IBE}}, \tau))$ with $(com^*, c_{\mathsf{IBE}}, \tau) \neq c^*$.

```
main IND-RKA$^A_{\mathsf{BK[IBE,EC,MAC]},\Phi}(\lambda)$ / $\boxed{\mathrm{G}^A_0(\lambda) \ / \ \mathrm{G}^A_1(\lambda) \ / \ \mathrm{G}^A_2(\lambda) \ / \ \mathrm{G}^A_3(\lambda)}$
$b \leftarrow_{\$} \{0,1\}$ ; $c^* \leftarrow \perp$
$pp_{\mathsf{IBE}} \leftarrow_{\$} \mathsf{IBE.Pg}(1^\lambda)$ ; $pp_{\mathsf{EC}} \leftarrow_{\$} \mathsf{EC.Pg}(1^\lambda)$ ; $pp_{\mathsf{MAC}} \leftarrow_{\$} \mathsf{MAC.Pg}(1^\lambda)$
$(msk, mpk) \leftarrow_{\$} \mathsf{IBE.MKg}(pp_{\mathsf{IBE}})$
$\boxed{(K^*, com^*, dec^*) \leftarrow_{\$} \mathsf{EC.Enc}(pp_{\mathsf{EC}})}$
$b' \leftarrow_{\$} A^{\mathrm{Dec,LR}}((pp_{\mathsf{IBE}}, pp_{\mathsf{EC}}, pp_{\mathsf{MAC}}), mpk)$
Return $(b = b')$

proc Dec$(\phi, c)$  // IND-RKA$^A_{\mathsf{BK[IBE,EC,MAC]},\Phi}(\lambda)$ / $\mathrm{G}^A_0(\lambda)$ / $\boxed{\mathrm{G}^A_1(\lambda) \ / \ \mathrm{G}^A_2(\lambda) \ / \ \mathrm{G}^A_3(\lambda)}$
If $(\phi \notin \Phi)$ then Return $\perp$
$msk' \leftarrow \phi(msk)$
If $((msk' = msk) \wedge (c = c^*))$ then Return $\perp$
$(com, c_{\mathsf{IBE}}, \tau) \leftarrow c$
$\boxed{\text{If } (com = com^*) \text{ then Return } \perp}$
$u \leftarrow com$
$usk \leftarrow_{\$} \mathsf{IBE.UKg}(pp_{\mathsf{IBE}}, msk', u)$
$m' \leftarrow \mathsf{IBE.Dec}(pp_{\mathsf{IBE}}, usk, c_{\mathsf{IBE}})$
$m || dec \leftarrow m'$
$K \leftarrow \mathsf{EC.Dec}(pp_{\mathsf{EC}}, com, dec)$
If $(K = \perp)$ then Return $\perp$
If $(!\mathsf{MAC.Verify}(pp_{\mathsf{MAC}}, K, c_{\mathsf{IBE}}, \tau))$ then Return $\perp$
Return $m$

proc LR$(m_0, m_1)$  // $\boxed{\text{IND-RKA}^A_{\mathsf{BK[IBE,EC,MAC]},\Phi}(\lambda)}$ / $\mathrm{G}^A_0(\lambda)$ / $\mathrm{G}^A_1(\lambda)$ / $\boxed{\mathrm{G}^A_2(\lambda)}$
If $(c^* \neq \perp)$ then Return $\perp$
If $(|m_0| \neq |m_1|)$ then Return $\perp$
$\boxed{(K^*, com^*, dec^*) \leftarrow_{\$} \mathsf{EC.Enc}(pp_{\mathsf{EC}})}$
$u \leftarrow com^*$
$c^*_{\mathsf{IBE}} \leftarrow_{\$} \mathsf{IBE.Enc}(pp_{\mathsf{IBE}}, mpk, u, m_b || dec)$
$\boxed{c^*_{\mathsf{IBE}} \leftarrow_{\$} \mathsf{IBE.Enc}(pp_{\mathsf{IBE}}, mpk, u, 0^{|m_0|} || 0^n)}$
$\tau^* \leftarrow \mathsf{MAC.Gen}(pp_{\mathsf{MAC}}, K^*, c^*_{\mathsf{IBE}})$
$c^* \leftarrow (com^*, c^*_{\mathsf{IBE}}, \tau^*)$
Return $c^*$

proc LR$(m_0, m_1)$  // $\mathrm{G}^A_3(\lambda)$
If $(c^* \neq \perp)$ then Return $\perp$
If $(|m_0| \neq |m_1|)$ then Return $\perp$
$u \leftarrow com^*$
$c^*_{\mathsf{IBE}} \leftarrow_{\$} \mathsf{IBE.Enc}(pp_{\mathsf{IBE}}, mpk, u, 0^{|m_0|} || 0^n)$
$K' \leftarrow_{\$} \mathsf{MAC.KSp}(pp_{\mathsf{MAC}})$
$\tau^* \leftarrow \mathsf{MAC.Gen}(pp_{\mathsf{MAC}}, K', c^*_{\mathsf{IBE}})$
$c^* \leftarrow (com^*, c^*_{\mathsf{IBE}}, \tau^*)$
Return $c^*$
```

Figure 5.5: Games used in the proof of Theorem 5.2.1.

- Let nobind_i be the event that A playing G_i makes a valid decryption query of the form $(\phi, (com^*, c_{\mathsf{IBE}}, \tau))$ with $(com^*, c_{\mathsf{IBE}}, \tau) \neq c^*$, such that c_{IBE} decrypts under $usk \leftarrow_{\$} \mathsf{UKg}(pp_{\mathsf{IBE}}, \phi(msk), com^*)$ to $m||dec$, and $\mathsf{EC.Dec}(pp_{\mathsf{EC}}, com^*, dec) = K$ with $K \notin \{K^*, \perp\}$.

- Let forge_i, for $i \in \{0,1,2\}$, be the event that A playing game G_i makes a valid decryption query of the form $(\phi, (com^*, c_{\mathsf{IBE}}, \tau))$ with $(com^*, c_{\mathsf{IBE}}, \tau) \neq c^*$ such that $\mathsf{MAC.Verify}(K^*, c_{\mathsf{IBE}}, \tau) = 1$.

- Let forge_3 denote the event that A playing game G_3 makes a valid decryption query of the form $(\phi, (com^*, c_{\mathsf{IBE}}, \tau))$ with $(com^*, c_{\mathsf{IBE}}, \tau) \neq c^*$ such that $\mathsf{MAC.Verify}(pp_{\mathsf{MAC}}, K', c_{\mathsf{IBE}}, \tau) = 1$.

We will build A_1, A_2, A_3, A_4, A_5 so that for all $\lambda \in \mathbb{N}$ we have

$$\Pr[\mathsf{nobind}_1] = \mathbf{Adv}^{\mathrm{bind}}_{\mathsf{EC},A_1}(\lambda) \tag{5.1}$$
$$\Pr[\mathrm{G}_1^A(\lambda)] - \Pr[\mathrm{G}_2^A(\lambda)] = \mathbf{Adv}^{\mathrm{ind\text{-}sid\text{-}rka}}_{\mathsf{IBE},\Phi,A_2}(\lambda) \tag{5.2}$$
$$\Pr[\mathsf{forge}_1] - \Pr[\mathsf{forge}_2] = \mathbf{Adv}^{\mathrm{ind\text{-}sid\text{-}rka}}_{\mathsf{IBE},\Phi,A_3}(\lambda) \tag{5.3}$$
$$\Pr[\mathsf{forge}_2] - \Pr[\mathsf{forge}_3] = \mathbf{Adv}^{\mathrm{hide}}_{\mathsf{EC},A_4}(\lambda) \tag{5.4}$$
$$\Pr[\mathsf{forge}_3]/q(\lambda) = \mathbf{Adv}^{\mathrm{ot\text{-}suf}}_{\mathsf{MAC},A_5}(\lambda)\ . \tag{5.5}$$

Since $\mathrm{IND\text{-}RKA}_{\mathsf{BK[IBE,EC,MAC]},\Phi}$ and G_0 are identical from A's point of view we have that

$$\Pr[\mathrm{IND\text{-}RKA}^A_{\mathsf{BK[IBE,EC,MAC]},\Phi}(\lambda)] = \Pr[\mathrm{G}_0^A(\lambda)]\ . \tag{5.6}$$

Games G_0 and G_1 are identical unless A makes a decryption query for which the response is $\perp$ in G_1 but not in G_0. This is exactly the event valid_0 (or valid_1) so the Fundamental Lemma of Game-Playing [23] we have that

$$\Pr[\mathrm{G}_0^A(\lambda)] - \Pr[\mathrm{G}_1^A(\lambda)] \leq \Pr[\mathsf{valid}_0] = \Pr[\mathsf{valid}_1]\ . \tag{5.7}$$

When valid_1 occurs, a ciphertext $(com^*, c_{\mathsf{IBE}}, \tau)$ was submitted such that the decapsulated key $K \neq \perp$, and the message authentication tag verifies. So either the decapsulated key $K = K^*$ and the event forge_1 occurred, or the decapsulated key $K \neq \{K^*, \perp\}$, in which case the event nobind_1 occurred. So we can bound valid_1 as

$$\Pr[\mathsf{valid}_1] \leq \Pr[\mathsf{nobind}_1] + \Pr[\mathsf{forge}_1]\ . \tag{5.8}$$

Since in game G_2 the challenge ciphertext is independent of b, we have that

$$\Pr[\mathrm{G}_2^A(\lambda)] = \frac{1}{2} \ . \tag{5.9}$$

By Equations (5.1)-(5.9), for all $\lambda \in \mathbb{N}$ we have

$$\begin{aligned}
\mathbf{Adv}^{\text{ind-rka}}_{\mathsf{BK[IBE,EC,MAC]},\Phi,A}(\lambda) &= 2\Pr[\text{IND-RKA}^A_{\mathsf{BK[IBE,EC,MAC]},\Phi}(\lambda)] - 1 \\
&= 2(\Pr[\mathrm{G}_0^A(\lambda)] - \Pr[\mathrm{G}_1^A(\lambda)]) + 2(\Pr[\mathrm{G}_1^A(\lambda)] - \Pr[\mathrm{G}_2^A(\lambda)]) \\
&\quad + 2\Pr[\mathrm{G}_2^A(\lambda)] - 1 \\
&\le 2\Pr[\mathsf{valid}_1] + 2\mathbf{Adv}^{\text{ind-sid-rka}}_{\mathsf{IBE},\Phi,A_2}(\lambda) + 2(\frac{1}{2}) - 1 \\
&\le 2(\Pr[\mathsf{nobind}_1] + \Pr[\mathsf{forge}_1]) + 2\mathbf{Adv}^{\text{ind-sid-rka}}_{\mathsf{IBE},\Phi,A_2}(\lambda) \\
&\le 2\Pr[\mathsf{nobind}_1] + 2(\Pr[\mathsf{forge}_1] - \Pr[\mathsf{forge}_2]) \\
&\quad + 2(\Pr[\mathsf{forge}_2] - \Pr[\mathsf{forge}_3]) + 2\Pr[\mathsf{forge}_3] \\
&\qquad + 2\mathbf{Adv}^{\text{ind-sid-rka}}_{\mathsf{IBE},\Phi,A_2}(\lambda) \\
&\le 2\mathbf{Adv}^{\text{bind}}_{\mathsf{EC},A_1}(\lambda) + 2\mathbf{Adv}^{\text{ind-sid-rka}}_{\mathsf{IBE},\Phi,A_3}(\lambda) + 2\mathbf{Adv}^{\text{hide}}_{\mathsf{EC},A_4}(\lambda) \\
&\quad + 2q(\lambda)\mathbf{Adv}^{\text{ot-suf}}_{\mathsf{MAC},A_5}(\lambda) + 2\mathbf{Adv}^{\text{ind-sid-rka}}_{\mathsf{IBE},\Phi,A_2}(\lambda)
\end{aligned}$$

as desired. We proceed to the constructions of A_1, A_2, A_3, A_4, A_5. Adversary A_1 against the binding property of EC behaves as follows:

```
$A_1(pp_{\mathsf{EC}}, com^*, dec^*)$
$K^* \leftarrow \mathsf{EC.Dec}(pp_{\mathsf{EC}}, com^*, dec^*)$
$dec' \leftarrow \perp$
$pp_{\mathsf{IBE}} \leftarrow_\$ \mathsf{IBE.Pg}(1^\lambda)$
$pp_{\mathsf{MAC}} \leftarrow_\$ \mathsf{MAC.Pg}(1^\lambda)$
$(msk, mpk) \leftarrow_\$ \mathsf{IBE.MKg}(pp_{\mathsf{IBE}})$
$b \leftarrow_\$ \{0,1\}$ ; $c^* \leftarrow \perp$
$b' \leftarrow_\$ A^{\mathrm{DEC},\mathrm{LR}}((pp_{\mathsf{IBE}}, pp_{\mathsf{EC}}, pp_{\mathsf{MAC}}), mpk)$
Return $(dec')$

$\mathrm{LR}(m_0, m_1)$
If $(c^* \ne \perp)$ then Return $\perp$
If $(|m_0| \ne |m_1|)$ then Return $\perp$
$u \leftarrow_\$ com^*$
$c^*_{\mathsf{IBE}} \leftarrow_\$ \mathsf{IBE.Enc}(pp_{\mathsf{IBE}}, mpk, u, m_b||dec)$
$\tau^* \leftarrow \mathsf{MAC.Gen}(pp_{\mathsf{MAC}}, K^*, c^*_{\mathsf{IBE}})$
$c^* \leftarrow (com^*, c^*_{\mathsf{IBE}}, \tau^*)$
Return $c^*$
```

```
$\mathrm{DEC}(\phi, c)$
If $(\phi \notin \Phi)$ then Return $\perp$
$msk' \leftarrow \phi(msk)$
If $((msk' = msk) \wedge (c = c^*))$
    then Return $\perp$
$(com, c_{\mathsf{IBE}}, \tau) \leftarrow c$
$u \leftarrow com$
$usk \leftarrow_\$ \mathsf{IBE.UKg}(pp_{\mathsf{IBE}}, msk', u)$
$m' \leftarrow \mathsf{IBE.Dec}(pp_{\mathsf{IBE}}, usk, c_{\mathsf{IBE}})$
$m||dec \leftarrow m'$
$K \leftarrow \mathsf{EC.Dec}(pp_{\mathsf{EC}}, com, dec)$
If $(K = \perp)$ then Return $\perp$
If $((com = com^*) \wedge (K \ne K^*))$
    then $dec' \leftarrow dec$
If $(com = com^*)$ then Return $\perp$
If $(!\mathsf{MAC.Verify}(pp_{\mathsf{MAC}}, K, c_{\mathsf{IBE}}, \tau))$
    then Return $\perp$
Return $m$
```

A_1 uses its knowledge of *msk* to respond to A's decryption queries as specified by game G_1, checking for the event nobind_1 and, should it occur, recording the decommitment string dec'. Adversary A_1 is successful exactly when nobind_1 occurs, establishing Equation (5.1).

Adversary A_2 against the selective-ID Φ-RKA security of IBE behaves as follows:

```
$A_2^{\mathrm{sID},\textsc{Dec},\mathrm{LR}}(pp_{\mathsf{IBE}})$
$b \leftarrow_{\$} \{0,1\}, c^* \leftarrow \perp$
$pp_{\mathsf{EC}} \leftarrow_{\$} \mathsf{EC.Pg}(1^\lambda)$
$pp_{\mathsf{MAC}} \leftarrow_{\$} \mathsf{MAC.Pg}(1^\lambda)$
$(K^*, com^*, dec^*) \leftarrow_{\$} \mathsf{EC.Enc}(pp_{\mathsf{EC}})$
$mpk \leftarrow \mathrm{sID}(com^*)$
$b' \leftarrow_{\$} A^{\textsc{DecSim},\textsc{LRSim}}((pp_{\mathsf{IBE}}, pp_{\mathsf{EC}}, pp_{\mathsf{MAC}}), mpk)$
If $(b' = b)$ then Return 1
Return 0

$\textsc{LRSim}(m_0, m_1)$
If $(c^* \neq \perp)$ then Return $\perp$
If $(|m_0| \neq |m_1|)$ then Return $\perp$
$c^*_{\mathsf{IBE}} \leftarrow \mathrm{LR}(0^{|m_0|}||0^n, m_b||dec^*)$
$\tau^* \leftarrow \mathsf{MAC.Gen}(pp_{\mathsf{MAC}}, K^*, c^*_{\mathsf{IBE}})$
$c^* \leftarrow (com^*, c^*_{\mathsf{IBE}}, \tau^*)$
Return $c^*$
```

```
$\textsc{DecSim}(\phi, c)$
If $(\phi \notin \Phi)$ then Return $\perp$
$(com, c_{\mathsf{IBE}}, \tau) \leftarrow c$
If $(com = com^*)$ then Return $\perp$
$u \leftarrow com$
$usk \leftarrow \mathrm{KD}(\phi, u)$
$m' \leftarrow \mathsf{IBE.Dec}(pp_{\mathsf{IBE}}, usk, c_{\mathsf{IBE}})$
$m||dec \leftarrow m'$
$K \leftarrow \mathsf{EC.Dec}(pp_{\mathsf{EC}}, com, dec)$
If $(K = \perp)$ then Return $\perp$
If $(!\mathsf{MAC.Verify}(pp_{\mathsf{MAC}}, K, c_{\mathsf{IBE}}, \tau))$
    then Return $\perp$
Return $m$
```

When the hidden bit in A_2's game is 0 then c^*_{IBE} is an encryption of $0^{m_0}||0^n$, so A's view is that of game G_2, while when the hidden bit in A_2's game is 1 then c^*_{IBE} is an encryption of $m_b||dec^*$, so A's view is that of game G_1. Since $\textsc{DecSim}$ returns $\perp$ on queries where $com = com^*$, the A_2 never calls its KD oracle on the challenge identity. We then have that

$$\begin{aligned}\mathbf{Adv}^{\text{ind-sid-rka}}_{\mathsf{IBE},\Phi,A_2}(\lambda) &= 2 \cdot \Pr[\text{IND-sID-RKA}^{A_2}_{\mathsf{IBE},\Phi}(\lambda)] - 1 \\ &= 2 \cdot \left(\tfrac{1}{2}\Pr[\overline{\mathrm{G}^A_2(\lambda)}] + \tfrac{1}{2}\Pr[\mathrm{G}^A_1(\lambda)]\right) - 1 \\ &= \Pr[\mathrm{G}^A_1(\lambda)] - \Pr[\mathrm{G}^A_2(\lambda)] \ ,\end{aligned}$$

as required.

Adversary A_3 against the selective-ID Φ-RKA security of IBE behaves as follows:

```
$\underline{A_3^{\mathrm{sID,DEC,LR}}(pp_{\mathsf{IBE}})}$
$b \leftarrow_\$ \{0,1\}$ ; $c^* \leftarrow \perp$ ; $\mathsf{forge} \leftarrow \mathsf{false}$
$pp_{\mathsf{EC}} \leftarrow_\$ \mathsf{EC.Pg}(1^\lambda)$
$pp_{\mathsf{MAC}} \leftarrow_\$ \mathsf{MAC.Pg}(1^\lambda)$
$pp \leftarrow (pp_{\mathsf{IBE}}, pp_{\mathsf{EC}}, pp_{\mathsf{MAC}})$
$(K^*, com^*, dec^*) \leftarrow_\$ \mathsf{EC.Enc}(pp_{\mathsf{EC}})$
$mpk \leftarrow \mathrm{sID}(com^*)$
$b' \leftarrow_\$ A^{\mathrm{DECSIM,LRSIM}}(pp, mpk)$
If ($\mathsf{forge}$) then Return 1
Return 0

$\underline{\mathrm{LRSIM}(m_0, m_1)}$
If $(c^* \neq \perp)$ then Return $\perp$
If $(|m_0| \neq |m_1|)$ then Return $\perp$
$c^*_{\mathsf{IBE}} \leftarrow \mathrm{LR}(0^{|m_0|}||0^n, m_b||dec^*)$
$\tau^* \leftarrow \mathsf{MAC.Gen}(pp_{\mathsf{MAC}}, K^*, c^*_{\mathsf{IBE}})$
$c^* \leftarrow (com^*, c^*_{\mathsf{IBE}}, \tau^*)$
Return $c^*$
```

```
$\underline{\mathrm{DECSIM}(\phi, c)}$
If $(\phi \notin \Phi)$ then Return $\perp$
$(com, c_{\mathsf{IBE}}, \tau) \leftarrow c$
If $(c = c^*)$ then Return $\perp$
If $((com = com^*) \wedge$
        $(\mathsf{MAC.Verify}(pp_{\mathsf{MAC}}, K^*, c_{\mathsf{IBE}}, \tau))$
    then $\mathsf{forge} \leftarrow \mathsf{true}$
If $(com = com^*)$ then Return $\perp$
$u \leftarrow com$
$usk \leftarrow \mathrm{KD}(\phi, u)$
$m' \leftarrow \mathsf{IBE.Dec}(pp_{\mathsf{IBE}}, usk, c_{\mathsf{IBE}})$
$m||dec \leftarrow m'$
$K \leftarrow \mathsf{EC.Dec}(pp_{\mathsf{EC}}, com, dec)$
If $(K = \perp)$ then Return $\perp$
If $(!\mathsf{MAC.Verify}(pp_{\mathsf{MAC}}, K, c_{\mathsf{IBE}}, \tau))$
    then Return $\perp$
Return $m$
```

Adversary A_3 behaves much as A_2, except it checks for the event forge and sets a flag if it occurs, returning 1 is the flag is set, and 0 otherwise. We then have that

$$\begin{aligned}\mathbf{Adv}^{\text{ind-sid-rka}}_{\mathsf{IBE},\Phi,A_3}(\lambda) &= 2 \cdot \Pr[\text{IND-sID-RKA}^{A_3}_{\mathsf{IBE},\Phi}(\lambda)] - 1 \\ &= 2 \cdot \left(\tfrac{1}{2}\Pr[\overline{\mathsf{forge}_2}] + \tfrac{1}{2}\Pr[\mathsf{forge}_1]\right) - 1 \\ &= \Pr[\mathsf{forge}_1] - \Pr[\mathsf{forge}_2] \ ,\end{aligned}$$

as required.

Adversary A_4 against the hiding property of EC behaves as follows:

```
$\underline{A_4(pp_{\mathsf{EC}}, com^*, \tilde{K})}$
$c^* \leftarrow \perp$ ; $\mathsf{forge} \leftarrow \mathsf{false}$
$pp_{\mathsf{IBE}} \leftarrow_\$ \mathsf{IBE.Pg}(1^\lambda)$
$pp_{\mathsf{MAC}} \leftarrow_\$ \mathsf{MAC.Pg}(1^\lambda)$
$pp \leftarrow (pp_{\mathsf{IBE}}, pp_{\mathsf{EC}}, pp_{\mathsf{MAC}})$
$(msk, mpk) \leftarrow_\$ \mathsf{IBE.MKg}(pp_{\mathsf{IBE}})$
$b' \leftarrow_\$ A^{\textsc{DecSim},\textsc{LRSim}}(pp, mpk)$
If $(\mathsf{forge})$ then Return 1
Return 0

$\underline{\textsc{LRSim}(m_0, m_1)}$
If $(c^* \neq \perp)$ then Return $\perp$
If $(|m_0| \neq |m_1|)$ then Return $\perp$
$c^*_{\mathsf{IBE}} \leftarrow_\$ \mathsf{IBE.Enc}(pp_{\mathsf{IBE}}, mpk, com^*, 0^{|m_0|}||0^n)$
$\tau^* \leftarrow \mathsf{MAC.Gen}(pp_{\mathsf{MAC}}, \tilde{K}, c^*_{\mathsf{IBE}})$
$c^* \leftarrow (com^*, c^*_{\mathsf{IBE}}, \tau^*)$
Return $c^*$
```

```
$\underline{\textsc{DecSim}(\phi, c)}$
If $(\phi \notin \Phi)$ then Return $\perp$
$msk' \leftarrow \phi(msk)$
$(com, c_{\mathsf{IBE}}, \tau) \leftarrow c$
If $(c = c^*)$ then Return $\perp$
If $((com = com^*) \wedge$
        $(\mathsf{MAC.Verify}(pp_{\mathsf{MAC}}, \tilde{K}, c_{\mathsf{IBE}}, \tau))$
    then $\mathsf{forge} \leftarrow \mathsf{true}$
If $(com = com^*)$ then Return $\perp$
$u \leftarrow com$
$usk \leftarrow_\$ \mathsf{IBE.UKg}(pp_{\mathsf{IBE}}, msk', u)$
$m' \leftarrow \mathsf{IBE.Dec}(pp_{\mathsf{IBE}}, usk, c_{\mathsf{IBE}})$
$m||dec \leftarrow m'$
$K \leftarrow \mathsf{EC.Dec}(pp_{\mathsf{EC}}, com, dec)$
If $(K = \perp)$ then Return $\perp$
If $(!\mathsf{MAC.Verify}(pp_{\mathsf{MAC}}, K, c_{\mathsf{IBE}}, \tau))$
    then Return $\perp$
Return $m$
```

If the hidden bit is 1 in the hiding game then $\tilde{K}$ is the value encapsulated by com^* and A's view is that of game G_2, so A_4 outputs 1 with probability forge_2, while if the hidden bit is 0 in the hiding game then $\tilde{K}$ 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

$$\begin{aligned}\mathbf{Adv}^{\mathrm{hide}}_{\mathsf{EC},A_4}(\lambda) &= 2 \cdot \Pr[\mathrm{HIDE}^{A_4}_{\mathsf{EC}}(\lambda)] - 1 \\ &= 2 \cdot \left(\tfrac{1}{2}\Pr[\overline{\mathsf{forge}_3}] + \tfrac{1}{2}\Pr[\mathsf{forge}_2]\right) - 1 \\ &= \Pr[\mathsf{forge}_2] - \Pr[\mathsf{forge}_3] \ .\end{aligned}$$

Adversary A_5 against the strong one-time unforgeability of MAC behaves as follows:

```
$\underline{A_5^{\mathrm{Gen}}(pp_{\mathsf{MAC}})}$
$c^* \leftarrow \perp$ ; $j \leftarrow_\$ \{1\ldots,q(\lambda)\}$ ; $i \leftarrow 0$
$(m_j,\tau_j) \leftarrow \perp$
$pp_{\mathsf{IBE}} \leftarrow_\$ \mathsf{IBE.Pg}(1^\lambda)$
$pp_{\mathsf{EC}} \leftarrow_\$ \mathsf{EC.Pg}(1^\lambda)$
$(msk, mpk) \leftarrow_\$ \mathsf{IBE.MKg}(pp_{\mathsf{IBE}})$
$(K^*, com^*, dec^*) \leftarrow_\$ \mathsf{EC.Enc}(pp_{\mathsf{EC}})$
$b' \leftarrow_\$ A^{\mathrm{DecSim},\mathrm{LRSim}}((pp_{\mathsf{IBE}}, pp_{\mathsf{EC}}, pp_{\mathsf{MAC}}), mpk)$
Return $(m_j,\tau_j)$

$\underline{\mathrm{LRSim}(m_0, m_1)}$
If $(c^* \neq \perp)$ then Return $\perp$
If $(|m_0| \neq |m_1|)$ then Return $\perp$
$c^*_{\mathsf{IBE}} \leftarrow_\$ \mathsf{IBE.Enc}(pp_{\mathsf{IBE}}, mpk, com^*, 0^{|m_0|}||0^n)$
$\tau^* \leftarrow \mathrm{Gen}(c^*_{\mathsf{IBE}})$
$c^* \leftarrow (com^*, c^*_{\mathsf{IBE}}, \tau^*)$
Return $c^*$

$\underline{\mathrm{DecSim}(\phi, c)}$
$i \leftarrow i+1$
If $(\phi \notin \Phi)$ then Return $\perp$
$msk' \leftarrow \phi(msk)$
$(com, c_{\mathsf{IBE}}, \tau) \leftarrow c$
If $(i = j)$ then $(m_j,\tau_j) \leftarrow (c_{\mathsf{IBE}}, \tau)$
If $(com = com^*)$ then Return $\perp$
$u \leftarrow com$
$usk \leftarrow_\$ \mathsf{IBE.UKg}(pp_{\mathsf{IBE}}, msk', u)$
$m' \leftarrow \mathsf{IBE.Dec}(pp_{\mathsf{IBE}}, usk, c_{\mathsf{IBE}})$
$m||dec \leftarrow m'$
$K \leftarrow \mathsf{EC.Dec}(pp_{\mathsf{EC}}, com, dec)$
If $(K = \perp)$ then Return $\perp$
If $(!\mathsf{MAC.Verify}(pp_{\mathsf{MAC}}, K, c_{\mathsf{IBE}}, \tau))$
    then Return $\perp$
Return $m$
```

Since A_5 doesn't know the key K' in the MAC forgery game, it can't detect forge_3 occurring. It instead makes a guess that event will occur on A's jth decryption query for a randomly chosen j. If the event forge_3 does occur, that is if A made a decryption query of the form $(\phi, (com^*, c_{\mathsf{IBE}}, \tau))$ with $\mathsf{MAC.Verify}(pp_{\mathsf{MAC}}, K', c_{\mathsf{IBE}}, \tau) = 1$, then with probability $1/q(\lambda)$ it was on A's jth query, so the probability that A_5 makes a successful forgery is $\Pr[\mathsf{forge}_3]/q(\lambda)$, as required. ▮

We recall from [14] that, given a selectively Φ-RKA-secure IBE scheme, the CHK transform [32] yields a Φ-RKA-secure PKE scheme at the cost of adding a strongly unforgeable one-time secure signature and its verification key to the IBE ciphertexts. Above we showed that the more efficient Boneh-Katz transform [32] can also be used to the same effect. We omit the details of the Φ^{aff}-RKA-secure PKE schemes that result from applying these transforms to the aff-BF and aff-Wat IBE schemes. We simply note that the resulting PKE schemes are as efficient as the pairing-based schemes of Wee [99], which are only Φ^{lin}-RKA-secure. Similarly, using a result of [14], 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 [38] and Waters [98] 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 [38, 98].

As in the affine case, we may apply results of [14] or Theorem 5.2.1 to the poly-Wat

```
main IND-RKA$^A_{\mathsf{KEM},\Phi}(\lambda)$
$b \leftarrow_\$ \{0,1\}$ ; $c^* \leftarrow \perp$
$pp \leftarrow_\$ \mathsf{KEM.Pg}(1^\lambda)$ ; $(sk, pk) \leftarrow_\$ \mathsf{KEM.Kg}(pp)$
$(c^*, K^*) \leftarrow_\$ \mathsf{KEM.Enc}(pp, pk)$
If $(b = 0)$ then $K^* \leftarrow_\$ \mathsf{KSp}(pp)$
$b' \leftarrow_\$ A^{\text{Dec,LR}}(pp, c^*, K^*)$
Return $(b = b')$

proc Dec$(\phi, c)$
If $(\phi \notin \Phi)$ then Return $\perp$
$sk' \leftarrow \phi(sk)$
If $((sk' = sk) \wedge (c = c^*))$ then Return $\perp$
Return $K \leftarrow \mathsf{KEM.Dec}(pp, sk', c)$
```

Figure 5.6: Game IND-RKA defining Φ-RKA indistinguishability of key encapsulation mechanism KEM under chosen-ciphertext attack.

scheme to obtain a $\Phi^{\text{poly}(d)}$-RKA-secure PKE scheme and a $\Phi^{\text{poly}(d)}$-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 EDBDH assumption.

5.2.2 RKA Security for the KEM-DEM Paradigm

As noted in the previous section, the framework for constructing Φ-RKA-secure IBE schemes developed in the previous chapter can be combined with results from [14] to build Φ-RKA-secure PKE schemes. In this section, we give a more efficient, direct construction for a Φ-RKA-secure PKE scheme based on the KEM of Boyen, Mei and Waters [39]. Compared to the schemes of the previous section, our scheme enjoys shorter ciphertexts and smaller public parameters. We begin by providing an appropriate definition of RKA security for a KEM and show that the KEM-DEM paradigm of [49] extends to the RKA setting in a natural way. We then show how to modify the BMW scheme to build an efficient 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 [99].

We define Φ-RKA-security for a KEM through the game in Figure 5.6, which is the natural extension of IND-CCA security for a KEM to the RKA setting.

We say a key encapsulation mechanism KEM is Φ-*RKA indistinguishable under chosen ciphertext attack* or Φ-*RKA secure* if $\mathbf{Adv}^{\text{ind-rka}}_{\mathsf{KEM},\Phi,A}(\cdot)$ is negligible for all PT adversaries A, where $\mathbf{Adv}^{\text{ind-rka}}_{\mathsf{KEM},\Phi,A}(\lambda) = 2\Pr[\text{IND-RKA}^{A}_{\mathsf{KEM},\Phi}(\lambda)] - 1$ and game IND-RKA is in Figure 5.6.

We now show that the KEM-DEM composition theorem of [49] also holds in the RKA setting. Note that while the KEM must be RKA-secure, the DEM need only meet the standard one-time CCA security notion.

Theorem 5.2.2 *Let* KEM *be a* Φ*-RKA-secure KEM, and* DEM *be a OT-CCA-secure DEM. Then the hybrid encryption scheme* $\mathsf{HPKE}[\mathsf{KEM},\mathsf{DEM}]$ *in Figure 2.13 is* Φ*-RKA-secure.*

Proof: Let A be an adversary against the Φ-RKA-security of the PKE scheme $\mathsf{HPKE}[\mathsf{KEM},\mathsf{DEM}]$. We build PT adversaries A_1, A_2 such that

$$\mathbf{Adv}^{\text{ind-rka}}_{\mathsf{HPKE}[\mathsf{KEM},\mathsf{DEM}],\Phi,A}(\lambda) \le 2\mathbf{Adv}^{\text{ind-rka}}_{\mathsf{KEM},\Phi,A_1}(\lambda) + \mathbf{Adv}^{\text{ot-ind-cca}}_{\mathsf{DEM},A_2}(\lambda)\ ,$$

for all $\lambda \in \mathbb{N}$, from which the theorem follows.

The proof uses the sequence of games in Figure 5.7. Game G_0 is as game $\text{IND-RKA}_{\mathsf{HPKE}[\mathsf{KEM},\mathsf{DEM}],\Phi}$, except that in game G_0 the key K^* is encapsulated in advance, and if the adversary submits for decryption a ciphertext whose first component is c_1^* and a ϕ such that $\phi(sk) = sk$, the key K^* is used directly rather than obtaining this key through decapsulating c_1^* under key K. The behaviour of the games is identical, so we have that

$$\Pr[\text{IND-RKA}^{A}_{\mathsf{HPKE}[\mathsf{KEM},\mathsf{DEM}],\Phi}(\lambda)] = \Pr[\mathrm{G}_0^A(\lambda)]\ .$$

Game G_1 is as game G_0 except that the key K^* encapsulated by the first component of the challenge ciphertext is replaced by a random key K'.

We will build A_1, A_2 so that for all $\lambda \in \mathbb{N}$ we have

$$\Pr[\mathrm{G}_0^A(\lambda)] - \Pr[\mathrm{G}_1^A(\lambda)] \le \mathbf{Adv}^{\text{ind-rka}}_{\mathsf{KEM},\Phi,A_1}(\lambda) \qquad (5.10)$$

$$2\Pr[\mathrm{G}_1^A] - 1 = \mathbf{Adv}^{\text{ot-ind-cca}}_{\mathsf{DEM},A_2}(\lambda) \qquad (5.11)$$

```
main IND-RKA$^A_{\mathsf{HPKE[KEM,DEM]},\Phi}(\lambda)$ / $\boxed{G^A_0(\lambda)}$ / $G^A_1(\lambda)$
$b \leftarrow_\$ \{0,1\}$ ; $c^* \leftarrow \perp$
$pp_{\mathsf{KEM}} \leftarrow_\$ \mathsf{KEM.Pg}(1^\lambda)$
$pp_{\mathsf{DEM}} \leftarrow_\$ \mathsf{DEM.Pg}(1^\lambda)$
$(sk, pk) \leftarrow_\$ \mathsf{KEM.Kg}(pp_{\mathsf{KEM}})$
$\boxed{(c_1^*, K^*) \leftarrow_\$ \mathsf{KEM.Enc}(pp_{\mathsf{KEM}}, pk)}$
$(c_1^*, K^*) \leftarrow_\$ \mathsf{KEM.Enc}(pp_{\mathsf{KEM}}, pk)$ ; $K' \leftarrow_\$ \mathsf{KEM.KSp}(pp_{\mathsf{KEM}})$
$b' \leftarrow_\$ A^{\mathrm{Dec,LR}}((pp_{\mathsf{KEM}}, pp_{\mathsf{DEM}}), pk)$
Return $(b = b')$

proc Dec$(\phi, c)$ // $\boxed{G^A_0(\lambda)}$ / $G^A_1(\lambda)$
If $(\phi \notin \Phi)$ then Return $\perp$
$sk' \leftarrow \phi(sk)$
If $((sk' = sk) \wedge (c = c^*))$ then Return $\perp$
$\boxed{\text{If } ((sk' = sk) \wedge (c_1 = c_1^*)) \text{ then Return } \mathsf{DEM.Dec}(pp_{\mathsf{DEM}}, K^*, c_2)}$
If $((sk' = sk) \wedge (c_1 = c_1^*))$ then Return $\mathsf{DEM.Dec}(pp_{\mathsf{DEM}}, K', c_2)$
$K \leftarrow_\$ \mathsf{KEM.Dec}(pp_{\mathsf{KEM}}, sk', c_1)$
Return $\mathsf{DEM.Dec}(pp_{\mathsf{DEM}}, K, c_2)$

proc LR$(m_0, m_1)$ // IND-RKA$^A_{\mathsf{HPKE[KEM,DEM]},\Phi}(\lambda)$
If $(c^* \neq \perp)$ then Return $\perp$
If $(|m_0| \neq |m_1|)$ then Return $\perp$
$(c_1^*, K^*) \leftarrow_\$ \mathsf{KEM.Enc}(pp_{\mathsf{KEM}}, pk)$
$c_2^* \leftarrow_\$ \mathsf{DEM.Enc}(K^*, m_b)$
$c^* \leftarrow (c_1^*, c_2^*)$
Return $c^*$

proc LR$(m_0, m_1)$ // $\boxed{G^A_0(\lambda)}$ / $G^A_1(\lambda)$
If $(c^* \neq \perp)$ then Return $\perp$
If $(|m_0| \neq |m_1|)$ then Return $\perp$
$\boxed{c_2^* \leftarrow_\$ \mathsf{DEM.Enc}(K^*, m_b)}$
$c_2^* \leftarrow_\$ \mathsf{DEM.Enc}(K', m_b)$
$c^* \leftarrow (c_1^*, c_2^*)$
Return $c^*$
```

Figure 5.7: Games used in the proof of Theorem 5.2.2.

By the above equations, for all $\lambda \in \mathbb{N}$ we have

$$\begin{aligned}\mathbf{Adv}^{\text{ind-rka}}_{\mathsf{HPKE[KEM,DEM]},\Phi,A}(\lambda) &= 2\Pr[\text{IND-RKA}^{A}_{\mathsf{HPKE[KEM,DEM]},\Phi}(\lambda)] - 1\\ &= 2\Pr[\mathrm{G}_0^A(\lambda)] - 1\\ &= 2(\Pr[\mathrm{G}_0^A(\lambda)] - \Pr[\mathrm{G}_1^A(\lambda)]) + 2\Pr[\mathrm{G}_1^A(\lambda)] - 1\\ &\leq 2\mathbf{Adv}^{\text{ind-rka}}_{\mathsf{KEM},\Phi,A_1}(\lambda) + \mathbf{Adv}^{\text{ot-ind-cca}}_{\mathsf{DEM},A_2}(\lambda)\end{aligned}$$

as desired. We proceed to the constructions of A_1, A_2. Adversary A_1 against the Φ-RKA security of KEM behaves as follows:

```
$\underline{A_1^{\text{Dec}}(pp_{\mathsf{KEM}}, pk, c_1^*, K^*)}$
$b \leftarrow_{\$} \{0,1\}\ ;\ c^* \leftarrow \perp$
$pp_{\mathsf{DEM}} \leftarrow_{\$} \mathsf{DEM.Pg}(1^\lambda)$
$pp \leftarrow (pp_{\mathsf{KEM}}, pp_{\mathsf{DEM}})$
$b' \leftarrow_{\$} A^{\text{DecSim,LR}}(pp, pk)$
If $(b = b')$ then Return 0
Else Return 1

$\underline{\text{LR}(m_0, m_1)}$
If $(c^* \neq \perp)$ then Return $\perp$
If $(|m_0| \neq |m_1|)$ then Return $\perp$
$c_2^* \leftarrow_{\$} \mathsf{DEM.Enc}(K^*, m_b)$
$c^* \leftarrow (c_1^*, c_2^*)$
Return $c^*$
```

```
$\underline{\text{DecSim}(\phi, c)}$
If $(\phi \notin \Phi)$ then Return $\perp$
$sk' \leftarrow \phi(sk)$
If $(c_1 = c_1^*)$
  then $K \leftarrow \text{Dec}(\phi, c_1^*)$
  If $((K = \perp) \wedge (c_2 = c_2^*))$
    then Return $\perp$
  If $((K = \perp) \wedge (c_2 \neq c_2^*))$
    then Return $\mathsf{DEM.Dec}(pp_{\mathsf{DEM}}, K^*, c_2)$
  If $(K \neq \perp)$
    then Return $\mathsf{DEM.Dec}(pp_{\mathsf{DEM}}, K, c_2)$
$K \leftarrow \text{Dec}(\phi, c_1)$
Return $\mathsf{DEM.Dec}(pp_{\mathsf{DEM}}, K, c_2)$
```

Adversary A_1 must detect when a ciphertext (c_1^*, c_2) is submitted for decryption under a ϕ such that $\phi(sk) = sk$. It does this by submitting to its own decryption oracle (ϕ, c_1^*), and if the result is $\perp$ then $\phi(sk) = sk$. When the hidden bit in A_1's game is 1, the key K^* is the real key, so A is provided with a perfect simulation of game G_0, while when A_1's challenge bit is 0, K^* is a random key so A is playing game G_1 where K' is used in place of K^*. This gives us that

$$\begin{aligned}\mathbf{Adv}^{\text{ind-rka}}_{\mathsf{KEM},\Phi,A_1}(\lambda) &= 2\Pr[\text{IND-RKA}^{A_1}_{\mathsf{KEM},\Phi}(\lambda)] - 1\\ &= 2 \cdot \left(\tfrac{1}{2}\Pr[\overline{\mathrm{G}_0^A}(\lambda)] + \tfrac{1}{2}\Pr[\mathrm{G}_1^A(\lambda)]\right) - 1\\ &= \Pr[\mathrm{G}_1^A(\lambda)] - \Pr[\mathrm{G}_0^A(\lambda)]\ ,\end{aligned}$$

as required.

Adversary A_2 against the OT-CCA security of DEM behaves as follows:

```
$\underline{A_2^{\text{Dec,LR}}(pp_{\mathsf{DEM}})}$
$c^* \leftarrow \perp$
$pp_{\mathsf{KEM}} \leftarrow_\$ \mathsf{KEM.Pg}(1^\lambda)$
$(sk, pk) \leftarrow_\$ \mathsf{KEM.Kg}(pp_{\mathsf{KEM}})$
$(c_1^*, K^*) \leftarrow_\$ \mathsf{KEM.Enc}(pp_{\mathsf{KEM}}, pk)$
$b' \leftarrow_\$ A^{\text{DecSim,LRSim}}((pp_{\mathsf{KEM}}, pp_{\mathsf{DEM}}), pk)$
Return $b'$

$\underline{\text{LRSim}(m_0, m_1)}$
If $(c^* \neq \perp)$ then Return $\perp$
If $(|m_0| \neq |m_1|)$ then Return $\perp$
$c_2^* \leftarrow \text{LR}(m_0, m_1)$
$c^* \leftarrow (c_1^*, c_2^*)$
Return $c^*$
```

```
$\underline{\text{DecSim}(\phi, c)}$
If $(\phi \notin \Phi)$ then Return $\perp$
$sk' \leftarrow \phi(sk)$
If $((sk' = sk) \wedge (c = c^*))$
    then Return $\perp$
If $((sk' = sk) \wedge (c_1 = c_1^*))$
    then Return $\text{Dec}(c_2)$
$K \leftarrow \mathsf{KEM.Dec}(pp_{\mathsf{KEM}}, sk', c_1)$
Return $\mathsf{DEM.Dec}(pp_{\mathsf{DEM}}, K, c_2)$
```

A_2 provides a perfect simulation of G_1 for A and correctly determines the challenge bit with the same probability as A, so Equation (5.11) holds. ▌

5.2.3 An RKA-Secure KEM from the BMW Scheme

Boyen, Mei and Waters [39] build a very efficient KEM based on the first IBE scheme of Boneh and Boyen [30] and show the KEM is secure under the DBDH assumption in the setting of asymmetric pairings. The algorithms of their KEM BMW are shown on the left-hand side of Figure 5.8.

The KEM BMW from [39] 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 h_0 of the secret key, admitting the following attack. On receipt of the public key (Z, u_1, u_2), a key K^*, and a ciphertext c^* of the form $(g^t, u_1^t u_2^{tw})$, where $w = \mathsf{H.Eval}(pp_{\mathsf{H}}, g^t)$ and t is the randomness used to generate the challenge ciphertext, the adversary then makes a query $\text{Dec}(\phi_a, C^*)$ 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 1 if so and 0 if not, winning the Φ-RKA-security game.

The right-hand side of Figure 5.8 shows the algorithms of MBMW, an enhanced version of the KEM of [39] 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 $\mathbb{G}_1$

```
$\underline{\mathsf{BMW.Pg}(1^\lambda):}$
$(p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e) \leftarrow_\$ \mathsf{G.Pg}(1^\lambda)$
$g \leftarrow_\$ \mathbb{G}_1^* \,;\, h \leftarrow_\$ \mathbb{G}_2^*$
$pp_\mathsf{H} \leftarrow_\$ \mathsf{H.Pg}(1^\lambda)$
$pp \leftarrow (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e, g, h, pp_\mathsf{H})$
Return $pp$

$\underline{\mathsf{BMW.Kg}(pp):}$
$(p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e, g, h, pp_\mathsf{H}) \leftarrow pp$
$s \leftarrow_\$ \mathbb{Z}_p^*$
$h_0 \leftarrow h^s$
$Z \leftarrow e(g, h_0)$
$y_1, y_2 \leftarrow_\$ \mathbb{Z}_p^*$
$u_1 \leftarrow g^{y_1} \,;\, u_2 \leftarrow g^{y_2}$
$sk \leftarrow (h_0, y_1, y_2)$
$pk \leftarrow (Z, u_1, u_2)$
Return $(sk, pk)$

$\underline{\mathsf{BMW.Enc}(pp, pk):}$
$(p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e, g, h, pp_\mathsf{H}) \leftarrow pp$
$(Z, u_1, u_2) \leftarrow pk$
$t \leftarrow_\$ \mathbb{Z}_p^*$
$c_1 \leftarrow g^t$
$w \leftarrow \mathsf{H.Eval}(pp_\mathsf{H}, c_1)$
$c_2 \leftarrow u_1^t u_2^{tw}$
$K \leftarrow Z^t$
Return $((c_1, c_2), K)$

$\underline{\mathsf{BMW.Dec}(pp, sk, c):}$
$(p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e, g, h, pp_\mathsf{H}) \leftarrow pp$
$(h_0, y_1, y_2) \leftarrow sk$
$w \leftarrow \mathsf{H.Eval}(pp_\mathsf{H}, c_1)$
$w' \leftarrow y_1 + y_2 w \mod p$
If $(c_1^{w'} \neq c_2)$ then Return $\perp$
Return $e(c_1, h_0)$
```

```
$\underline{\mathsf{MBMW.Pg}(1^\lambda):}$
$(p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e) \leftarrow_\$ \mathsf{G.Pg}(1^\lambda)$
$g \leftarrow_\$ \mathbb{G}_1^* \,;\, h \leftarrow_\$ \mathbb{G}_2^*$
$y_1, y_2 \leftarrow_\$ \mathbb{Z}_p^*$
$u_1 \leftarrow h^{y_1} \,;\, u_2 \leftarrow h^{y_2}$
$pp_\mathsf{H} \leftarrow_\$ \mathsf{H.Pg}(1^\lambda)$
$pp \leftarrow (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e, g, h, u_1, u_2, pp_\mathsf{H})$
Return $pp$

$\underline{\mathsf{MBMW.Kg}(pp):}$
$(p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e, g, h, u_1, u_2, pp_\mathsf{H}) \leftarrow pp$
$sk \leftarrow_\$ \mathbb{Z}_p^*$
$pk \leftarrow e(g, h)^{sk}$
Return $(sk, pk)$

$\underline{\mathsf{MBMW.Enc}(pp, pk):}$
$(p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e, g, h, u_1, u_2, pp_\mathsf{H}) \leftarrow pp$
$t \leftarrow_\$ \mathbb{Z}_p^*$
$c_1 \leftarrow g^t$
$w \leftarrow \mathsf{H.Eval}(pp_\mathsf{H}, c_1 || pk)$
$c_2 \leftarrow u_1^t u_2^{tw}$
$K \leftarrow pk^t$
Return $((c_1, c_2), K)$

$\underline{\mathsf{MBMW.Dec}(pp, sk, c):}$
$(p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e, g, h, u_1, u_2, pp_\mathsf{H}) \leftarrow pp$
$w \leftarrow \mathsf{H.Eval}(pp_\mathsf{H}, c_1 || e(g, h)^{sk})$
If $(e(c_1, u_1 u_2^w) \neq e(g, c_2))$ then Return $\perp$
Return $e(c_1, h)^{sk}$
```

Figure 5.8: Left: Original Boyen-Mei-Waters KEM BMW. Right: RKA-secure variant MBMW.

```
main IND-RKA$^A_{\mathsf{MBMW},\Phi}(\lambda)$ / G$^A_0(\lambda)$
$b \leftarrow_\$ \{0,1\}$ ; $c^* \leftarrow \perp$
$(p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e) \leftarrow_\$ \mathsf{G.Pg}(1^\lambda)$
$g \leftarrow_\$ \mathbb{G}_1^*$ ; $h \leftarrow_\$ \mathbb{G}_2^*$
$y_1, y_2 \leftarrow_\$ \mathbb{Z}_p^*$
$u_1 \leftarrow h^{y_1}$ ; $u_2 \leftarrow h^{y_2}$
$pp_{\mathsf{H}} \leftarrow_\$ \mathsf{H.Pg}(1^\lambda)$
$pp \leftarrow (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e, g, h, u_1, u_2, pp_{\mathsf{H}})$
$sk \leftarrow_\$ \mathbb{Z}_p^*$
$pk \leftarrow e(g,h)^{sk}$
$t \leftarrow_\$ \mathbb{Z}_p^*$
$c_1^* \leftarrow g^t$
$w^* \leftarrow \mathsf{H.Eval}(pp_{\mathsf{H}}, c_1^* || pk)$
$c_2^* \leftarrow u_1^t u_2^{tw^*}$
$c^* \leftarrow (c_1^*, c_2^*)$
$K^* \leftarrow pk^t$
If $(b = 0)$ then $K^* \leftarrow_\$ \mathsf{KSp}(pp)$
$b' \leftarrow_\$ A^{\text{Dec,LR}}(pp, c^*, K^*)$
Return $(b = b')$

proc Dec$(\phi, c)$  // $\boxed{\text{G}_0^A(\lambda)}$
If $(\phi \notin \Phi)$ then Return $\perp$
$sk' \leftarrow \phi(sk)$
If $((sk' = sk) \wedge (c = c^*))$ then Return $\perp$
$w \leftarrow \mathsf{H.Eval}(pp_{\mathsf{H}}, c_1 || e(g,h)^{sk'})$
If $(e(c_1, u_1 u_2^w) \neq e(g, c_2))$ then Return $\perp$
If $(w = w^*)$ then $\mathsf{bad} \leftarrow \mathsf{true}$ ; $\boxed{\text{Return } \perp}$
Return $e(c_1, h)^{sk'}$
```

Figure 5.9: Games used in the proof of Theorem 5.2.3.

to $\mathbb{G}_2$ to accommodate these changes. We store the exponent sk instead of the group element h^{sk} so that the secret key is a single element of $\mathbb{Z}_p$. We then apply the technique of Section 4.4 by including the public key in the hash, resulting in a Φ^{aff}-RKA-secure KEM. These modifications result in a scheme that is slightly less efficient than the original KEM.

Theorem 5.2.3 *Let* H *be a collision-resistant hash function and DBDH be hard for* G*. Then* MBMW *is* Φ^{aff}*-RKA secure.*

Proof: Let A be a PT adversary playing game IND-RKA. We build PT adversaries

A_1, A_2 such that

$$\mathbf{Adv}^{\text{ind-rka}}_{\mathsf{MBMW},\Phi,A}(\lambda) \leq 2\mathbf{Adv}^{\text{coll}}_{\mathsf{H},A_1}(\lambda) + \mathbf{Adv}^{\text{dbdh}}_{\mathsf{G},A_1}(\lambda)$$

for all $\lambda \in \mathbb{N}$, from which the theorem follows.

The proof uses the games in Figure 5.9. Game G_0 is as game IND-RKA$_{\mathsf{MBMW},\Phi}$, but sets the flag bad and returns $\perp$ if the adversary makes a decapsulation query for a valid ciphertext with the same hash value as the challenge ciphertext.

We will build A_1, A_2 so that for all $\lambda \in \mathbb{N}$ we have

$$\Pr[\mathrm{G}_0^A(\lambda) \text{ sets } \mathsf{bad}] \leq \mathbf{Adv}^{\text{coll}}_{\mathsf{H},A_1}(\lambda) \tag{5.12}$$

$$2\Pr[\mathrm{G}_0^A(\lambda)] - 1 \leq \mathbf{Adv}^{\text{dbdh}}_{\mathsf{G},A_2}(\lambda)\ . \tag{5.13}$$

Games IND-RKA$_{\mathsf{MBMW},\Phi}$ and G_0 are identical until bad, so by the Fundamental Lemma of Game-Playing [23] and the above, for all $\lambda \in \mathbb{N}$ we have:

$$\begin{aligned}
\mathbf{Adv}^{\text{ind-rka}}_{\mathsf{MBMW},\Phi,A}(\lambda) &= 2\Pr[\text{IND-RKA}^A_{\mathsf{MBMW},\Phi}(\lambda)] - 1 \\
&= 2(\Pr[\text{IND-RKA}^A_{\mathsf{MBMW},\Phi}(\lambda)] - \Pr[\mathrm{G}_0^A(\lambda)]) + 2\Pr[\mathrm{G}_0^A(\lambda)] - 1 \\
&\leq 2\Pr[\mathrm{G}_0^A(\lambda) \text{ sets } \mathsf{bad}] + 2\Pr[\mathrm{G}_0^A(\lambda)] - 1 \\
&\leq 2\mathbf{Adv}^{\text{coll}}_{\mathsf{H},A_1}(\lambda) + \mathbf{Adv}^{\text{dbdh}}_{\mathsf{G},A_2}(\lambda)
\end{aligned}$$

as desired. We proceed to the constructions of A_1, A_2. Adversary A_1 against the collision resistance of H behaves as follows:

```
$\underline{A_1(pp_{\mathsf{H}})}$
$x' \leftarrow \perp$
$b \leftarrow_\$ \{0,1\}$ ; $c^* \leftarrow \perp$
$(p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e) \leftarrow_\$ \mathsf{G.Pg}(1^\lambda)$
$g \leftarrow_\$ \mathbb{G}_1^*$ ; $h \leftarrow_\$ \mathbb{G}_2^*$
$y_1, y_2 \leftarrow_\$ \mathbb{Z}_p^*$
$u_1 \leftarrow h^{y_1}$ ; $u_2 \leftarrow h^{y_2}$
$pp \leftarrow (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e, g, h, u_1, u_2, pp_{\mathsf{H}})$
$sk \leftarrow_\$ \mathbb{Z}_p^*$
$pk \leftarrow e(g,h)^{sk}$
$t \leftarrow_\$ \mathbb{Z}_p^*$
$c_1^* \leftarrow g^t$
$w^* \leftarrow \mathsf{H.Eval}(pp_{\mathsf{H}}, c_1^* || pk)$
$x \leftarrow c_1^* || pk$
$c_2^* \leftarrow u_1^t u_2^{tw^*}$
$c^* \leftarrow (c_1^*, c_2^*)$
$K^* \leftarrow pk^t$
If $(b = 0)$ then $K^* \leftarrow_\$ \mathsf{KSp}(pp)$
$b' \leftarrow_\$ A^{\text{DEC}}(pp, pk, c^*, K^*)$
Return $(x, x')$
```

```
$\underline{\text{DEC}(\phi, c)}$
If $(\phi \notin \Phi)$ then Return $\perp$
$sk' \leftarrow \phi(sk)$
If $((sk' = sk) \wedge (c = c^*))$
   then Return $\perp$
$w \leftarrow \mathsf{H.Eval}(pp_{\mathsf{H}}, c_1 || e(g,h)^{sk'})$
If $(e(c_1, u_1 u_2^w) \neq e(g, c_2))$
   then Return $\perp$
If $(w = w^*)$
   then $\mathsf{bad} \leftarrow \mathsf{true}$
   $x' \leftarrow c_1 || e(g,h)^{sk'}$ ; Return $\perp$
Return $e(c_1, h)^{sk'}$
```

If bad is set 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 $(g^s, u_1^s u_2^{xs})$ for some $s \in \mathbb{Z}_p$ where $w = \mathsf{H.Eval}(pp_{\mathsf{H}}, g^s || e(g,h)^{\phi(sk)})$. Then we have a pair of values $x = g^t || e(g,h)^{sk}$ and $x' = g^s || e(g,h)^{\phi(sk)}$ with $\mathsf{H.Eval}(pp_{\mathsf{H}}, x) = \mathsf{H.Eval}(pp_{\mathsf{H}}, x')$. To show this is a collision it remains to show that $x \neq x'$. If $x = x'$ then $e(g,h)^{sk} = e(g,h)^{\phi(sk)}$, so $\phi(sk) = sk$ (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 third 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 $g^t \neq g^s$, contradicting the assumption that $x = x'$. We deduce that x and x' are not equal. A_1 outputs (x, x'), and we then have that $\Pr[\mathrm{G}_0^A(\lambda)$ sets $\mathsf{bad}] \leq \mathbf{Adv}^{\mathrm{coll}}_{\mathsf{H},A_1}(\lambda)$, establishing Equation (5.12).

Adversary A_2 against the DBDH problem behaves as follows:

```
$A_2(g, g^\alpha, g^\beta, g^\gamma, h, h^\alpha, h^\beta, h^\gamma, T)$
$b \leftarrow\!\!\$\ \{0,1\}\ ;\ c^* \leftarrow \perp$
$(p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e) \leftarrow\!\!\$\ \mathsf{G.Pg}(1^\lambda)$
$g \leftarrow\!\!\$\ \mathbb{G}_1^*\ ;\ h \leftarrow\!\!\$\ \mathbb{G}_2^*$
$pp_\mathsf{H} \leftarrow\!\!\$\ \mathsf{H.Pg}(1^\lambda)$
$pk \leftarrow e(g^\alpha, h^\beta)$
$c_1^* \leftarrow g^\gamma$
$w^* \leftarrow \mathsf{H.Eval}(pp_\mathsf{H}, c_1^* || pk)$
$\delta \leftarrow\!\!\$\ \mathbb{Z}_p$
$u_1 \leftarrow (h^\alpha)^{-w^*} h^\delta$
$u_2 \leftarrow h^\alpha$
$pp \leftarrow (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e, g, h, u_1, u_2, pp_\mathsf{H})$
$c_2^* \leftarrow (h^\gamma)^\delta$
$c^* \leftarrow (c_1^*, c_2^*)$
$K^* \leftarrow T$
$b' \leftarrow\!\!\$\ A^{\mathrm{DEC}}(pp, pk, c^*, K^*)$
Return $b'$
```

```
$\mathrm{DEC}(\phi_{a,b}, c)$
If $(\phi_{a,b} \notin \Phi)$ then Return $\perp$
$pk' \leftarrow pk^a e(g, h)^b$
$w \leftarrow \mathsf{H.Eval}(pp_\mathsf{H}, c_1 || pk')$
If $((pk' = pk) \wedge (c = c^*))$
  then Return $\perp$
If $(e(c_1, u_1 u_2^w) \neq e(g, c_2))$
  then Return $\perp$
If $(w = w^*)$
  then Return $\perp$
$K \leftarrow \frac{e(c_1, (h^\beta)^{\frac{-\delta}{w-w*}} u_1 u_2^w)}{e((g^\beta)^{\frac{-1}{w-w^*}} g, c_2)}$
Return $K$
```

A_2 computes $pk \leftarrow e(g^\alpha, h^\beta)$ so that the private key sk is the unknown value $\alpha\beta$. It chooses $\delta \leftarrow\!\!\$\ \mathbb{Z}_p$ and sets

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

computing the challenge ciphertext as $c^* \leftarrow (g^\gamma, (h^\gamma)^\delta)$. This is a valid ciphertext as $c^* = (g^\gamma, u_1^\gamma u_2^{\gamma w^*})$. A_2 sets the challenge session key $K^* \leftarrow T$, so that it is the correct session key when the DBDH tuple has $T = e(g, h)^{\alpha\beta\gamma}$, and a random element of $\mathbb{G}_T$ otherwise. Despite not knowing sk, A_2 can detect when $\phi(sk) = sk$ by checking whether $pk = pk^a \cdot e(g, h)^b$. Keys are decapsulated as

$$K \leftarrow \frac{e(c_1, (h^\beta)^{\frac{-\delta}{w-w*}} u_1 u_2^w)}{e((g^\beta)^{\frac{-1}{w-w^*}} g, c_2)} .$$

Note that, assuming all the decapsulation oracle's checks hold, we have:

$$K = \left(\frac{e(g, h)^{\alpha(w-w*)+\delta-\beta\frac{\delta}{w-w^*}}}{e(g, h)^{\alpha(w-w*)+\delta-\beta\frac{\delta}{w-w^*}-\alpha\beta\frac{w-w^*}{w-w^*}}} \right)^t = e(g, h)^{\alpha\beta t} ,$$

where $c_1 = g^t$, i.e. t is the randomness used in creating the ciphertext. Hence this decapsulated key is correct. A_2 provides a perfect simulation of G_0 for A_2, so A_2 wins its game precisely when A wins, giving $2\Pr[\mathrm{G}_0^A(\lambda)] - 1 \leq \mathbf{Adv}^{\mathrm{dbdh}}_{\mathsf{G},A_2}(\lambda)$, establishing Equation (5.13). ▌

Our scheme bears comparison to the PKE schemes of Wee [99]. We reiterate that Wee only achieves security for the claw-free RKD set Φ^{lin}, whereas our scheme

MBMW is Φ^{aff}-RKA secure. The most directly comparable scheme is the one in [99, 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 MBMW, 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 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 MBMW.

By comparison, our scheme MBMW has ciphertexts that consist of two group elements, one in $\mathbb{G}_1$ and one in $\mathbb{G}_2$. 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 parameters are larger. The costs of encapsulation and decapsulation for MBMW 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 MBMW is RKA secure for a significantly larger class of RKD functions.

5.3 Joint Security in the RKA Setting

We combine the existing security definitions for Φ-RKA security and joint security to produce a new security model for Φ-RKA security of joint encryption and signature schemes. We also extend the results of [14] and Chapter 3 to show that the joint encryption and signature scheme JES[IBE, DS] of Chapter 3 is Φ-RKA secure if the starting IBE scheme is Φ-RKA-secure. This construction can be applied to obtain efficient JES schemes for interesting sets Φ using any of the RKA-secure IBE schemes from Chapter 4.

The games associated with the notions of security for JES schemes are extended to Φ-RKA-security in Figure 5.10

We say that JES is Φ-RKA IND-secure if $\mathbf{Adv}^{\text{ind-rka}}_{\mathsf{JES},\Phi,A}(\cdot)$ is negligible for all PT adversaries A, where $\mathbf{Adv}^{\text{ind-rka}}_{\mathsf{JES},\Phi,A}(\lambda) = 2\Pr[\text{IND-RKA}^{A}_{\mathsf{JES},\Phi}(\lambda)] - 1$ and game IND-RKA

```
main IND-RKA$^A_{\mathsf{JES},\Phi}(\lambda)$
$b \leftarrow_\$ \{0,1\}$ ; $c^* \leftarrow \perp$
$pp \leftarrow_\$ \mathsf{JES.Pg}(1^\lambda)$
$(sk, pk) \leftarrow_\$ \mathsf{JES.Kg}(pp)$
$b' \leftarrow_\$ A^{\text{Dec},\text{Sign},\text{LR}}(pp, pk)$
Return $(b = b')$

proc Dec$(\phi, c)$
If $(\phi \notin \Phi)$ then Return $\perp$
$sk' \leftarrow \phi(sk)$
If $((sk' = sk) \wedge (c = c^*))$ then Return $\perp$
Return $m \leftarrow \mathsf{JES.Dec}(pp, sk', c)$

proc Sign$(\phi, m)$
If $(\phi \notin \Phi)$ then Return $\perp$
$sk' \leftarrow \phi(sk)$
Return $\sigma \leftarrow_\$ \mathsf{JES.Sign}(pp, sk', m)$

proc LR$(m_0, m_1)$
If $(c^* \neq \perp)$ then Return $\perp$
If $(|m_0| \neq |m_1|)$ then Return $\perp$
$c^* \leftarrow_\$ \mathsf{JES.Enc}(pp, pk, m_b)$
Return $c^*$
```

```
main EUF-RKA$^A_{\mathsf{JES},\Phi}(\lambda)$
$Q \leftarrow \emptyset$
$pp \leftarrow_\$ \mathsf{JES.Pg}(1^\lambda)$
$(sk, pk) \leftarrow_\$ \mathsf{JES.Kg}(pp)$
$(m, \sigma) \leftarrow_\$ A^{\text{Sign},\text{Dec}}(pp, pk)$
Return $(\mathsf{JES.Verify}(pp, pk, m, \sigma) \wedge m \notin Q)$

proc Sign$(\phi, m)$
If $(\phi \notin \Phi)$ then Return $\perp$
$sk' \leftarrow \phi(sk)$
$\sigma \leftarrow_\$ \mathsf{JES.Sign}(pp, sk', m)$
If $(sk' = sk)$ then $Q \leftarrow Q \cup \{m\}$
Return $\sigma$

proc Dec$(\phi, c)$
If $(\phi \notin \Phi)$ then Return $\perp$
$sk' \leftarrow \phi(sk)$
Return $m \leftarrow \mathsf{JES.Dec}(pp, sk', c)$
```

Figure 5.10: Left: Game IND-RKA defining indistinguishability of joint encryption and signature scheme JES under chosen-ciphertext related-key attack in the presence of a signing oracle. Right: Game EUF-RKA defining existential unforgeability under chosen-message related-key attack in the presence of a decryption oracle.

```
main OW-aID-RKA$^A_{\mathsf{IBE},\Phi}(\lambda)$
$c^* \leftarrow \perp$ ; $u^* \leftarrow \perp$ ; $U \leftarrow \emptyset$
$pp \leftarrow_\$ \mathsf{IBE.Pg}(1^\lambda)$
$m^* \leftarrow_\$ \mathsf{IBE.MSp}(pp)$
$(msk, mpk) \leftarrow_\$ \mathsf{IBE.MKg}(pp)$
$m \leftarrow_\$ A^{\mathrm{KD},\textsc{Chal}}(pp, mpk)$
Return $(m^* = m)$

proc $\phi, \mathrm{KD}(\phi, u)$
If $(\phi \notin \Phi)$ then Return $\perp$
$msk' \leftarrow \phi(msk)$
If $(msk' = msk)$ then $U \leftarrow U \cup \{u\}$
If $(u^* \in U)$ then Return $\perp$
Return $\mathsf{IBE.UKg}(pp, msk, u)$

proc $\textsc{Chal}(u)$
If $(c^* \neq \perp)$ then Return $\perp$
$u^* \leftarrow u$
If $(u^* \in U)$ then Return $\perp$
$c^* \leftarrow_\$ \mathsf{IBE.Enc}(pp, mpk, u^*, m^*)$
Return $c^*$
```

Figure 5.11: Game OW-aID-RKA defining Φ-RKA one-wayness of identity-based encryption scheme IBE.

is on the left-hand side of Figure 5.10. This represents indistinguishability under chosen-ciphertext related-key attack in the presence of a signing oracle.

We say that JES is Φ-RKA EUF-secure if $\mathbf{Adv}^{\text{euf-rka}}_{\mathsf{JES},\Phi,A}(\cdot)$ is negligible for all PT adversaries A, where $\mathbf{Adv}^{\text{euf-rka}}_{\mathsf{JES},\Phi,A}(\lambda) = \Pr[\text{EUF-RKA}^A_{\mathsf{JES},\Phi}(\lambda)]$ and game EUF-RKA is on the right-hand side of Figure 5.10. This represents existential unforgeability under chosen-message related-key attack in the presence of a decryption oracle.

Informally, we say that a JES scheme is Φ-RKA secure if it is both Φ-RKA IND-secure and Φ-RKA EUF-secure.

We will make use of the following definition of IBE security. We say an identity-based encryption scheme IBE is *Φ-RKA one-way under chosen plaintext attack* or *Φ-RKA one-way secure* if $\mathbf{Adv}^{\text{ow-aid-rka}}_{\mathsf{IBE},\Phi,A}(\cdot)$ is negligible for all PT adversaries A, where $\mathbf{Adv}^{\text{ow-aid-rka}}_{\mathsf{PKE},\Phi,A}(\lambda) = \Pr[\text{OW-aID-RKA}^A_{\mathsf{IBE},\Phi}(\lambda)]$ and game OW-aID-RKA is in Figure 5.11.

The following theorems show that the JES scheme $\mathsf{JES}[\mathsf{IBE},\mathsf{DS}]$ of Chapter 3 inherits Φ-RKA security from the starting IBE scheme.

Theorem 5.3.1 *Let* IBE *be a selectively* Φ*-RKA secure IBE scheme and* DS *be a one-time strongly secure signature scheme. Then* $\mathsf{JES}[\mathsf{IBE},\mathsf{DS}]$ *is* Φ*-RKA IND-secure.*

Proof: Let A be a PT adversary playing game IND-RKA. We build PT adversaries A_1, A_2 such that

$$\mathbf{Adv}^{\text{ind-rka}}_{\mathsf{JES}[\mathsf{IBE},\mathsf{DS}],\Phi,A}(\lambda) \leq 2\mathbf{Adv}^{\text{ot-suf-cma}}_{\mathsf{DS},A_1}(\lambda) + \mathbf{Adv}^{\text{ind-sid-rka}}_{\mathsf{IBE},\Phi,A_2}(\lambda)$$

for all $\lambda \in \mathbb{N}$, from which the theorem follows.

The proof uses the games in Figure 5.12. We will build A_1, A_2 so that for all $\lambda \in \mathbb{N}$ we have

$$\Pr[\mathrm{G}_0^A(\lambda) \text{ sets } \mathsf{bad}] \leq \mathbf{Adv}^{\text{ot-suf-cma}}_{\mathsf{DS},A_1}(\lambda) \tag{5.14}$$

$$2\Pr[\mathrm{G}_0^A(\lambda)] - 1 \leq \mathbf{Adv}^{\text{ind-sid-rka}}_{\mathsf{IBE},\Phi,A_2}(\lambda)\ . \tag{5.15}$$

Games $\text{IND-RKA}_{\mathsf{JES}[\mathsf{IBE},\mathsf{DS}],\Phi}$ and G_0 are identical until bad, so by the Fundamental Lemma of Game-Playing [23] and the above, for all $\lambda \in \mathbb{N}$ we have:

$$\begin{aligned}
\mathbf{Adv}^{\text{ind-rka}}_{\mathsf{JES}[\mathsf{IBE},\mathsf{DS}],\Phi,A}(\lambda) &= 2\Pr[\text{IND-RKA}^A_{\mathsf{JES}[\mathsf{IBE},\mathsf{DS}],\Phi}(\lambda)] - 1 \\
&= 2(\Pr[\text{IND-RKA}^A_{\mathsf{JES}[\mathsf{IBE},\mathsf{DS}],\Phi}(\lambda)] - \Pr[\mathrm{G}_0^A(\lambda)]) \\
&\qquad + 2\Pr[\mathrm{G}_0^A(\lambda)] - 1 \\
&\leq 2\Pr[\mathrm{G}_0^A(\lambda) \text{ sets } \mathsf{bad}] + 2\Pr[\mathrm{G}_0^A(\lambda)] - 1 \\
&\leq 2\mathbf{Adv}^{\text{ot-suf-cma}}_{\mathsf{DS},A_1}(\lambda) + \mathbf{Adv}^{\text{ind-sid-rka}}_{\mathsf{IBE},\Phi,A_2}(\lambda)
\end{aligned}$$

as desired. We proceed to the constructions of A_1, A_2,. Adversary A_1 against the one-time strong unforgeability of DS behaves as follows:

```
main IND-RKA$^A_{\mathsf{JES[IBE,DS]},\Phi}(\lambda)$ / $\boxed{G_0^A(\lambda)}$
$b \leftarrow_\$ \{0,1\}$ ; $c^* \leftarrow \perp$ ; $pk^*_{\mathsf{DS}} \leftarrow \perp$
$pp_{\mathsf{IBE}} \leftarrow_\$ \mathsf{IBE.Pg}(1^\lambda)$ ; $pp_{\mathsf{DS}} \leftarrow_\$ \mathsf{DS.Pg}(1^\lambda)$
$(msk, mpk) \leftarrow_\$ \mathsf{IBE.MKg}(pp_{\mathsf{IBE}})$
$b' \leftarrow_\$ A^{\text{Dec,Sign,LR}}((pp_{\mathsf{IBE}}, pp_{\mathsf{DS}}), mpk)$
Return $(b = b')$

proc Dec$(\phi, c)$
If $(\phi \notin \Phi)$ then Return $\perp$
$msk' \leftarrow \phi(msk)$
If $((msk' = msk) \wedge (c = c^*))$ then Return $\perp$
$(pk_{\mathsf{DS}}, c_{\mathsf{IBE}}, \sigma_{\mathsf{DS}}) \leftarrow c$
If $(!\mathsf{DS.Verify}(pp_{\mathsf{DS}}, pk_{\mathsf{DS}}, c_{\mathsf{IBE}}, \sigma_{\mathsf{DS}}))$ then Return $\perp$
If $((pk_{\mathsf{DS}} = pk^*_{\mathsf{DS}}) \wedge ((c_{\mathsf{IBE}}, \sigma_{\mathsf{DS}}) \neq (c^*_{\mathsf{IBE}}, \sigma^*_{\mathsf{DS}})))$ then $\mathsf{bad} \leftarrow \mathsf{true}$ ; $\boxed{\text{Return } \perp}$
$u \leftarrow 1||pk_{\mathsf{DS}}$
$usk \leftarrow_\$ \mathsf{IBE.UKg}(pp_{\mathsf{IBE}}, msk', u)$
Return $\mathsf{IBE.Dec}(pp_{\mathsf{IBE}}, usk, c_{\mathsf{IBE}})$

proc Sign$(m)$
If $(\phi \notin \Phi)$ then Return $\perp$
$msk' \leftarrow \phi(msk)$
$u \leftarrow 0||m$
Return $\mathsf{IBE.UKg}(pp_{\mathsf{IBE}}, msk', u)$

proc LR$(m_0, m_1)$
If $(c^* \neq \perp)$ then Return $\perp$
If $(|m_0| \neq |m_1|)$ then Return $\perp$
$(sk^*_{\mathsf{DS}}, pk^*_{\mathsf{DS}}) \leftarrow_\$ \mathsf{DS.Kg}(pp_{\mathsf{DS}})$
$u \leftarrow 1||pk^*_{\mathsf{DS}}$
$c^*_{\mathsf{IBE}} \leftarrow_\$ \mathsf{IBE.Enc}(pp_{\mathsf{IBE}}, mpk, u, m)$
$\sigma^*_{\mathsf{DS}} \leftarrow_\$ \mathsf{DS.Sign}(pp_{\mathsf{DS}}, sk^*_{\mathsf{DS}}, c^*_{\mathsf{IBE}})$
$c^* \leftarrow (pk^*_{\mathsf{DS}}, c^*_{\mathsf{IBE}}, \sigma^*_{\mathsf{DS}})$
Return $c^*$
```

Figure 5.12: Games used in the proof of Theorem 5.3.1.

```
$\underline{A_1^{\mathrm{SIGN}}(pp_{\mathsf{DS}}, pk^*_{\mathsf{DS}})}$
$(m^*, \sigma^*) \leftarrow \perp$
$b \leftarrow\$ \{0,1\}$ ; $c^* \leftarrow \perp$
$pp_{\mathsf{IBE}} \leftarrow\$ \mathsf{IBE.Pg}(1^\lambda)$
$(msk, mpk) \leftarrow\$ \mathsf{IBE.MKg}(pp_{\mathsf{IBE}})$
$b' \leftarrow\$ A^{\mathrm{DEC},\mathrm{SIGNSIM},\mathrm{LR}}((pp_{\mathsf{IBE}}, pp_{\mathsf{DS}}), mpk)$
Return $(m^*, \sigma^*)$

$\underline{\mathrm{DEC}(\phi, c)}$
If $(\phi \notin \Phi)$ then Return $\perp$
$msk' \leftarrow \phi(msk)$
If $((msk' = msk) \wedge (c = c^*))$ then Return $\perp$
$(pk_{\mathsf{DS}}, c_{\mathsf{IBE}}, \sigma_{\mathsf{DS}}) \leftarrow c$
If $(!\mathsf{DS.Verify}(pp_{\mathsf{DS}}, pk_{\mathsf{DS}}, c_{\mathsf{IBE}}, \sigma_{\mathsf{DS}}))$
   then Return $\perp$
If $((pk_{\mathsf{DS}} = pk^*_{\mathsf{DS}}) \wedge ((c_{\mathsf{IBE}}, \sigma_{\mathsf{DS}}) \neq (c^*_{\mathsf{IBE}}, \sigma^*_{\mathsf{DS}})))$
   then $(m^*, \sigma^*) \leftarrow (c_{\mathsf{IBE}}, \sigma_{\mathsf{DS}})$ ; Return $\perp$
$u \leftarrow 1||pk_{\mathsf{DS}}$
$usk \leftarrow\$ \mathsf{IBE.UKg}(pp_{\mathsf{IBE}}, msk', u)$
Return $\mathsf{IBE.Dec}(pp_{\mathsf{IBE}}, usk, c_{\mathsf{IBE}})$

$\underline{\mathrm{SIGNSIM}(\phi, m)}$
If $(\phi \notin \Phi)$ then Return $\perp$
$msk' \leftarrow \phi(msk)$
$u \leftarrow 0||m$
Return $\mathsf{IBE.UKg}(pp_{\mathsf{IBE}}, msk', u)$

$\underline{\mathrm{LR}(m_0, m_1)}$
If $(c^* \neq \perp)$ then Return $\perp$
If $(|m_0| \neq |m_1|)$ then Return $\perp$
$u \leftarrow 1||pk^*_{\mathsf{DS}}$
$c^*_{\mathsf{IBE}} \leftarrow\$ \mathsf{IBE.Enc}(pp_{\mathsf{IBE}}, mpk, u, m_b)$
$\sigma^*_{\mathsf{DS}} \leftarrow\$ \mathrm{SIGN}(c^*_{\mathsf{IBE}})$
$c^* \leftarrow (pk^*_{\mathsf{DS}}, c^*_{\mathsf{IBE}}, \sigma^*_{\mathsf{DS}})$
Return $c^*$
```

Adversary A_1 simulates game G_0 for A, using its challenge public key in creating the challenge ciphertext for A. Since SIGN is only called when A calls LR and $c^* = \perp$, and this situation can occur only on A's first call to LR, A_1 makes at most one call to its SIGN oracle. The forgery that A_1 outputs is valid exactly when the flag bad is set in game G_0. The "If" statements before the forgery (m^*, σ^*) is set ensure this. The third "If" statement ensures the signature is valid, and the fourth ensures the pair differs from that output by the signing oracle. This establishes Equation (5.14).

Adversary A_2 against the selective-ID Φ-RKA security of IBE behaves as follows:

```
$\underline{A_2^{\mathrm{sID,KD,LR}}(pp_{\mathsf{IBE}})}$
$c^* \leftarrow \perp$
$pp_{\mathsf{DS}} \leftarrow\!\!\$\ \mathsf{DS.Pg}(1^\lambda)$
$(sk^*_{\mathsf{DS}}, pk^*_{\mathsf{DS}}) \leftarrow\!\!\$\ \mathsf{DS.Kg}(pp_{\mathsf{DS}})$
$u^* \leftarrow 1 || pk^*_{\mathsf{DS}}$
$mpk \leftarrow \mathrm{sID}(u^*)$
$b' \leftarrow\!\!\$\ A^{\mathrm{Dec,Sign,LRSim}}((pp_{\mathsf{IBE}}, pp_{\mathsf{DS}}), mpk)$
Return $b'$

$\underline{\mathrm{Dec}(\phi, c)}$
If $(\phi \notin \Phi)$ then Return $\perp$
$(pk_{\mathsf{DS}}, c_{\mathsf{IBE}}, \sigma_{\mathsf{DS}}) \leftarrow c$
If $(!\mathsf{DS.Verify}(pp_{\mathsf{DS}}, pk_{\mathsf{DS}}, c_{\mathsf{IBE}}, \sigma_{\mathsf{DS}}))$
    then Return $\perp$
If $((pk_{\mathsf{DS}} = pk^*_{\mathsf{DS}}) \wedge ((c_{\mathsf{IBE}}, \sigma_{\mathsf{DS}}) \neq (c^*_{\mathsf{IBE}}, \sigma^*_{\mathsf{DS}})))$
    then Return $\perp$
$u \leftarrow 1 || pk_{\mathsf{DS}}$
$usk \leftarrow\!\!\$\ \mathrm{KD}(\phi, u)$
If $(usk = \perp)$ then Return $\perp$
Return $\mathsf{IBE.Dec}(pp_{\mathsf{IBE}}, usk, c_{\mathsf{IBE}})$

$\underline{\mathrm{Sign}(\phi, m)}$
If $(\phi \notin \Phi)$ then Return $\perp$
$u \leftarrow 0 || m$
Return $\mathrm{KD}(\phi, u)$

$\underline{\mathrm{LRSim}(m_0, m_1)}$
If $(c^* \neq \perp)$ then Return $\perp$
If $(|m_0| \neq |m_1|)$ then Return $\perp$
$c^*_{\mathsf{IBE}} \leftarrow\!\!\$\ \mathrm{LR}(m_0, m_1)$
$\sigma^*_{\mathsf{DS}} \leftarrow\!\!\$\ \mathsf{DS.Sign}(pp_{\mathsf{DS}}, sk^*_{\mathsf{DS}}, c^*_{\mathsf{IBE}})$
$c^* \leftarrow (pk^*_{\mathsf{DS}}, \sigma^*_{\mathsf{DS}}, c^*_{\mathsf{IBE}})$
Return $c^*$
```

Adversary A_2 simulates game G_0 for A, choosing up-front the public verification key that will determine the identity used in the challenge ciphertext. Adversary A_2 must return $\perp$ when A makes a decryption query with $(\phi(msk) = msk) \wedge (c = c^*)$, without knowing msk. If the ciphertext A submits for decryption is the challenge ciphertext, then the identity submitted to A_2's KD oracle is the challenge identity, and if $\phi(msk) = msk$ then the KD oracle will return $\perp$, so A_2 detects this case and returns $\perp$ as required.

The challenge identity will never otherwise be queried to the KD oracle by Dec as the third if statement will cause $\perp$ to be returned before the KD oracle is called, and will not be queried by Sign as the challenge identity begins with 1 and all Sign's KD queries are for identities beginning with 0. Adversary A_2 outputs A's guess as its own, winning whenever A does, establishing Equation (5.15). ▌

Theorem 5.3.2 *Let* IBE *be a* Φ*-RKA one-way IBE scheme. Then* $\mathsf{JES}[\mathsf{IBE}, \mathsf{DS}]$ *is* Φ*-RKA EUF-secure.*

Proof: Let A be a PT adversary playing game EUF-RKA. We build a PT adversary

A_1 such that

$$\mathbf{Adv}^{\text{euf-rka}}_{\mathsf{JES[IBE,DS]},\Phi,A}(\lambda) \leq \mathbf{Adv}^{\text{ow-aid-rka}}_{\mathsf{IBE},\Phi,A_1}(\lambda) \tag{5.16}$$

for all $\lambda \in \mathbb{N}$, from which the theorem follows. Adversary A_1 against the Φ-RKA one-wayness of IBE behaves as follows:

```
$A_1^{\text{KD},\textsc{Chal}}(pp_{\mathsf{IBE}})$
$pp_{\mathsf{DS}} \leftarrow_\$ \mathsf{DS.Pg}(1^\lambda)$
$(m, \sigma) \leftarrow_\$ A^{\textsc{Sign},\textsc{Dec}}((pp_{\mathsf{IBE}}, pp_{\mathsf{DS}}), mpk)$
$usk \leftarrow \sigma$
$u \leftarrow 0||m$
$c^*_{\mathsf{IBE}} \leftarrow \textsc{Chal}(u)$
Return $\mathsf{IBE.Dec}(pp_{\mathsf{IBE}}, usk, c^*_{\mathsf{IBE}})$

$\textsc{Sign}(\phi, m)$
If $(\phi \notin \Phi)$ then Return $\perp$
$u \leftarrow 0||m$
Return $\text{KD}(\phi, u)$

$\textsc{Dec}(\phi, c)$
If $(\phi \notin \Phi)$ then Return $\perp$
$(pk_{\mathsf{DS}}, c_{\mathsf{IBE}}, \sigma_{\mathsf{DS}}) \leftarrow c$
If $(!\mathsf{DS.Verify}(pp_{\mathsf{DS}}, pk_{\mathsf{DS}}, c_{\mathsf{IBE}}, \sigma_{\mathsf{DS}}))$
   then Return $\perp$
$u \leftarrow 1||pk_{\mathsf{DS}}$
$usk \leftarrow_\$ \text{KD}(\phi, u)$
Return $\mathsf{IBE.Dec}(pp_{\mathsf{IBE}}, usk, c_{\mathsf{IBE}})$
```

Adversary A_1 simulates game EUF-RKA for A, using its KD oracle to answer Dec and Sign queries. When A halts outputting a message m and signature σ, A_1 submits $0||m$ as its challenge identity. If A did not submit m to its Sign oracle then this identity has not been queried to the KD oracle so the challenge oracle will respond with a challenge ciphertext. If A output a valid signature on m then the challenge ciphertext will decrypt successfully using the signature as a user-level key, establishing Equation (5.16). ▌

5.4 RKA-Secure Symmetric Encryption

We now show how to build an RKA-secure symmetric encryption scheme starting from an IBE scheme meeting certain malleability properties and admitting a collision-resistant identity renaming scheme.

We first show that the natural method of converting a PKE scheme into a symmetric encryption scheme gives a Φ-RKA-secure CCA symmetric encryption scheme when the PKE scheme meets a strong notion of RKA security, wherein the challenge ciphertext is encrypted under a related key of the adversary's choosing. We then show that if the underlying IBE scheme meets a similar notion of strong RKA security, the CHK transform preserves this strong RKA security, giving strong Φ-RKA-secure PKE from selective-ID strong Φ-RKA-secure IBE.

Finally, we show that applying the identity renaming transform of Section 4.4 to an IBE scheme with the appropriate properties results in an IBE scheme that achieves selective-ID strong Φ-RKA-security. Waters' IBE scheme and the extended Waters IBE scheme are shown to have the required properties.

5.4.1 RKA-Secure Symmetric Encryption from PKE

A definition of Φ-RKA-security for a symmetric encryption scheme is presented in Figure 5.13. 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 [14], 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. We say a symmetric encryption scheme SE is Φ-*RKA secure* if $\mathbf{Adv}^{\text{ind-rka}}_{\mathsf{SE},\Phi,A}(\cdot)$ is negligible for all PT adversaries A, where $\mathbf{Adv}^{\text{ind-rka}}_{\mathsf{SE},\Phi,A}(\lambda) = 2\Pr[\text{IND-RKA}^{A}_{\mathsf{SE},\Phi}(\lambda)] - 1$ and game IND-RKA is in Figure 5.13.

We want to build a symmetric encryption scheme meeting this notion of security. Since we already built RKA-secure PKE in Section 5.2, our starting point is a PKE scheme, with the whole kepair (sk, pk) serving as the symmetric key K, and encrypting and decrypting as usual. However with this approach, the public key component of K is also subject to modification by the related-key derivation function. To avoid this, we require that the PKE scheme be canonical, and store only the private key component as the symmetric key, deterministically computing the public key when it is needed for encryption via $pk \leftarrow \mathsf{PKE.PK}(pp, sk)$. The algorithms of this scheme can be seen in Figure 5.14.

In the standard (non-RKA) setting, it is clear that security of the SE scheme follows from security of the PKE scheme. The reduction uses the PKE scheme adversary's own decryption oracle to handle decryption queries, and knowledge of the public key to handle encryption queries and the LR query. In the RKA setting, we can still handle decryption queries with the PKE adversary's own decryption oracle, however we must now be able to give encryptions relative to transformed public keys $pk \leftarrow \mathsf{PKE.PK}(pp, \phi(K))$.

```
MAIN $\text{IND-RKA}^A_{\mathsf{SE},\Phi}(\lambda)$
$b \leftarrow_{\$} \{0,1\}$ ; $K^* \leftarrow \perp$ ; $c^* \leftarrow \perp$
$pp \leftarrow_{\$} \mathsf{SE.Pg}(1^\lambda)$ ; $K \leftarrow_{\$} \mathsf{SE.SKSp}(pp)$
$b' \leftarrow_{\$} A^{\text{Enc},\text{Dec},\text{LR}}(pp)$
Return $(b = b')$

proc $\text{Enc}(\phi, m)$
If $(\phi \notin \Phi)$ then Return $\perp$
$K' \leftarrow \phi(K)$
Return $\mathsf{SE.Enc}(pp, K', m)$

proc $\text{Dec}(\phi, c)$
If $(\phi \notin \Phi)$ then Return $\perp$
$K' \leftarrow \phi(K)$
If $((K' = K^*) \wedge (c = c^*))$ then Return $\perp$
Return $\mathsf{SE.Dec}(pp, K', c)$

proc $\text{LR}(\phi, m_0, m_1)$
If $(\phi \notin \Phi)$ then Return $\perp$
If $(c^* \neq \perp)$ then Return $\perp$
If $(|m_0| \neq |m_1|)$ then Return $\perp$
$K^* \leftarrow \phi(K)$
$c^* \leftarrow_{\$} \mathsf{SE.Enc}(pp, K^*, m_b)$
Return $c^*$
```

Figure 5.13: Game defining Φ-RKA security of symmetric encryption scheme SE.

```
$\mathsf{SE[PKE].Pg}(1^\lambda)$ :
Return $\mathsf{PKE.Pg}(1^\lambda)$

$\mathsf{SE[PKE].Enc}(pp, K, m)$ :
$pk \leftarrow \mathsf{PKE.PK}(pp, K)$
Return $\mathsf{PKE.Enc}(pp, pk, m)$

$\mathsf{SE[PKE].Kg}(pp)$ :
$K \leftarrow_{\$} \mathsf{PKE.SKSp}(pp)$
Return $K$

$\mathsf{SE[PKE].Dec}(pp, K, c)$ :
Return $\mathsf{PKE.Dec}(pp, K, c)$
```

Figure 5.14: SE scheme SE[PKE] from canonical PKE scheme PKE.

```
main IND-SRKA$^A_{\mathsf{PKE},\Phi}(\lambda)$
$b \leftarrow_\$ \{0,1\}$ ; $sk^* \leftarrow \perp$ ; $c^* \leftarrow \perp$
$pp \leftarrow_\$ \mathsf{PKE.Pg}(1^\lambda)$ ; $sk \leftarrow_\$ \mathsf{PKE.SKSp}(pp)$ ; $pk \leftarrow \mathsf{PKE.PK}(pp, sk)$
$b' \leftarrow_\$ A^{\text{Dec},\text{Enc},\text{LR}}(pp, pk)$
Return $(b = b')$

proc Dec$(\phi, c)$
If $(\phi \notin \Phi)$ then Return $\perp$
$sk' \leftarrow \phi(sk)$
If $((sk' = sk^*) \wedge (c = c^*))$ then Return $\perp$
Return $m \leftarrow \mathsf{PKE.Dec}(pp, sk', c)$

proc Enc$(\phi, m)$
If $(\phi \notin \Phi)$ then Return $\perp$
$sk' \leftarrow \phi(sk)$
$pk' \leftarrow \mathsf{PKE.PK}(pp, sk')$
Return $\mathsf{PKE.Enc}(pp, pk', m)$

proc LR$(\phi, m_0, m_1)$
If $(\phi \notin \Phi)$ then Return $\perp$
If $(c^* \neq \perp)$ then Return $\perp$
If $(|m_0| \neq |m_1|)$ then Return $\perp$
$sk^* \leftarrow \phi(sk)$
$pk^* \leftarrow \mathsf{PKE.PK}(pp, sk^*)$
$c^* \leftarrow_\$ \mathsf{PKE.Enc}(pp, pk^*, m_b)$
Return $c^*$
```

Figure 5.15: Game defining strong Φ-RKA security of PKE scheme PKE.

The notion of *strong-RKA security* is defined for signatures in [14], requiring that an adversary be unable to forge a signature even relative to a transformed public key. Translating this to the case of PKE, we have a notion of security where the adversary cannot distinguish between messages encrypted under a transformed public key. This notion of strong Φ-RKA-security for PKE is exactly what is needed here. We say a public-key encryption scheme PKE is *strong Φ-RKA secure* if $\mathbf{Adv}^{\text{ind-srka}}_{\mathsf{PKE},\Phi,A}(\cdot)$ is negligible for all PT adversaries A, where $\mathbf{Adv}^{\text{ind-srka}}_{\mathsf{PKE},\Phi,A}(\lambda) = 2\Pr[\text{IND-SRKA}^A_{\mathsf{PKE},\Phi}(\lambda)] - 1$ and game IND-SRKA is in Figure 5.15. We then have the following theorem.

Theorem 5.4.1 *Let* PKE *be a strong Φ-RKA-secure PKE scheme. Then the symmetric encryption scheme* $\mathsf{SE}[\mathsf{PKE}]$ *is Φ-RKA secure.*

Proof: Let A be a PT adversary playing game IND-RKA. We build a PT adversary

A_1 such that

$$\mathbf{Adv}^{\text{ind-rka}}_{\mathsf{SE[PKE]},\Phi,A}(\lambda) \leq \mathbf{Adv}^{\text{ind-srka}}_{\mathsf{PKE},\Phi,A_1}(\lambda)$$

for all $\lambda \in \mathbb{N}$, from which the theorem follows.

We will build A_1 so that for all $\lambda \in \mathbb{N}$ we have

$$2\Pr[\text{IND-RKA}^{A}_{\mathsf{SE[PKE]},\Phi}(\lambda)] - 1 \leq \mathbf{Adv}^{\text{ind-srka}}_{\mathsf{PKE},\Phi,A_1}(\lambda) \tag{5.17}$$

as desired. Adversary A_1 behaves as follows:

```
$\underline{A_1^{\text{Enc,Dec,LR}}(pp, pk)}$
$b' \leftarrow_{\$} A^{\text{DecSim,EncSim,LRSim}}(pp)$
Return $b'$

$\underline{\text{DecSim}(\phi, c)}$
Return $\text{Dec}(\phi, c)$

$\underline{\text{EncSim}(\phi, m)}$
Return $\text{Enc}(\phi, m)$

$\underline{\text{LRSim}(\phi, m_0, m_1)}$
Return $\text{LR}(\phi, m_0, m_1)$
```

That is the adversary simply runs A, forwarding all its queries to its own oracles. When A halts and outputs a bit b', A_1 does the same, winning precisely when A does, establishing Equation (5.17). ▌

5.4.2 Strong RKA-Secure PKE from IBE

As shown in Section 5.2.1 and in [14], RKA-secure PKE can be constructed from selective-ID RKA-secure IBE, so to construct strong RKA-secure PKE a natural starting point is strong RKA-secure IBE. We say an IBE scheme IBE is *selectively strong Φ-RKA secure* if $\mathbf{Adv}^{\text{ind-sid-srka}}_{\mathsf{PKE},\Phi,A}(\cdot)$ is negligible for all PT adversaries A, where $\mathbf{Adv}^{\text{ind-sid-srka}}_{\mathsf{PKE},\Phi,A}(\lambda) = 2\Pr[\text{IND-sID-SRKA}^{A}_{\mathsf{PKE},\Phi}(\lambda)] - 1$ and game IND-sID-SRKA is in Figure 5.16. 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. 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 master 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 5.16 is sufficient as a step on the way to our end goal of RKA-secure symmetric encryption.

```
MAIN IND-sID-SRKA$^{A}_{\mathsf{IBE},\Phi}(\lambda)$
$b \leftarrow_{\$} \{0,1\}$ ; $c^* \leftarrow \perp$ ; $u^* \leftarrow \perp$ ; $msk^* \leftarrow \perp$ ; $U \leftarrow \emptyset$
$pp \leftarrow_{\$} \mathsf{IBE.Pg}(1^\lambda)$
$msk \leftarrow_{\$} \mathsf{IBE.MSKSp}(pp)$
$mpk \leftarrow \mathsf{IBE.MPK}(pp, msk)$
$b' \leftarrow_{\$} A^{\mathrm{sID,KD,LR}}(pp, mpk)$
Return $(b = b')$

proc sID($u$)
If $(u^* \neq \perp)$ then Return $\perp$
$u^* \leftarrow u$
Return $mpk$

proc KD($\phi, u$)
If $(u^* = \perp)$ then Return $\perp$
If $(\phi \notin \Phi)$ then Return $\perp$
$msk' \leftarrow \phi(msk)$
If $(u = u^*)$ then $U \leftarrow U \cup \{msk'\}$
If $(msk^* \in U)$ then Return $\perp$
Return $\mathsf{IBE.UKg}(pp, msk', u)$

proc ENC($\phi, u, m$)
If $(u^* = \perp)$ then Return $\perp$
If $(\phi \notin \Phi)$ then Return $\perp$
$msk' \leftarrow \phi(msk)$
$mpk' \leftarrow \mathsf{IBE.MPK}(pp, msk')$
Return $\mathsf{IBE.Enc}(pp, mpk', u, m)$

proc LR($\phi, m_0, m_1$)
If $(u^* = \perp)$ then Return $\perp$
If $(\phi \notin \Phi)$ then Return $\perp$
If $(c^* \neq \perp)$ then Return $\perp$
If $(|m_0| \neq |m_1|)$ then Return $\perp$
$msk^* \leftarrow \phi(msk)$
$mpk^* \leftarrow \mathsf{IBE.MPK}(pp, msk^*)$
If $(msk^* \in U)$ then Return $\perp$
$c^* \leftarrow_{\$} \mathsf{IBE.Enc}(pp, mpk^*, u^*, m_b)$
Return $c^*$
```

Figure 5.16: Game IND-sID-SRKA defining selective-ID strong Φ-RKA security of IBE scheme IBE.

We show that applying the CHK transform to a selective-ID strong Φ-RKA-secure IBE scheme results in a strong Φ-RKA-secure PKE scheme.

Theorem 5.4.2 *Let* IBE *be a selectively strong* Φ*-RKA-secure IBE scheme and* DS *be a one-time strongly secure signature scheme. Then the PKE scheme* $\mathsf{CHK}[\mathsf{IBE},\mathsf{DS}]$ *as defined in Figure 2.12 is strong* Φ*-RKA secure.*

Proof: Let A be a PT adversary playing game IND-SRKA. We build PT adversaries A_1, A_2 such that

$$\mathbf{Adv}^{\text{ind-srka}}_{\mathsf{CHK}[\mathsf{IBE},\mathsf{DS}],\Phi,A}(\lambda) \le 2\mathbf{Adv}^{\text{ot-suf-cma}}_{\mathsf{DS},A_1}(\lambda) + \mathbf{Adv}^{\text{ind-sid-srka}}_{\mathsf{IBE},\Phi,A_2}(\lambda)$$

for all $\lambda \in \mathbb{N}$, from which the theorem follows.

The proof uses the games in Figure 5.17. We will build A_1, A_2 so that for all $\lambda \in \mathbb{N}$ we have

$$\Pr[\mathrm{G}_0^A(\lambda) \text{ sets } \mathsf{bad}] \le \mathbf{Adv}^{\text{ot-suf-cma}}_{\mathsf{DS},A_1}(\lambda) \quad (5.18)$$

$$2\Pr[\mathrm{G}_0^A(\lambda)] - 1 \le \mathbf{Adv}^{\text{ind-sid-srka}}_{\mathsf{IBE},\Phi,A_2}(\lambda)\ . \quad (5.19)$$

Games IND-SRKA$_{\mathsf{CHK}[\mathsf{IBE},\mathsf{DS}],\Phi}$ and G_0 are identical until bad, so by the Fundamental Lemma of Game-Playing [23] and the above, for all $\lambda \in \mathbb{N}$ we have:

$$\begin{aligned}
\mathbf{Adv}^{\text{ind-srka}}_{\mathsf{CHK}[\mathsf{IBE},\mathsf{DS}],\Phi,A}(\lambda) &= 2\Pr[\text{IND-SRKA}^A_{\mathsf{CHK}[\mathsf{IBE},\mathsf{DS}],\Phi}(\lambda)] - 1 \\
&= 2(\Pr[\text{IND-SRKA}^A_{\mathsf{CHK}[\mathsf{IBE},\mathsf{DS}],\Phi}(\lambda)] - \Pr[\mathrm{G}_0^A(\lambda)]) \\
&\qquad + 2\Pr[\mathrm{G}_0^A(\lambda)] - 1 \\
&\le 2\Pr[\mathrm{G}_0^A(\lambda) \text{ sets } \mathsf{bad}] + 2\Pr[\mathrm{G}_0^A(\lambda)] - 1 \\
&\le 2\mathbf{Adv}^{\text{ot-suf-cma}}_{\mathsf{DS},A_1}(\lambda) + \mathbf{Adv}^{\text{ind-sid-srka}}_{\mathsf{IBE},\Phi,A_2}(\lambda)
\end{aligned}$$

as desired. We proceed to the constructions of A_1, A_2. Adversary A_1 against the one-time strong unforgeability of DS behaves as follows:

```
main $\mathrm{IND\text{-}SRKA}^{A}_{\mathsf{CHK[IBE,DS]},\Phi}(\lambda)$ / $\boxed{\mathrm{G}_0^A(\lambda)}$
$b \leftarrow_\$ \{0,1\}$ ; $msk^* \leftarrow \bot$ ; $c^* \leftarrow \bot$ ; $pk^*_{\mathsf{DS}} \leftarrow \bot$ ; $\sigma^*_{\mathsf{DS}} \leftarrow \bot$ ; $c^*_{\mathsf{IBE}} \leftarrow \bot$
$pp_{\mathsf{IBE}} \leftarrow_\$ \mathsf{IBE.Pg}(1^\lambda)$ ; $pp_{\mathsf{DS}} \leftarrow_\$ \mathsf{DS.Pg}(1^\lambda)$
$msk \leftarrow_\$ \mathsf{MSKSp}(pp_{\mathsf{IBE}})$ ; $mpk \leftarrow \mathsf{IBE.MPK}(pp_{\mathsf{IBE}}, msk)$
$b' \leftarrow_\$ A^{\mathrm{Dec,Enc,LR}}((pp_{\mathsf{IBE}}, pp_{\mathsf{DS}}), mpk)$
Return $(b = b')$

proc $\mathrm{Dec}(\phi, c)$
If $(\phi \notin \Phi)$ then Return $\bot$
$msk' \leftarrow \phi(msk)$
If $((msk' = msk^*) \wedge (c = c^*))$ then Return $\bot$
$(pk_{\mathsf{DS}}, c_{\mathsf{IBE}}, \sigma_{\mathsf{DS}}) \leftarrow c$
If $(!\mathsf{DS.Verify}(pp_{\mathsf{DS}}, pk_{\mathsf{DS}}, c_{\mathsf{IBE}}, \sigma_{\mathsf{DS}}))$ then Return $\bot$
If $((pk_{\mathsf{DS}} = pk^*_{\mathsf{DS}}) \wedge ((c_{\mathsf{IBE}}, \sigma_{\mathsf{DS}}) \neq (c^*_{\mathsf{IBE}}, \sigma^*_{\mathsf{DS}})))$ then $\mathsf{bad} \leftarrow \mathsf{true}$ ; $\boxed{\text{Return } \bot}$
$u \leftarrow pk_{\mathsf{DS}}$
$usk \leftarrow_\$ \mathsf{IBE.UKg}(pp_{\mathsf{IBE}}, msk', u)$
Return $\mathsf{IBE.Dec}(pp_{\mathsf{IBE}}, usk, c_{\mathsf{IBE}})$

proc $\mathrm{Enc}(\phi, m)$
If $(\phi \notin \Phi)$ then Return $\bot$
$msk' \leftarrow \phi(msk)$
$pk' \leftarrow \mathsf{IBE.MPK}(pp_{\mathsf{IBE}}, msk')$
$(sk_{\mathsf{DS}}, pk_{\mathsf{DS}}) \leftarrow_\$ \mathsf{DS.Kg}(pp_{\mathsf{DS}})$
$u \leftarrow pk_{\mathsf{DS}}$
$c_{\mathsf{IBE}} \leftarrow_\$ \mathsf{IBE.Enc}(pp_{\mathsf{IBE}}, mpk', u, m)$
$\sigma_{\mathsf{DS}} \leftarrow_\$ \mathsf{DS.Sign}(pp_{\mathsf{DS}}, sk_{\mathsf{DS}}, c_{\mathsf{IBE}})$
Return $(pk_{\mathsf{DS}}, c_{\mathsf{IBE}}, \sigma_{\mathsf{DS}})$

proc $\mathrm{LR}(\phi, m_0, m_1)$
If $(\phi \notin \Phi)$ then Return $\bot$
If $(c^* \neq \bot)$ then Return $\bot$
If $(|m_0| \neq |m_1|)$ then Return $\bot$
$msk^* \leftarrow \phi(msk)$
$pk' \leftarrow \mathsf{IBE.MPK}(pp_{\mathsf{IBE}}, msk^*)$
$(sk^*_{\mathsf{DS}}, pk^*_{\mathsf{DS}}) \leftarrow_\$ \mathsf{DS.Kg}(pp_{\mathsf{DS}})$
$u \leftarrow pk^*_{\mathsf{DS}}$
$c^*_{\mathsf{IBE}} \leftarrow_\$ \mathsf{IBE.Enc}(pp_{\mathsf{IBE}}, mpk^*, u, m)$
$\sigma^*_{\mathsf{DS}} \leftarrow_\$ \mathsf{DS.Sign}(pp_{\mathsf{DS}}, sk^*_{\mathsf{DS}}, c^*_{\mathsf{IBE}})$
$c^* \leftarrow (pk^*_{\mathsf{DS}}, c^*_{\mathsf{IBE}}, \sigma^*_{\mathsf{DS}})$
Return $c^*$
```

Figure 5.17: Games used in the proof of Theorem 5.4.2.

```
$\underline{A_1^{\textsc{Sign}}(pp_{\mathsf{DS}}, pk^*_{\mathsf{DS}})}$
$(m^*, \sigma^*) \leftarrow \perp$
$b \leftarrow\$ \{0,1\}$ ; $msk^* \leftarrow \perp$ ; $c^* \leftarrow \perp$
$pk^*_{\mathsf{DS}} \leftarrow \perp$ ; $\sigma^*_{\mathsf{DS}} \leftarrow \perp$ ; $c^*_{\mathsf{IBE}} \leftarrow \perp$
$pp_{\mathsf{IBE}} \leftarrow\$ \mathsf{IBE.Pg}(1^\lambda)$
$msk \leftarrow\$ \mathsf{MSKSp}(pp_{\mathsf{IBE}})$
$mpk \leftarrow \mathsf{IBE.MPK}(pp_{\mathsf{IBE}}, msk)$
$b' \leftarrow\$ A^{\textsc{Dec},\textsc{Enc},\mathrm{LR}}((pp_{\mathsf{IBE}}, pp_{\mathsf{DS}}), mpk)$
Return $(m^*, \sigma^*)$

$\underline{\textsc{Dec}(\phi, c)}$
If $(\phi \notin \Phi)$ then Return $\perp$
$msk' \leftarrow \phi(msk)$
If $((msk' = msk^*) \wedge (c = c^*))$
    then Return $\perp$
$(pk_{\mathsf{DS}}, c_{\mathsf{IBE}}, \sigma_{\mathsf{DS}}) \leftarrow c$
If $(!\mathsf{DS.Verify}(pp_{\mathsf{DS}}, pk_{\mathsf{DS}}, c_{\mathsf{IBE}}, \sigma_{\mathsf{DS}}))$
    then Return $\perp$
If $((pk_{\mathsf{DS}} = pk^*_{\mathsf{DS}}) \wedge ((c_{\mathsf{IBE}}, \sigma_{\mathsf{DS}}) \neq (c^*_{\mathsf{IBE}}, \sigma^*_{\mathsf{DS}})))$
    then $(m^*, \sigma^*) \leftarrow (c_{\mathsf{IBE}}, \sigma_{\mathsf{DS}})$ ; Return $\perp$
$u \leftarrow pk_{\mathsf{DS}}$
$usk \leftarrow\$ \mathsf{IBE.UKg}(pp_{\mathsf{IBE}}, msk', u)$
Return $\mathsf{IBE.Dec}(pp_{\mathsf{IBE}}, usk, c_{\mathsf{IBE}})$

$\underline{\textsc{Enc}(\phi, m)}$
If $(\phi \notin \Phi)$ then Return $\perp$
$msk' \leftarrow \phi(msk)$
$pk' \leftarrow \mathsf{IBE.MPK}(pp_{\mathsf{IBE}}, msk')$
$(sk_{\mathsf{DS}}, pk_{\mathsf{DS}}) \leftarrow\$ \mathsf{DS.Kg}(pp_{\mathsf{DS}})$
$u \leftarrow pk_{\mathsf{DS}}$
$c_{\mathsf{IBE}} \leftarrow\$ \mathsf{IBE.Enc}(pp_{\mathsf{IBE}}, mpk', u, m)$
$\sigma_{\mathsf{DS}} \leftarrow\$ \mathsf{DS.Sign}(pp_{\mathsf{DS}}, sk_{\mathsf{DS}}, c_{\mathsf{IBE}})$
Return $(pk_{\mathsf{DS}}, c_{\mathsf{IBE}}, \sigma_{\mathsf{DS}})$

$\underline{\mathrm{LR}(\phi, m_0, m_1)}$
If $(\phi \notin \Phi)$ then Return $\perp$
If $(c^* \neq \perp)$ then Return $\perp$
If $(|m_0| \neq |m_1|)$ then Return $\perp$
$msk^* \leftarrow \phi(msk)$
$pk' \leftarrow \mathsf{IBE.MPK}(pp, msk^*)$
$u \leftarrow pk^*_{\mathsf{DS}}$
$c^*_{\mathsf{IBE}} \leftarrow\$ \mathsf{IBE.Enc}(pp_{\mathsf{IBE}}, mpk^*, u, m_b)$
$\sigma^*_{\mathsf{DS}} \leftarrow\$ \textsc{Sign}(c^*_{\mathsf{IBE}})$
$c^* \leftarrow (pk^*_{\mathsf{DS}}, c^*_{\mathsf{IBE}}, \sigma^*_{\mathsf{DS}})$
Return $c^*$
```

Adversary A_1 simulates game G_0 for A, using its challenge public key in creating the challenge ciphertext for A. Since Sign is only called when A calls LR and $c^* = \perp$, and this situation can occur only on A's first call to LR, A_1 makes at most one call to its Sign oracle. The forgery that A_1 outputs is valid exactly when the flag bad is set in game G_0. The "If" statements before the forgery (m^*, σ^*) is set ensure this. The third "If" statement ensures the signature is valid, and the fourth ensures the pair differs from that output by the signing oracle. This establishes Equation (5.18).

Adversary A_2 against the selective-ID strong Φ-RKA security of IBE behaves as follows:

```
$A_2^{\mathrm{sID,KD,Enc,LR}}(pp_{\mathsf{IBE}})$
$c^* \leftarrow \perp$ ; $pk^*_{\mathsf{DS}} \leftarrow \perp$ ; $\sigma^*_{\mathsf{DS}} \leftarrow \perp$ ; $c^*_{\mathsf{IBE}} \leftarrow \perp$
$pp_{\mathsf{DS}} \leftarrow\$ \mathsf{DS.Pg}(1^\lambda)$
$(sk^*_{\mathsf{DS}}, pk^*_{\mathsf{DS}}) \leftarrow\$ \mathsf{DS.Kg}(pp_{\mathsf{DS}})$
$u^* \leftarrow pk^*_{\mathsf{DS}}$
$mpk \leftarrow \mathrm{sID}(u^*)$
$b' \leftarrow\$ A^{\mathrm{Dec,EncSim,LRSim}}((pp_{\mathsf{IBE}}, pp_{\mathsf{DS}}), mpk)$
Return $b'$

$\mathrm{Dec}(\phi, c)$
If $(\phi \notin \Phi)$ then Return $\perp$
$(pk_{\mathsf{DS}}, c_{\mathsf{IBE}}, \sigma_{\mathsf{DS}}) \leftarrow c$
If $(!\mathsf{DS.Verify}(pp_{\mathsf{DS}}, pk_{\mathsf{DS}}, c_{\mathsf{IBE}}, \sigma_{\mathsf{DS}}))$
   then Return $\perp$
If $((pk_{\mathsf{DS}} = pk^*_{\mathsf{DS}}) \wedge ((c_{\mathsf{IBE}}, \sigma_{\mathsf{DS}}) \neq (c^*_{\mathsf{IBE}}, \sigma^*_{\mathsf{DS}})))$
   then $(m^*, \sigma^*) \leftarrow (c_{\mathsf{IBE}}, \sigma_{\mathsf{DS}})$ ; Return $\perp$
$u \leftarrow pk_{\mathsf{DS}}$
$usk \leftarrow\$ \mathrm{KD}(\phi, u)$
If $(usk = \perp)$ then Return $\perp$
Return $\mathsf{IBE.Dec}(pp_{\mathsf{IBE}}, usk, c_{\mathsf{IBE}})$

$\mathrm{EncSim}(\phi, m)$
If $(\phi \notin \Phi)$ then Return $\perp$
$(sk_{\mathsf{DS}}, pk_{\mathsf{DS}}) \leftarrow\$ \mathsf{DS.Kg}(pp_{\mathsf{DS}})$
$u \leftarrow pk_{\mathsf{DS}}$
$c_{\mathsf{IBE}} \leftarrow\$ \mathrm{Enc}(\phi, u, m)$
$\sigma_{\mathsf{DS}} \leftarrow\$ \mathsf{DS.Sign}(pp_{\mathsf{DS}}, sk_{\mathsf{DS}}, c_{\mathsf{IBE}})$
Return $(pk_{\mathsf{DS}}, c_{\mathsf{IBE}}, \sigma_{\mathsf{DS}})$

$\mathrm{LRSim}(\phi, m_0, m_1)$
If $(\phi \notin \Phi)$ then Return $\perp$
If $(c^* \neq \perp)$ then Return $\perp$
If $(|m_0| \neq |m_1|)$ then Return $\perp$
$c^*_{\mathsf{IBE}} \leftarrow\$ \mathrm{LR}(\phi, m_0, m_1)$
$\sigma^*_{\mathsf{DS}} \leftarrow\$ \mathsf{DS.Sign}(pp_{\mathsf{DS}}, sk^*_{\mathsf{DS}}, c^*_{\mathsf{IBE}})$
$c^* \leftarrow (pk^*_{\mathsf{DS}}, c^*_{\mathsf{IBE}}, \sigma^*_{\mathsf{DS}})$
Return $c^*$
```

Adversary A_2 simulates game G_0 for A, choosing up-front the public verification key that will determine the identity used in the challenge ciphertext. Adversary A_2 must return $\perp$ when A makes a decryption query with $(\phi(msk) = msk^*) \wedge (c = c^*)$, without knowing msk or msk^*. If the ciphertext A submits for decryption is the challenge ciphertext, then the identity submitted to A_2's KD oracle is the challenge identity, and if $\phi(msk) = msk^*$ then the KD oracle will return $\perp$, so A_2 detects this case and returns $\perp$ as required.

The challenge identity will never otherwise be queried to the KD oracle by Dec as the third "If" statement will cause $\perp$ to be returned before the KD oracle is called. Adversary A_2 outputs A's guess as its own, winning whenever A does, establishing Equation (5.19). ▌

5.4.3 Strong RKA-Secure IBE

Finally, we show how to build strong Φ-RKA-secure IBE using the techniques of Chapter 4 and some additional properties of the IBE scheme. Our schemes will satisfy the adaptive notion of strong Φ-RKA security given in Figure 5.18, which implies the selective notion of the previous section. We say an IBE scheme IBE

```
main IND-SRKA$^A_{\mathsf{IBE},\Phi}(\lambda)$
$b \leftarrow_\$ \{0,1\}$ ; $c^* \leftarrow \perp$ ; $u^* \leftarrow \perp$ ; $msk^* \leftarrow \perp$ ; $U \leftarrow \emptyset$
$pp \leftarrow_\$ \mathsf{IBE.Pg}(1^\lambda)$
$msk \leftarrow_\$ \mathsf{IBE.MSKSp}(pp)$
$mpk \leftarrow \mathsf{IBE.MPK}(pp, msk)$
$b' \leftarrow_\$ A^{\mathrm{KD},\mathrm{LR}}(pp, mpk)$
Return $(b = b')$

proc KD$(\phi, u)$
If $(\phi \notin \Phi)$ then Return $\perp$
$msk' \leftarrow \phi(msk)$
$U \leftarrow U \cup \{(u, msk')\}$
If $((u^*, msk^*) \in U)$ then Return $\perp$
Return $\mathsf{IBE.UKg}(pp, msk', u)$

proc Enc$(\phi, u, m)$
If $(\phi \notin \Phi)$ then Return $\perp$
$msk' \leftarrow \phi(msk)$
$mpk' \leftarrow \mathsf{IBE.MPK}(pp, msk')$
Return $\mathsf{IBE.Enc}(pp, mpk', u, m)$

proc LR$(\phi, u, m_0, m_1)$
If $(c^* \neq \perp)$ then Return $\perp$
If $(|m_0| \neq |m_1|)$ then Return $\perp$
$u^* \leftarrow u$
$msk^* \leftarrow \phi(msk)$
If $((u^*, msk^*) \in U)$ then Return $\perp$
$mpk^* \leftarrow \mathsf{IBE.MPK}(pp, msk^*)$
$c^* \leftarrow_\$ \mathsf{IBE.Enc}(pp, mpk^*, u^*, m_b)$
Return $c^*$
```

Figure 5.18: Game IND-SRKA defining strong Φ-RKA security of IBE scheme IBE.

is *strong* Φ*-RKA secure* if $\mathbf{Adv}^{\text{ind-srka}}_{\mathsf{IBE},\Phi,A}(\cdot)$ is negligible for all PT adversaries A, where $\mathbf{Adv}^{\text{ind-srka}}_{\mathsf{IBE},\Phi,A}(\lambda) = 2\Pr[\text{IND-SRKA}^{A}_{\mathsf{IBE},\Phi}(\lambda)] - 1$ and game IND-SRKA is in Figure 5.18.

Master public key malleability. In Section 4.4 we showed how to use an identity renaming transform to construct an RKA-secure IBE scheme from an adaptively secure 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 IBE with parameters pp is Φ-mpk-malleable if there exists an algorithm R and a function $f \in \mathsf{Fun}(\mathsf{MSp}(pp), \mathsf{MSp}(pp))$ such that

$$\begin{aligned}&\mathsf{R}(pp, \mathsf{IBE.Enc}(pp, \mathsf{IBE.MPK}(pp, msk), u, f(m); t), \phi)\\ &\quad = \mathsf{IBE.Enc}(pp, \mathsf{IBE.MPK}(pp, \phi(msk)), u, m; t)\end{aligned}$$

for all $msk \in \mathsf{MSKSp}(pp)$, $u \in \mathsf{USp}(pp)$, $m \in \mathsf{MSp}(pp)$, $\phi \in \Phi$, and random coins t. This property says that we can take a ciphertext encrypting a message $f(m)$ related to m under master public key $\mathsf{IBE.MPK}(pp, msk)$ and use it to build a ciphertext encrypting m under master public key $\mathsf{IBE.MPK}(pp, \phi(msk))$, for the same user u and under the same randomness, t.

Theorem 5.4.3 *Let* $\mathsf{IBE} = (\mathsf{Pg}, \mathsf{MKg}, \mathsf{UKg}, \mathsf{Enc}, \mathsf{Dec})$ *be a* Φ*-key-malleable and* Φ*-mpk-malleable adaptively secure IBE scheme with functions* R *and* f *and key simulator* T*. Let* (SI, PI) *be a statistically collision-resistant renaming scheme. Let* $\overline{\mathsf{IBE}} = (\mathsf{Pg}, \mathsf{MKg}, \overline{\mathsf{UKg}}, \overline{\mathsf{Enc}}, \mathsf{Dec})$ *be obtained from* IBE *and renaming scheme* (SI, PI) *via the transform* **IRT** *described in Section 4.4. Then* $\overline{\mathsf{IBE}}$ *is strong* Φ*-RKA secure.*

Proof: The proof is very similar to that of Theorem 4.4.1, and uses the games in Figure 5.19 and Figure 5.20. KD of game G_0 moves the identity renaming up before the list of queried identities is updated and then adds the transformed identity to a list. LR is likewise modified so its test now involves the transformed (rather than original) identities. Additionally, the secret renaming function SI is used instead of PI, a modification allowed by the compatibility property of the renaming scheme. We claim this makes no difference, meaning

$$\Pr[\text{IND-SRKA}^{\overline{A}}_{\overline{\mathsf{IBE}},\Phi}(\lambda)] = \Pr[\text{G}_0^{\overline{A}}(\lambda)]\ .$$

Indeed, statistical collision-resistance tell us that

$$(msk', \overline{u}) = (msk^*, \overline{u}^*) \text{ iff } \text{SI}(pp, msk', \overline{u}) = \text{SI}(pp, msk^*, \overline{u}^*)\ .$$

```
MAIN $\dbox{\text{IND-SRKA}^{\overline{A}}_{\mathsf{IBE}}(\lambda)}$ / $\boxed{G_0^{\overline{A}}(\lambda) \;/\; G_1^{\overline{A}}(\lambda) \;/\; G_2^{\overline{A}}(\lambda) \;/\; G_3^{\overline{A}}(\lambda)}$
$b \leftarrow_{\$} \{0,1\}$ ; $c^* \leftarrow \perp$ ; $\dbox{\overline{u}^* \leftarrow \perp \;;\; msk^* \leftarrow \perp \;;\; \overline{U} \leftarrow \emptyset}$ $\boxed{u^* \leftarrow \perp \;;\; U \leftarrow \emptyset}$
$pp \leftarrow_{\$} \mathsf{IBE.Pg}(1^\lambda)$ ; $msk \leftarrow_{\$} \mathsf{IBE.MSKSp}(pp)$ ; $mpk \leftarrow \mathsf{IBE.MPK}(pp, msk)$
$b' \leftarrow_{\$} \overline{A}^{\text{KD,LR}}(pp, mpk)$
Return $(b = b')$

proc KD$(\phi, \overline{u})$  // $\dbox{\text{IND-SRKA}^{\overline{A}}_{\mathsf{IBE}}(\lambda)}$ / $\boxed{G_0^{\overline{A}}(\lambda)}$
If $(\phi \notin \Phi)$ then Return $\perp$
$msk' \leftarrow \phi(msk)$
$\dbox{\overline{U} \leftarrow \overline{U} \cup \{(\overline{u}, msk')\} \;;\; \text{If } ((\overline{u}^*, msk^*) \in \overline{U}) \text{ then Return } \perp}$
$u \leftarrow \mathrm{SI}(pp, msk', \overline{u})$
$\boxed{U \leftarrow U \cup \{u\} \;;\; \text{If } (u^* \in U) \text{ then Return } \perp}$
Return $\mathsf{IBE.UKg}(pp, msk', u)$

proc ENC$(\phi, \overline{u}, m)$  // $\dbox{\text{IND-SRKA}^{\overline{A}}_{\mathsf{IBE}}(\lambda) \;/\; G_0^{\overline{A}}(\lambda)}$ / $\boxed{G_1^{\overline{A}}(\lambda)}$
If $(\phi \notin \Phi)$ then Return $\perp$
$msk' \leftarrow \phi(msk)$ ; $mpk' \leftarrow \mathsf{IBE.MPK}(pp, msk')$
$\dbox{u \leftarrow \mathrm{PI}(pp, mpk', \overline{u}, \mathsf{id})}$ $\boxed{u \leftarrow \mathrm{PI}(pp, mpk, \overline{u}, \phi)}$
Return $\mathsf{IBE.Enc}(pp, mpk', u, m)$

proc LR$(\phi, \overline{u}, m_0, m_1)$  // $\dbox{\text{IND-SRKA}^{\overline{A}}_{\mathsf{IBE}}(\lambda)}$ / $\boxed{G_0^{\overline{A}}(\lambda)}$
If $(\phi \notin \Phi)$ then Return $\perp$
If $(c^* \neq \perp)$ then Return $\perp$
If $(|m_0| \neq |m_1|)$ then Return $\perp$
$\dbox{\overline{u}^* \leftarrow \overline{u} \;;\; msk^* \leftarrow \phi(msk) \;;\; \text{If } ((\overline{u}^*, msk^*) \in \overline{U}) \text{ then Return } \perp}$
$\dbox{mpk^* \leftarrow \mathsf{IBE.MPK}(pp, msk^*) \;;\; u^* \leftarrow \mathrm{PI}(pp, mpk^*, \overline{u}^*, \mathsf{id})}$
$\boxed{msk^* \leftarrow \phi(msk) \;;\; mpk^* \leftarrow \mathsf{IBE.MPK}(pp, msk^*) \;;\; u^* \leftarrow \mathrm{SI}(pp, msk^*, \overline{u})}$
$\boxed{\text{If } (u^* \in U) \text{ then Return } \perp}$
$c^* \leftarrow_{\$} \mathsf{IBE.Enc}(pp, mpk^*, u^*, m_b)$
Return $c^*$
```

Figure 5.19: Games used in the proof of Theorem 5.4.3.

```
proc KD($\phi, \overline{u}$)   // $\boxed{G_1^{\overline{A}}(\lambda) \ / \ G_2^{\overline{A}}(\lambda)}$ / $G_3^{\overline{A}}(\lambda)$
If ($\phi \notin \Phi$) then Return $\perp$
$u \leftarrow \mathrm{PI}(pp, mpk, \overline{u}, \phi)$
$U \leftarrow U \cup \{u\}$
If ($u^* \in U$) then Return $\perp$
$\boxed{\text{Return } \mathsf{IBE.UKg}(pp, \phi(msk), u)}$
$usk \leftarrow \mathsf{IBE.UKg}(pp, msk, u)$
Return $T(pp, mpk, u, usk, \phi)$

proc Enc($\phi, \overline{u}, m$)   // $G_2^{\overline{A}}(\lambda)$ / $G_3^{\overline{A}}(\lambda)$
If ($\phi \notin \Phi$) then Return $\perp$
$u \leftarrow \mathrm{PI}(pp, mpk, \overline{u}, \phi)$
Return $\mathsf{R}(pp, \mathsf{IBE.Enc}(pp, mpk, u, f(m)), \phi)$

proc LR($\phi, \overline{u}, m_0, m_1$)   // $\boxed{G_1^{\overline{A}}(\lambda)}$ / $G_2^{\overline{A}}(\lambda) \ / \ G_3^{\overline{A}}(\lambda)$
If ($\phi \notin \Phi$) then Return $\perp$
If ($c^* \neq \perp$) then Return $\perp$
If ($|m_0| \neq |m_1|$) then Return $\perp$
$\boxed{msk^* \leftarrow \phi(msk) \ ; \ mpk^* \leftarrow \mathsf{IBE.MPK}(pp, msk^*)}$
$u^* \leftarrow \mathrm{PI}(pp, mpk, \overline{u}, \phi)$
If ($u^* \in U$) then Return $\perp$
$\boxed{c^* \leftarrow_{\$} \mathsf{IBE.Enc}(pp, mpk^*, u^*, m_b)}$
$c^* \leftarrow \mathsf{R}(pp, \mathsf{IBE.Enc}(pp, mpk, u^*, f(m_b)), \phi)$
Return $c^*$
```

Figure 5.20: Games used in the proof of Theorem 5.4.3, continued.

This means that the dashed-boxed code of IND-SRKA$_{\overline{\mathsf{IBE}},\Phi}$ and the boxed code of G_0 are equivalent.

Compatibility is invoked to use PI in place of SI in both $\textsc{Enc}$ and in LR in G_1, so that

$$\Pr[\mathrm{G}_0^{\overline{A}}(\lambda)] = \Pr[\mathrm{G}_1^{\overline{A}}(\lambda)] \ .$$

Φ-mpk-malleability is invoked to use $\mathsf{R}(pp, \mathsf{Enc}(pp, \mathsf{MPK}(pp, msk), u, f(m)), \phi)$ in place of $\mathsf{Enc}(pp, \mathsf{MPK}(pp, \phi(msk)), u, m)$ in G_2, so that

$$\Pr[\mathrm{G}_1^{\overline{A}}(\lambda)] = \Pr[\mathrm{G}_2^{\overline{A}}(\lambda)] \ .$$

Rather than use $\phi(msk)$ for key generation as in the boxed code of KD of G_2, G_3 uses msk and then applies the key simulator T. We claim that key-malleability implies

$$\Pr[\mathrm{G}_2^{\overline{A}}(\lambda)] = \Pr[\mathrm{G}_3^{\overline{A}}(\lambda)] \ . \tag{5.20}$$

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

$$\Pr[\mathrm{KMReal}_{\mathsf{IBE},\Phi}^{M}(\lambda)] = \Pr[\mathrm{G}_2^{\overline{A}}(\lambda)] \quad \text{and} \quad \Pr[\mathrm{KMSim}_{\mathsf{IBE},\Phi,T}^{M}(\lambda)] = \Pr[\mathrm{G}_3^{\overline{A}}(\lambda)] \ .$$

Adversary M behaves as follows:

```
$M^{\mathrm{KD}}(pp, mpk)$
$b \leftarrow\$ \{0,1\}$ ; $c^* \leftarrow \perp$
$u^* \leftarrow \perp$ ; $U \leftarrow \emptyset$
$b' \leftarrow\$ \overline{A}^{\mathrm{KDSim},\textsc{Enc},\mathrm{LR}}(pp, mpk)$
Return $(b = b')$

$\mathrm{KDSim}(\phi, \overline{u})$
$u \leftarrow \mathrm{PI}(pp, mpk, \overline{u}, \phi)$
$U \leftarrow U \cup \{u\}$
If $(u^* \in U)$ then Return $\perp$
Return $\mathrm{KD}(\phi, u)$

$\textsc{Enc}(\phi, \overline{u}, m)$
If $(\phi \notin \Phi)$ then Return $\perp$
$u \leftarrow \mathrm{PI}(pp, mpk, \overline{u}, \phi)$
Return $\mathsf{R}(pp, \mathsf{IBE.Enc}(pp, mpk, u, f(m)), \phi)$

$\mathrm{LR}(\phi, \overline{u}, m_0, m_1)$
If $(c^* \neq \perp)$ then Return $\perp$
$u^* \leftarrow \mathrm{PI}(pp, mpk, \overline{u}, \phi)$
If $(u^* \in U)$ then Return $\perp$
$c^* \leftarrow\$ \mathsf{R}(pp, \mathsf{IBE.Enc}(pp, mpk, u^*, f(m_b)), \phi)$
Return $c^*$
```

If M is playing game KMReal then its KD oracle will behave as the boxed code in game G_2, while if M is playing game KMSim its KD oracle will behave as in game G_3. 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 IBE Equation (5.20) holds.

Finally, we design A so that

$$\mathbf{Adv}^{\text{ind-aid}}_{\mathsf{IBE},A}(\lambda) = 2\Pr[G_3^{\overline{A}}(\lambda)] - 1 \, .$$

Adversary A behaves as follows:

```
$\underline{A^{\mathrm{KD,LR}}(pp, mpk)}$
$b' \leftarrow_{\$} \overline{A}^{\mathrm{KDSim,Enc,LRSim}}(pp, mpk)$
Return $b'$

$\underline{\mathrm{Enc}(\phi, \overline{u}, m)}$
If $(\phi \notin \Phi)$ then Return $\perp$
$u \leftarrow \mathrm{PI}(pp, mpk, \overline{u}, \phi)$
Return $\mathsf{R}(pp, \mathsf{IBE.Enc}(pp, mpk, u, f(m)), \phi)$

$\underline{\mathrm{KDSim}(\phi, \overline{u})}$
$u \leftarrow \mathrm{PI}(pp, mpk, \overline{u}, \phi)$
$usk \leftarrow \mathrm{KD}(u)$
Return $T(pp, mpk, u, usk, \phi)$

$\underline{\mathrm{LRSim}(\phi, \overline{u}, m_0, m_1)}$
$u^* \leftarrow \mathrm{PI}(pp, mpk, \overline{u}, \phi)$
$c \leftarrow \mathrm{LR}(u^*, f(m_0), f(m_1))$
Return $\mathsf{R}(pp, c, \phi)$
```

Adversary A perfectly simulates G_3 for $\overline{A}$, winning whenever $\overline{A}$ does. ▌

Waters' IBE scheme is Φ-mpk-malleable for affine ϕ with $a \neq 0$. The restriction that the linear coefficient be non-zero is necessary, as otherwise the master secret key becomes the known value b. 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:

```
$\mathsf{R}(pp, (c'_1, c'_2, c'_3), \phi_{a,b})$:
    $c_1 \leftarrow c'_1$ ; $c_2 \leftarrow c'_2$
    $c_3 \leftarrow (c'_3)^a \cdot e((c'_1)^b, g_1)$
    Return $(c_1, c_2, c_3)$
```

Let

$$c \leftarrow \mathsf{Enc}(pp, \mathsf{MPK}(pp, msk), u, m^{a^{-1}}; t) = (g^t, \mathsf{H}(u)^t, e(\mathsf{MPK}(pp, msk), g_1)^t \cdot m^{a^{-1}})$$

be the ciphertext input to R. Then

$$\begin{aligned}
c_3 &= (c_3')^a \cdot e((c_1')^b, g_1)\\
&= (e(g^{msk}, g_1)^t \cdot m^{a^{-1}})^a \cdot e((g^t)^b, g_1)\\
&= e(g, g_1)^{a \cdot msk \cdot t} \cdot m \cdot e(g, g_1)^{bt}\\
&= e(g, g_1)^{(a \cdot msk + b)t} \cdot m\\
&= e(g^{(a \cdot msk + b)}, g_1)^t \cdot m\\
&= e(\mathsf{MPK}(\phi(pp, msk)), g_1)^t \cdot m \ ,
\end{aligned}$$

so

$$\begin{aligned}
(c_1, c_2, c_3) &= (g^t, \mathsf{H}(u)^t, e(\mathsf{MPK}(pp, \phi(msk)), g_1)^t \cdot m)\\
&= \mathsf{Enc}(pp, \mathsf{MPK}(pp, \phi(msk)), u, m; t)
\end{aligned}$$

and R's output is correct.

The extended Waters IBE scheme is Φ-mpk-malleable for polynomial $\phi_{a_0,a_1,\dots,a_d}$ 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:

```
$\mathsf{R}(pp, (c_1', c_2', c_3'), \phi_{a_0,a_1,\dots,a_d})$:
    $c_1 \leftarrow c_1'$ ; $c_2 \leftarrow c_2'$
    $c_3 \leftarrow (c_3')^{a_1} \cdot e((c_1')^{a_0}, g_1) \cdot e((c_1')^{a_2}, (g_1^{s^2})) \cdots e((c_1')^{a_d}, (g_1^{s^d}))$
    Return $(c_1, c_2, c_3)$
```

Let

$$c \leftarrow \mathsf{Enc}(pp, \mathsf{MPK}(pp, msk), u, m^{a_1^{-1}}; t) = (g^t, \mathsf{H}(u)^t, e(\mathsf{MPK}(pp, msk), g_1)^t \cdot m^{a_1^{-1}})$$

be the ciphertext input to R. Then

$$\begin{aligned}
c_3 &= (c_3')^{a_1} \cdot e((c_1')^{a_0}, g_1) \cdot e((c_1')^{a_2}, (g_1^{msk^2})) \cdots e((c_1')^{a_d}, (g_1^{msk^d}))\\
&= (e(g^{msk}, g_1)^t \cdot m^{a_1^{-1}})^{a_1} \cdot e((g^t)^{a_0}, g_1) \cdot e((g^t)^{a_2}, (g_1^{msk^2})) \cdots e((g^t)^{a_d}, (g_1^{msk^d}))\\
&= e(g, g_1)^{a_1 \cdot msk \cdot t} \cdot m \cdot e(g, g_1)^{a_0 t} \cdot e(g, g_1)^{a_2 \cdot msk^2 t} \cdots e(g, g_1)^{a_d \cdot msk^d t}\\
&= e(g, g_1)^{(a_0 + a_1 \cdot msk + a_2 \cdot msk^2 + \cdots + a_d \cdot msk^d)t} \cdot m\\
&= e(g^{(a_0 + a_1 \cdot msk + a_2 \cdot msk^2 + \cdots + a_d \cdot msk^d)}, g_1)^t \cdot m\\
&= e(\mathsf{MPK}(pp, \phi(msk)), g_1)^t \cdot m \ ,
\end{aligned}$$

so

$$\begin{aligned}
(c_1, c_2, c_3) &= (g^t, \mathsf{H}(u)^t, e(\mathsf{MPK}(pp, \phi(msk)), g_1)^t \cdot m)\\
&= \mathsf{Enc}(pp, \mathsf{MPK}(pp, \phi(msk)), u, m; t)
\end{aligned}$$

and R's output is correct.

If the linear coefficient of the RKD function is zero, then it is possible to compute the mask $e(g^{(a_0+a_2\cdot msk^2+\cdots+a_d\cdot msk^d)}, g_1)^t$ by pairing g^t and the $g_1^{msk^i}$ values, so the IND-aID security of the extended Waters scheme means it cannot be Φ-mpk-malleable for Φ including polynomials with zero linear coefficients. Strong RKA security is unachievable for constant RKD functions $\phi(msk) = a_0$ 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.

Constructing a symmetric encryption scheme 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 4.4 gives Φ-RKA-secure SE for affine and polynomial ϕ with non-zero linear coefficients. In [69] a CPA secure SE scheme RKA secure against RKD functions consisting of polynomials of non-zero degree is presented, while our SE scheme is RKA secure against the more restricted RKD set consisting of polynomials with non-zero linear coefficients. Note however that the construction of [69] relies on a specific hardness assumption rather than arising from a general framework as our instantiations do. Moreover, our schemes are CCA secure, whereas the scheme of [69] is only CPA secure.

5.5 Conclusion

We showed that the Boneh-Katz transform preserves RKA security, giving Φ-RKA-secure PKE from the Φ-RKA-secure IBE schemes of the previous chapter. Previously [14] showed the CHK transform has the same property, however the Boneh-Katz transform is more efficient.

We also constructed an RKA-secure KEM based on the KEM of Boyen, Mei and Waters [39] that is secure against an adversary applying affine transformations to the key. We showed that the KEM-DEM composition theorem holds in the RKA setting, leading to further RKA-secure PKE schemes.

We showed that our joint encryption and signature scheme of Section 3.5 preserves

RKA security, giving a scheme secure when the same key is used for both encryption and signature, and the adversary can additionally obtain signatures and decryptions under related keys.

We then turned our attention to symmetric encryption, extending the results of the previous chapter to show that when the base IBE scheme has a further malleability property, the PKE scheme obtained through the CHK transform can be converted into an RKA-secure CCA-SE (CCA-secure symmetric encryption) scheme. Instantiating the scheme with our affine and polynomial RKA-secure IBE schemes of the previous chapter gives the first symmetric encryption schemes secure against affine and polynomial RKAs in the CCA setting, such schemes having already been constructed in the CPA setting in [69]. In the CPA setting [69] gives a symmetric encryption scheme RKA secure against the class $\Phi^{\mathrm{poly}(d)} \setminus \Phi^c$, while we are able to achieve RKA security in the CCA setting against the set of polynomials of degree d with linear coefficient zero. Closing this gap and achieving RKA security against the full set $\Phi^{\mathrm{poly}(d)} \setminus \Phi^c$ in the CCA setting is an open problem.

We have achieved RKA security under affine and polynomial key transformations for many primitives for which previously constructions secure against only linear RKAs were known. An interesting direction for future research is to construct schemes secure against further classes of RKA such as those described by bit-flipping, shifting, and masking, which are arguably more natural transformations to consider in the setting of tampering attacks.

Chapter 6

Key-Versatile Signatures

Contents

This chapter introduces key-versatile signatures. Key-versatile signatures allow us to sign with keys already in use for another purpose, without changing the keys and without impacting the security of the original purpose. This allows us to obtain advances across a collection of challenging domains including joint encryption and signature, security against related-key attack (RKA) and security for key-dependent messages (KDM). Specifically we can (1) Add signing capability to existing encryption capability with zero overhead in the size of the public key (2) Obtain RKA-secure signatures from any RKA-secure one-way function, yielding new RKA-secure signature schemes (3) Add integrity to encryption while maintaining KDM-security.

6.1 Introduction

One of the recommended principles of sound cryptographic design is key separation, meaning that keys used for one purpose (e.g. encryption) should not be used for another purpose (e.g. signing). The reason is that, as demonstrated in Chapter 3, even if the individual uses are secure, the joint usage could be insecure [53]. This chapter shows, to the contrary, that there are important applications where key reuse is not only desirable but crucial to maintain security, and that when done "right" it works. We offer key-versatile signatures as a general tool to enable signing with existing keys already in use for another purpose, without adding key material and while maintaining security of both the new and the old usage of the keys. Our applications include: (1) adding signing capability to existing encryption capability with zero overhead in the size of the public key (2) obtaining RKA-secure signatures from RKA-secure one-way functions (3) adding integrity to encryption while preserving KDM security.

Closer look. Key-versatility refers to the ability to take an arbitrary one-way function F and return a signature scheme where the secret signing key is a random domain point x for F and the public verification key is its image $y = F(x)$. By requiring strong simulatability and key-extractability security conditions [46] from these "F-keyed" signatures, and then defining F based on keys already existing for another purpose, we will be able to add signing capability while maintaining existing keys and security.

The most compelling motivation comes from security against related-key attack and security for key-dependent messages (KDM), technically challenging areas where solutions create, and depend on, very specific key structures. We would like to expand the set of primitives for which we can provide these forms of security. Rather than start from scratch, we would like to leverage the existing, hard-won advances in these areas by modular design, transforming a primitive X into a primitive Y while preserving RKA or KDM security. Since security is relative to a set of functions (either key or message deriving) on the space of keys, the transform must preserve the existing keys. Key-versatile signatures will thus allow us to create new RKA and KDM secure primitives in a modular way.

We warn that our results are theoretical feasibility ones. They demonstrate that certain practical goals can in principle be reached, but the solutions are not efficient. Below we begin with a more direct application of key versatile signatures to joint encryption and signature and then go on to our RKA and KDM results.

Joining signatures to encryption with zero public-key overhead. Suppose Alice has keys $(sk_{\mathsf{PKE}}, pk_{\mathsf{PKE}})$ for a public-key encryption scheme and wants to also have signing capability. Certainly, she could pick new and separate keys $(sk_{\mathsf{DS}}, pk_{\mathsf{DS}})$ enabling her to use her favourite signature scheme. However, it means that Alice's public key, now $pk = (pk_{\mathsf{PKE}}, pk_{\mathsf{DS}})$, has doubled in size. Practitioners ask if one can do better. We want a joint encryption and signature scheme (recall from Chapter 3 a JES scheme has a single keypair (sk, pk) used for both encryption and signing). We aim to minimise the public-key overhead, (loosely) defined as the size of pk minus the size of the public key pk_{PKE} of the underlying encryption scheme.

Previous results in joint encryption and signature pertain to *specific* encryption schemes. We step back to ask a general theoretical question. Namely, suppose we are given an *arbitrary* IND-CCA-secure public-key encryption scheme. We wish to add signing capability to form a JES scheme. How low can the public-key overhead go? The (perhaps surprising) answer we provide is that we can achieve a public-key overhead of *zero*. The public key for our JES scheme remains *exactly* that of the given encryption scheme, meaning we add signing capability without changing the public key. (Zero public-key overhead has a particular advantage besides space savings, namely that, in adding signing, no new certificates are needed. This makes key management significantly easier for the potentially large number of entities already using Alice's public key. This advantage is absent if the public key is at all modified.) We emphasise again that this is for *any* starting encryption scheme.

To do this, we let F be the function that maps the secret key of the given encryption scheme to the public key. (Not all encryption schemes will directly derive the public key as a deterministic function of the secret key, although many, including Cramer-Shoup [49], do. However, we can modify any encryption scheme to have this property, *without changing the public key*, by using the coins of the key-generation algorithm as the secret key.) The assumed security of the encryption scheme means this function is one-way. Now, we simply use an F-keyed signature scheme, with the keys remaining those of the encryption scheme. No new keys are introduced. We need however to ensure that the joint use of the keys does not result in bad interactions that make either the encryption or the signature insecure. This amounts to showing that the JES security conditions, namely that encryption remains secure even given a signing oracle and signing remains secure even given a decryption oracle, are met. This will follow from the simulatability and key-extractability requirements we impose on our F-keyed signatures.

New RKA-secure signatures. Recall from Chapter 4 that in a related-key attack (RKA) [80, 24, 18, 14] an adversary can modify a stored secret key and observe outputs of the cryptographic primitive under the modified key. Achieving proven security against RKAs, however, is broadly recognised as very challenging. This has lead several authors [66, 14] to suggest that we "bootstrap," building higher-level Φ-

RKA-secure primitives from lower-level Φ-RKA-secure primitives. In this vein, [14] show how to build Φ-RKA signatures from Φ-RKA PRFs. Building Φ-RKA PRFs remains difficult, however, and we really have only one construction [13]. This has lead to the direct (non-bootstrapping) constructions in Chapter 5 of Φ-RKA signatures for classes Φ of polynomials over certain specific pairing groups.

We return to bootstrapping and provide a much stronger result, building Φ-RKA signatures from Φ-RKA one-way functions rather than from Φ-RKA PRFs. (For a one-way function, the input is the "key." In attempting to recover x from $F(x)$, the adversary may also obtain $F(x')$ where x' is created by applying to x some modification function from Φ. The definition is from [66].) The difference is significant because building Φ-RKA one-way functions under standard assumptions is easy. Adapting the key-malleability technique of [13], we show that many natural one-way functions are Φ-RKA secure (for appropriate classes Φ) *assuming nothing more than their standard one-wayness.* In particular this is true for discrete exponentiation over an arbitrary group and for the one-way functions underlying the LWE and LPN problems. In this way we obtain Φ-RKA signatures for many new and natural classes Φ.

The central challenge in our bootstrapping is to preserve the keyspace, meaning that the space of secret keys of the constructed signature scheme must be the domain of the given Φ-RKA one-way function F. (Without this, it is not even meaningful to talk of preserving Φ-RKA security, let alone to show that it happens.) This is exactly what an F-keyed signature scheme allows us to do. The proof that Φ-RKA security is preserved exploits strong features built into our definitions of simulatability and key-extractability for F-keyed signatures, in particular that these conditions hold even under secret keys selected by the adversary.

KDM-SECURE STORAGE. Over the last few years we have seen a large number of sophisticated schemes to address the (challenging) problem of encryption of key-dependent data (e.g., [27, 36, 9, 7, 43, 44, 26, 11, 87, 6, 40, 41, 17, 62, 75]). The most touted application is secure outsourced storage, where Alice's decryption key, or some function thereof, is in a file she is encrypting and uploading to the cloud. This can occur for example when making an offsite backup of a disk. But in this setting integrity is just as important as privacy. To this end, we would like to add signatures, thus enabling the server, based on Alice's public key, to validate her uploads, and enabling Alice herself to validate her downloads, all *while preserving KDM security.*

What emerges is a new goal that we call KDM-secure (encrypted and authenticated) storage. In Section 6.5 we formalise the corresponding primitive, providing both syntax and notions of security for key-dependent messages. Briefly, Alice uses a secret key sk to turn her message M into an encrypted and authenticated "data" object that she stores on the server. The server is able to check integrity based on Alice's public key. When Alice retrieves data, she can check integrity and decrypt based on her secret key. Security requires both privacy and integrity even when M depends on sk. (As we explain in more depth below, this goal is different from signcryption [100] and authenticated symmetric encryption [20, 93], even in the absence of KDM considerations.)

A natural approach to achieve our goal is for Alice to encrypt under a symmetric, KDM-secure scheme and sign the ciphertexts under a conventional signature scheme. But it is not clear how to prove the resulting storage scheme is KDM-secure. The difficulty is that sk would include the signing key in addition to the encryption (and decryption) key K, so that messages depend on both these keys while the KDM security of the encryption only covers messages depending on K. We could attempt to start from scratch and design a secure storage scheme meeting our notions. But key-versatile signatures offer a simpler and more modular solution. Briefly, we take a KDM-secure *public-key* encryption scheme and let F be the one-way function that maps a secret key to a public key. Alice holds (only) a secret key sk and the server holds $pk = F(sk)$. To upload M, Alice re-computes pk from sk, encrypts M under it using the KDM scheme, and signs the ciphertext with an F-keyed signature scheme using the *same* key sk. The server verifies signatures under pk.

In Section 6.5 we present in full the construction outlined above, and prove that it meets our notion of KDM security. The crux, as for our RKA-secure constructions, is that adding signing capability without changing the keys puts us in a position to exploit the assumed KDM security of the underlying encryption scheme. The strong simulatability and key-extractability properties of our signatures do the rest. We note that as an added bonus, we assume only CPA-KDM security of the base encryption scheme, yet our storage scheme achieves CCA-KDM security.

CONSTRUCTING F-KEYED SIGNATURES. In Section 6.2 we define F-keyed signatures and show how to construct them for arbitrary one-way F. This enables us to realise the above applications.

Our simulatability condition, adapting [46, 1, 45], asks for a trapdoor allowing the creation of simulated signatures given only the message and public key, even when the secret key underlying this public key is adversarially chosen. Our key-extractability condition, adapting [46], asks that, using the same trapdoor, one can extract from a valid signature the corresponding secret key, even when the public key is adversarially chosen. Theorem 6.2.1, showing these conditions imply not just standard but strong unforgeability, acts not just as a sanity check but as a way to introduce, in a simple form, a proof template that we will extend for our applications.

Our construction of an F-keyed signature scheme is a minor adaptation of a NIZK-based signature scheme of Dodis, Haralambiev, López-Alt and Wichs (DHLW) [56]. While DHLW [56] prove leakage-resilience of their scheme, we prove simulatability and key-extractability. The underlying SE NIZKs (defined in Chapter 2) are a variant of simulation-sound extractable NIZKs [50, 70, 71] introduced by [56] under the name tSE NIZKs and shown by [56, 73] to be achievable for all of **NP** under standard assumptions.

Discussion and related work. F-keyed signatures can be viewed as a special case of signatures of knowledge as introduced by Chase and Lysyanskaya [46]. The main novelty of our work is in the notion of key-versatility, namely that F-keyed signatures can add signing capability without changing keys, and the ensuing applications to joint encryption and signature, RKA security and KDM security. In particular our work shows that signatures of knowledge have applications beyond those envisaged in [46].

The first NIZK-based signature scheme was that of [15]. It achieved only unforgeability. Simulatability and extractability were achieved in [46] using dense cryptosystems [52, 51] and simulation-sound NIZKs [96, 50]. The DHLW construction we use can be viewed as a simplification and strengthening made possible by the significant advances in NIZK technology since then.

F-keyed signatures, and, more generally, signatures of knowledge [46] can be seen as a signing analogue of Witness encryption [63, 16], and we might have named them Witness Signatures. GGSW [63] show how witness encryption allows encryption with a flexible choice of keys, just as we show that F-keyed signatures allow signing with a flexible choice of keys.

Signcryption [100] (sometimes called authenticated public-key encryption [3]), JES (see Chapter 3) and our secure storage goal all have in common that both encryption and signature are involved. However, in signcryption, there are two parties and thus two sets of keys, Alice encrypting under Bob's public key and signing under her own secret key. In JES and secure storage, there is one set of keys, namely Alice's. Thus for signcryption the question of using the same keys for the two purposes, which is at the core of our goals and methods, does not arise. Self-signcryption [59] is however similar to secure storage, minus the key-dependent message aspect. Authenticated symmetric encryption [20, 93] also involves both encryption and authentication, but under a shared key, while JES and secure storage involve public keys. KDM-secure authenticated symmetric encryption was studied in [17, 10].

KDM-secure signatures were studied in [90], who show limitations on the security achievable. Our secure storage scheme bypasses these limitations by signing ciphertexts rather than plaintexts and by avoiding KDM-secure signatures altogether: we use F-keyed signatures and are making no standalone claims or assumptions regarding their KDM security. Combining KDM encryption and KDM signatures would not give us KDM-secure storage because the keys for the two primitives would be different and we want joint KDM security.

Secure storage is an amalgam of symmetric and asymmetric cryptography, encryption being of the former kind and authentication of the latter. With secure storage, we are directly modelling a goal of practical interest rather than trying to create a general-purpose tool like many of the other works just mentioned. The difference between JES and secure storage is that in the former, arbitrary messages may be signed, while in the latter only ciphertexts may be signed. The difference is crucial for KDM security, which for JES would inherit the limitations of KDM-secure signatures just mentioned, but is not so limited for secure storage.

6.2 Key-Versatile Signatures

We define F-keyed signature schemes, for F a family of functions rather than the single function F used for simplicity in Section 6.1. The requirement is that the secret key sk is an input for an instance pp_{F} of the family and the public key $pk = \mathsf{F.Eval}(pp_{\mathsf{F}}, sk)$ is the corresponding image under this instance, the instance pp_{F} itself specified in public parameters. We intend to use these schemes to add authenticity in a setting where keys (sk, pk) may already be in use for another purpose (such as encryption). We need to ensure that signing will neither lessen the security of the existing usage of the keys nor have its own security be lessened by it. To ensure this strong form of composability, we define simulatability and key-extractability requirements for our F-keyed schemes. The fact that the keys will already be in use

```
main $\mathrm{SIM}^{A}_{\mathsf{DS},\mathsf{F}}(\lambda)$
$b \leftarrow_{\$} \{0,1\}$
$(pp_{\mathsf{F}}, pp_{\mathrm{aux}1}) \leftarrow_{\$} \mathsf{DS.Pg}(1^{\lambda})$
$pp_1 \leftarrow (pp_{\mathsf{F}}, pp_{\mathrm{aux}1})$
$(pp_{\mathrm{aux}0}, std, xtd) \leftarrow_{\$} \mathsf{DS.SimPg}(1^{\lambda})$
$pp_0 \leftarrow (pp_{\mathsf{F}}, pp_{\mathrm{aux}0})$
$b' \leftarrow_{\$} A^{\mathrm{SIGN}}(pp_b)$
Return $(b = b')$

$\mathrm{SIGN}(sk, m)$
If $(sk \notin \mathsf{F.Dom}(pp_{\mathsf{F}}))$
    then Return $\perp$
$pk \leftarrow \mathsf{F.Eval}(pp_{\mathsf{F}}, sk)$
If $(b = 1)$
    then $\sigma \leftarrow_{\$} \mathsf{DS.Sign}(pp_1, sk, m)$
Else $\sigma \leftarrow_{\$} \mathsf{DS.SimSign}(pp_0, std, pk, m)$
Return $\sigma$
```

```
main $\mathrm{EXT}^{A}_{\mathsf{DS},\mathsf{F}}(\lambda)$
$Q \leftarrow \emptyset$
$pp_{\mathsf{F}} \leftarrow_{\$} \mathsf{F.Pg}(1^{\lambda})$
$(pp_{\mathrm{aux}}, std, xtd) \leftarrow_{\$} \mathsf{DS.SimPg}(1^{\lambda})$
$pp \leftarrow (pp_{\mathsf{F}}, pp_{\mathrm{aux}})$
$(pk, m, \sigma) \leftarrow_{\$} A^{\mathrm{SIGN}}(pp)$
If $(pk \notin \mathsf{F.Rng}(pp_{\mathsf{F}}))$
    then Return false
If $(!\mathsf{DS.Verify}(pp, pk, m, \sigma))$
    then Return false
If $((pk, m, \sigma) \in Q)$
    then Return false
$sk \leftarrow_{\$} \mathsf{DS.Ext}(pp, xtd, pk, m, \sigma)$
Return $(\mathsf{F.Eval}(pp_{\mathsf{F}}, sk) \neq pk)$

$\mathrm{SIGN}(sk, m)$
If $(sk \notin \mathsf{F.Dom}(pp_{\mathsf{F}}))$
    then Return $\perp$
$pk \leftarrow \mathsf{F.Eval}(pp_{\mathsf{F}}, sk)$
$\sigma \leftarrow_{\$} \mathsf{DS.SimSign}(pp, std, pk, m)$
$Q \leftarrow Q \cup \{(pk, m, \sigma)\}$
Return $\sigma$
```

Figure 6.1: Left: Game SIM defining simulatability of F-keyed signature scheme DS. Right: Game EXT defining key-extractability.

for another purpose also means that we do not have the luxury of picking the family F, but must work with an arbitrary family emerging from another setting. The only assumption we will make on F is thus that it is one-way. (This is necessary, else security is clearly impossible.) With the definitions in place, we go on to indicate how to build F-keyed signature schemes for arbitrary, one-way F.

We clarify that being F-keyed under an F assumed to be one-way does not mean that security (simulatability and key-extractability) of the signature scheme is based *solely* on the assumption that F is one-way. The additional assumption in our construction is a SE-secure NIZK. (But this itself can be built under standard assumptions.) It is possible to build a signature scheme that is unforgeable assuming only that a given F is one-way [94], but this scheme will not be F-keyed relative to the same F underlying its security, and it will not be simulatable or key-extractable.

F-KEYED SIGNATURE SCHEMES. Let F be a function family. We say that a signature scheme DS is F-*keyed* if the following are true:

- Parameter compatibility: Parameters pp for DS are a pair $pp = (pp_{\mathsf{F}}, pp_{\mathrm{aux}})$ consisting of parameters pp_{F} for F and auxiliary parameters pp_{aux}, these independently generated. Formally, there is a PT *auxiliary parameter generation* algorithm $\mathsf{DS.APg}$ such that $\mathsf{DS.Pg}(1^\lambda)$ picks $pp_{\mathsf{F}} \leftarrow_{\$} \mathsf{F.Pg}(1^\lambda)\,;\, pp_{\mathrm{aux}} \leftarrow_{\$} \mathsf{DS.APg}(1^\lambda)$ and returns $(pp_{\mathsf{F}}, pp_{\mathrm{aux}})$.
- Key compatibility: The signing key sk is a random point in the domain of $\mathsf{F.Eval}$ and the verifying key pk is its image under $\mathsf{F.Eval}$. Formally, $\mathsf{DS.Kg}((pp_{\mathsf{F}}, pp_{\mathrm{aux}}))$ picks $sk \leftarrow_{\$} \mathsf{F.Dom}(pp_{\mathsf{F}})$, lets $pk \leftarrow \mathsf{F.Eval}(pp_{\mathsf{F}}, sk)$ and returns (sk, pk). ($\mathsf{DS.Kg}$ ignores the auxiliary parameters pp_{aux}, meaning the keys do not depend on them.)

6.2.1 Security of F-Keyed Signature Schemes

We require two (strong) security properties of an F-keyed signature scheme DS:

- Simulatable: Under simulated auxiliary parameters and an associated simulation trapdoor std, a simulator, given $pk = \mathsf{F.Eval}(pp_{\mathsf{F}}, sk)$ and m, can produce a signature σ indistinguishable from the real one produced under sk, when not just m, *but even the secret key sk*, is adaptively chosen by the adversary. Formally, DS is *simulatable* if it specifies additional PT algorithms $\mathsf{DS.SimPg}$ (the auxiliary parameter simulator) and $\mathsf{DS.SimSign}$ (the signature simulator) such that $\mathbf{Adv}^{\mathrm{sim}}_{\mathsf{DS},A}(\cdot)$ is negligible for every PT adversary A, where $\mathbf{Adv}^{\mathrm{sim}}_{\mathsf{DS},A}(\lambda) = 2\Pr[\mathrm{SIM}^{A}_{\mathsf{DS}}(\lambda)] - 1$ and game SIM is specified on the left-hand side of Figure 6.1.
- Key-extractable: Under the same simulated auxiliary parameters and an associated extraction trapdoor xtd, an extractor can extract from any valid forgery relative to pk an underlying secret key sk, even when *pk is chosen by the adversary* and the adversary can adaptively obtain simulated signatures *under secret keys of its choice.* Formally, DS is *key-extractable* if it specifies another PT algorithm $\mathsf{DS.Ext}$ (the extractor) such that $\mathbf{Adv}^{\mathrm{ext}}_{\mathsf{DS},A}(\cdot)$ is negligible for every PT adversary A, where $\mathbf{Adv}^{\mathrm{ext}}_{\mathsf{DS},A}(\lambda) = \Pr[\mathrm{EXT}^{A}_{\mathsf{DS}}(\lambda)]$ and game EXT is specified on the right-hand side of Figure 6.1.

The EXT game includes a possibly non-PT test of membership in the range of the family, but we will ensure that adversaries (who must remain PT) do not perform this test. Our definition of simulatability follows [46, 1, 45]. Those definitions were for general signatures, not F-keyed ones, and one difference is that our simulator can set only the auxiliary parameters, not the full parameters, meaning it does not set pp_{F}.

Sim+Ext implies unforgeability. The simulatability and key-extractability notions we have defined may seem quite unrelated to the standard unforgeability requirement for signature schemes [67]. As a warm-up towards applying these new conditions, we show that in fact they imply not just the standard unforgeability but strong unforgeability, under the minimal assumption that F is one-way.

Theorem 6.2.1 *Let DS be an F-keyed signature scheme that is simulatable and key-extractable. If F is one-way then DS is strongly unforgeable.*

Here we sketch the intuition. First we switch to a game using simulated parameters, building an adversary A_1 against the simulatability property of DS to show this change is indistinguishable to an adversary. In the next game we extract a secret key from the adversary's forgery, and set a flag bad if this secret key does not correspond to the challenge public key. We show through an adversary A_2 against the extractability property of DS that this happens with negligible probability, and move to a game that outputs false when bad is set. Now we are in a position to build an inverter I for F. On input (pp_{F}, pk), adversary I generates simulated auxiliary parameters pp_{aux} together with simulation and extraction trapdoors. It now runs A with parameters $(pp_{\mathsf{F}}, pp_{\text{aux}})$, answering signing queries via the signature simulator. (Note the latter only needs the simulation trapdoor and the public key, not the secret key.) When A produces its forgery m, σ, the inverter I runs the extractor to obtain sk, which is a pre-image of pk under $\mathsf{F.Eval}(pp_{\mathsf{F}}, \cdot)$ with the same probability that A wins the final game.

The reader may note that the above theorem would hold under weaker simulatability and extractability conditions where the adversaries do not choose secret and public keys. This is true, but the stronger conditions are crucial to other upcoming applications in this chapter. We proceed to prove Theorem 6.2.1.

Proof: Let A be a PT adversary playing game SUF-CMA. We build PT adversaries A_1, A_2, I such that

$$\mathbf{Adv}^{\text{suf-cma}}_{\mathsf{DS},A}(\lambda) \leq \mathbf{Adv}^{\text{sim}}_{\mathsf{DS},A_1}(\lambda) + \mathbf{Adv}^{\text{ext}}_{\mathsf{DS},A_2}(\lambda) + \mathbf{Adv}^{\text{ow}}_{\mathsf{F},I}(\lambda)$$

for all $\lambda \in \mathbb{N}$, from which the theorem follows.

```
MAIN SUF-CMA$_{\mathsf{DS}}^{A}(\lambda)$
$Q \leftarrow \emptyset$
$(pp_{\mathsf{F}}, pp_{\mathrm{aux}}) \leftarrow\!\!\$\ \mathsf{DS.Pg}(1^{\lambda})$ ; $pp \leftarrow (pp_{\mathsf{F}}, pp_{\mathrm{aux}})$
$(sk, pk) \leftarrow\!\!\$\ \mathsf{DS.Kg}(pp)$
$(m, \sigma) \leftarrow\!\!\$\ A^{\textsc{Sign}}(pp, pk)$
Return $(\mathsf{DS.Verify}(pp, pk, m, \sigma) \wedge ((m, \sigma) \notin Q))$

MAIN $\mathrm{G}_0^{A}(\lambda)$ / $\boxed{\mathrm{G}_1^{A}(\lambda)}$
$Q \leftarrow \emptyset$
$pp_{\mathsf{F}} \leftarrow\!\!\$\ \mathsf{F.Pg}(1^{\lambda})$ ; $(pp_{\mathrm{aux}}, std, xtd) \leftarrow\!\!\$\ \mathsf{DS.SimPg}(1^{\lambda})$ ; $pp \leftarrow (pp_{\mathsf{F}}, pp_{\mathrm{aux}})$
$(sk, pk) \leftarrow\!\!\$\ \mathsf{DS.Kg}(pp)$
$(m, \sigma) \leftarrow\!\!\$\ A^{\textsc{Sign}}(pp, pk)$
$sk' \leftarrow\!\!\$\ \mathsf{DS.Ext}(pp, xtd, pk, m, \sigma)$
If $(\mathsf{DS.Verify}(pp, pk, m, \sigma) \wedge (m, \sigma) \notin Q)$ then
    $d \leftarrow \mathsf{true}$
    If $(\mathsf{F.Eval}(pp_{\mathsf{F}}, sk') \neq pk)$ then $\mathsf{bad} \leftarrow \mathsf{true}$ ; $\boxed{d \leftarrow \mathsf{false}}$
Return $d$

SIGN$(m)$   // SUF-CMA$_{\mathsf{DS}}^{A}(\lambda)$ / $\boxed{\mathrm{G}_0^{A}(\lambda)\ /\ \mathrm{G}_1^{A}(\lambda)}$
$\sigma \leftarrow\!\!\$\ \mathsf{DS.Sign}(pp, sk, m)$
$\boxed{\sigma \leftarrow\!\!\$\ \mathsf{DS.SimSign}(pp, std, pk, m)}$
$Q \leftarrow Q \cup \{(m, \sigma)\}$
Return $\sigma$
```

Figure 6.2: Games used in the proof of Theorem 6.2.1.

The proof uses the games in Figure 6.2. We will build A_1, A_2, I so that for all $\lambda \in \mathbb{N}$ we have

$$\Pr[\text{SUF-CMA}_{\mathsf{DS}}^{A}(\lambda)] - \Pr[\mathrm{G}_0^{A}(\lambda)] \leq \mathbf{Adv}_{\mathsf{DS},A_1}^{\mathrm{sim}}(\lambda) \tag{6.1}$$

$$\Pr[\mathrm{G}_0^{A}(\lambda) \text{ sets } \mathsf{bad}] \leq \mathbf{Adv}_{\mathsf{DS},A_2}^{\mathrm{ext}}(\lambda) \tag{6.2}$$

$$\Pr[\mathrm{G}_1^{A}(\lambda)] \leq \mathbf{Adv}_{\mathsf{F},I}^{\mathrm{ow}}(\lambda)\ . \tag{6.3}$$

Games G_0 and G_1 are identical until bad, so by the Fundamental Lemma of Game-Playing [23] and the above, for all $\lambda \in \mathbb{N}$ we have:

$$\begin{aligned}
\mathbf{Adv}^{\text{suf-cma}}_{\mathsf{DS},A}(\lambda) &= \Pr[\text{SUF-CMA}^A_{\mathsf{DS}}(\lambda)] \\
&= (\Pr[\text{SUF-CMA}^A_{\mathsf{DS}}(\lambda)] - \Pr[\mathrm{G}^A_0(\lambda)]) \\
&\qquad + (\Pr[\mathrm{G}^A_0(\lambda)] - \Pr[\mathrm{G}^A_1(\lambda)]) + \Pr[\mathrm{G}^A_1(\lambda)] \\
&\le (\Pr[\text{SUF-CMA}^A_{\mathsf{DS}}(\lambda)] - \Pr[\mathrm{G}^A_0(\lambda)]) \\
&\qquad + \Pr[\mathrm{G}^A_0(\lambda) \text{ sets } \mathsf{bad}] + \Pr[\mathrm{G}^A_1(\lambda)] \\
&\le \mathbf{Adv}^{\text{sim}}_{\mathsf{DS},A_1}(\lambda) + \mathbf{Adv}^{\text{ext}}_{\mathsf{DS},A_2}(\lambda) + \mathbf{Adv}^{\text{ow}}_{\mathsf{F},I}(\lambda)
\end{aligned}$$

as desired. We proceed to the constructions of A_1, A_2, I. Adversary A_1 against the simulatability of DS behaves as follows:

```
$A_1^{\textsc{Sign}}(pp)$
$Q \leftarrow \emptyset$
$(sk, pk) \leftarrow\$ \mathsf{DS.Kg}(pp)$
$(m, \sigma) \leftarrow\$ A^{\textsc{SignSim}}(pp, pk)$
If $(\mathsf{DS.Verify}(pp, pk, m, \sigma) \wedge ((m, \sigma) \notin Q))$
    then $b' \leftarrow 1$
Else $b' \leftarrow 0$
Return $b'$

$\textsc{SignSim}(m)$
$\sigma \leftarrow\$ \textsc{Sign}(sk, m)$
$Q \leftarrow Q \cup \{(m, \sigma)\}$
Return $\sigma$
```

The $\textsc{Sign}$ oracle that A_1 invokes is its own. When the challenge bit b in game SIM is 0, adversary A_1's $\textsc{Sign}$ oracle returns signatures computed through $\mathsf{DS.SimSign}$ and so A_1 simulates for A game G_0, while if $b = 1$, adversary A_1's $\textsc{Sign}$ oracle returns signatures computed through $\mathsf{DS.Sign}$ and so A_1 simulates game SUF-CMA. We thus have

$$\begin{aligned}
\Pr[\text{SUF-CMA}^A_{\mathsf{DS}}(\lambda)] - \Pr[\mathrm{G}^A_0(\lambda)] &= \Pr[b' = 1 \mid b = 1] - \Pr[b' = 1 \mid b = 0] \\
&\le \mathbf{Adv}^{\text{sim}}_{\mathsf{DS},A_1}(\lambda) \; ,
\end{aligned}$$

which establishes Equation (6.1). Adversary A_2 against the extractability of DS behaves as follows:

```
$A_2^{\textsc{Sign}}(pp)$
$(sk, pk) \leftarrow\$ \mathsf{DS.Kg}(pp)$
$(m, \sigma) \leftarrow\$ A^{\textsc{SignSim}}(pp, pk)$
Return $(pk, m, \sigma)$

$\textsc{SignSim}(m)$
$\sigma \leftarrow\$ \textsc{Sign}(sk, m)$
Return $\sigma$
```

We skip the simple analysis establishing Equation (6.2). Adversary I against the one-wayness of F behaves as follows:

```
I(pp_F, pk)
    $(pp_{\text{aux}}, std, xtd) \leftarrow\!\$\ \mathsf{DS.SimPg}(1^\lambda)$
    $pp \leftarrow (pp_{\mathsf{F}}, pp_{\text{aux}})$
    $(m, \sigma) \leftarrow\!\$\ A^{\text{SIGN}}(pp, pk)$
    $sk' \leftarrow\!\$\ \mathsf{DS.Ext}(pp, xtd, pk, m, \sigma)$
    Return $sk'$

SIGN(m)
    $\sigma \leftarrow\!\$\ \mathsf{DS.SimSign}(pp, std, pk, m)$
    Return $\sigma$
```

The value pk input to I is generated through choosing a random point in the domain of F and applying the evaluation function to it, just as key compatibility dictates $\mathsf{DS.Kg}$ generates pk in game G_1. We exploit in this game the parameter compatibility of DS, which allows us to generate pp_{aux} independently of pp_{F}. We skip the simple analysis establishing Equation (6.3). ▮

6.2.2 Constructing F-Keyed Signature Schemes

A *key-versatile signing schema* is a transform **KvS** that given an arbitrary family of functions F returns an F-keyed signature scheme $\mathsf{DS} = \mathbf{KvS}[\mathsf{F}]$. We want the constructed signature scheme to be simulatable and key-extractable. We now show that this is possible with the aid of appropriate NIZK systems which are themselves known to be constructable under standard assumptions.

Recall from Chapter 2 the definition of a SE (Simulation Extractable) NIZK system. SE was called tSE in [56] and is a variant of NIZK-security notions from [70, 50, 96]. The definition is based on [50, 70, 71, 56]. Roughly, it says that an adversary, given polynomially many simulated proofs of statements of his choosing, cannot come up with a new valid proof from which a witness cannot be extracted.

Before we use such proofs to construct a F-keyed signature scheme, we must know that they exist. The first construction of SE NIZKs (using a stronger notion of simulation extractability) was given in [70], but for a fairly restricted language related to sets of pairing product equations in bilinear groups. In [56] (and further formalised in [73]), the authors provide a generic construction of SE NIZKs from a (regular) NIZK, an IND-CCA encryption scheme, and a one-time signature, which establishes that SE NIZKs exist for all **NP**.

```
$\underline{\mathsf{DS.APg}(1^\lambda):}$
$crs \leftarrow\!\!\$\ \mathsf{NIZK.Pg}(1^\lambda)$
Return $crs$

$\underline{\mathsf{DS.Pg}(1^\lambda):}$
$crs \leftarrow\!\!\$\ \mathsf{DS.APg}(1^\lambda)$
$pp_\mathsf{F} \leftarrow\!\!\$\ \mathsf{F.Pg}(1^\lambda)$
Return $(pp_\mathsf{F}, crs)$

$\underline{\mathsf{DS.Kg}((pp_\mathsf{F}, crs)):}$
$sk \leftarrow\!\!\$\ \mathsf{F.Dom}(pp_\mathsf{F})$
$pk \leftarrow \mathsf{F.Eval}(pp_\mathsf{F}, sk)$
Return $(sk, pk)$

$\underline{\mathsf{DS.Sign}((pp_\mathsf{F}, crs), sk, m):}$
$pk \leftarrow \mathsf{F.Eval}(pp_\mathsf{F}, sk)$
Return $\mathsf{NIZK.P}(crs, (pp_\mathsf{F}, pk, m), sk)$

$\underline{\mathsf{DS.Verify}((pp_\mathsf{F}, crs), pk, m, \sigma):}$
Return $\mathsf{NIZK.V}(crs, (pp_\mathsf{F}, pk, m), \sigma)$

$\underline{\mathsf{DS.SimPg}(1^\lambda):}$
$(crs, std, xtd) \leftarrow\!\!\$\ \mathsf{NIZK.SimPg}(1^\lambda)$
Return $(crs, std, xtd)$

$\underline{\mathsf{DS.SimSign}((pp_\mathsf{F}, crs), std, pk, m):}$
Return $\mathsf{NIZK.SimP}(crs, std, (pp_\mathsf{F}, pk, m))$
$\underline{\mathsf{DS.Ext}((pp_\mathsf{F}, crs), xtd, pk, m, \sigma):}$
Return $\mathsf{NIZK.Ext}(crs, xtd, (pp_\mathsf{F}, pk, m), \sigma)$
```

Figure 6.3: F-keyed signature scheme $\mathsf{DS} = \mathbf{KvS}[\mathsf{F}]$.

The scheme is simple. We define the relation $\mathsf{R}((pp_\mathsf{F}, pk, m), sk)$ to return true iff $\mathsf{F.Eval}(pp_\mathsf{F}, sk) = pk$. A signature of m under sk is then a SE-secure NIZK proof for this relation in which the witness is sk and the instance (input) is (pp_F, pk, m). The interesting aspect of this construction is that it at first sounds blatantly insecure, since the relation R ignores the message m. Does this not mean that a signature is independent of the message, in which case an adversary could violate unforgeability by requesting a signature σ of a message m under pk and then outputting (m', σ) as a forgery for some $m' \neq m$? What prevents this is the strength of the SE notion of NIZKs. The message m is present in the instance (pp_F, pk, m), even if it is ignored by the relation; the proof in turn depends on the instance, making the signature depend on m. Intuitively the SE-secure NIZK guarantees a form of non-malleability, so signatures (proofs) for one message (instance) cannot be transferred to another.

Formally, Let F be a function family. We associate to it the $\mathbf{NP}$-relation R defined by $\mathsf{R}((pp_\mathsf{F}, pk, m), sk) = (\mathsf{F.Eval}(pp_\mathsf{F}, sk) = pk)$ for all $\lambda \in \mathbb{N}$ and all $pp_\mathsf{F}, pk, m, sk \in \{0,1\}^*$. Let NIZK be a NI system for R that is zero knowledge and simulation extractable. The signature scheme $\mathsf{DS} = \mathbf{KvS}[\mathsf{F}]$ is specified in Figure 6.3.

A similar construction of signatures was given in [56] starting from a leakage-resilient hard relation rather than (as in our case) a relation arising from a one-way function. Our construction could be considered a special case of theirs, with the added difference that they use labelled NIZKs with the message as the label while we avoid labels and put the message in the input. The claims established about the construction are however different, with [56] establishing leakage resilience and unforgeability of the signature and our work showing simulatability and key-extractability. The technique of [56] was also used by [45] to construct malleable signatures. Going back further, the first NIZK-based signature scheme was that of [15]. This used PRFs and commitment, but only regular (as opposed to SE) NIZKs, these being all that was available at the time. One might see the simpler and more elegant modern NIZK-based signatures as being made possible by the arrival of the stronger NIZK systems of works like [50, 70, 71, 56].

Security of the construction. Simulatability of the signature scheme follows directly from the zero knowledge property of the NIZK. The key extractability of the signature scheme likewise follows from the SE security of the NIZK.

Theorem 6.2.2 *Assume there exist SE NIZK systems for all of* **NP***. Then there is a key-versatile signing schema* **KvS** *such that if* F *is any family of functions then the signature scheme* $\mathsf{DS} = \mathbf{KvS}[\mathsf{F}]$ *is simulatable and key-extractable.*

Proof: Let A be a PT adversary playing game SIM. We construct a PT adversary A_1 such that $\mathbf{Adv}^{\mathrm{sim}}_{\mathsf{DS},\mathsf{F},A}(\lambda) \le \mathbf{Adv}^{\mathrm{zk}}_{\mathsf{NIZK},\mathsf{R},A_1}(\lambda)$ for all $\lambda \in \mathbb{N}$. Adversary A_1 against the zero-knowledge property of NIZK behaves as follows:

```
$A_1^{\textsc{Prove}}(crs)$                     | $\textsc{Sign}(sk, m)$
$pp_{\mathsf{F}} \leftarrow\$ \mathsf{F.Pg}(1^\lambda)$   | If $(sk \notin \mathsf{F.Dom}(pp_{\mathsf{F}}))$ then Return $\perp$
$pp \leftarrow (pp_{\mathsf{F}}, crs)$           | $pk \leftarrow \mathsf{F.Eval}(pp_{\mathsf{F}}, sk)$
$b' \leftarrow\$ A^{\textsc{Sign}}(pp)$          | $\pi \leftarrow\$ \textsc{Prove}((pp_{\mathsf{F}}, pk, m), sk)$
Return $b'$                                   | Return $\pi$
```

Let A be a PT adversary playing game EXT. We construct a PT adversary A_2 such that $\mathbf{Adv}^{\mathrm{ext}}_{\mathsf{DS},\mathsf{F},A}(\lambda) \le \mathbf{Adv}^{\mathrm{se}}_{\mathsf{NIZK},\mathsf{R},A_2}(\lambda)$ for all $\lambda \in \mathbb{N}$. Adversary A_2 against the simulation extractability of NIZK behaves as follows:

```
$A_2^{\textsc{Prove}}(crs)$                       | $\textsc{Sign}(sk, m)$
$pp_{\mathsf{F}} \leftarrow\$ \mathsf{F.Pg}(1^\lambda)$     | If $(sk \notin \mathsf{F.Dom}(pp_{\mathsf{F}}))$ then Return $\perp$
$pp \leftarrow (pp_{\mathsf{F}}, crs)$             | $pk \leftarrow \mathsf{F.Eval}(pp_{\mathsf{F}}, sk)$
$(pk, m, \sigma) \leftarrow\$ A^{\textsc{Sign}}(pp)$ | $\pi \leftarrow\$ \textsc{Prove}((pp_{\mathsf{F}}, pk, m), sk)$
Return $((pp_{\mathsf{F}}, pk, m), \sigma)$       | Return $\pi$
```

If $(pk, m, \sigma) \notin Q$ in game EXT then $((pp_{\mathsf{F}}, pk, m), \sigma) \notin Q$ in game SE, the sets being those defined in the games. Furthermore, by the definition of $\mathsf{DS.Ext}$ and R, if $sk \leftarrow \mathsf{DS.Ext}(crs, xtd, pk, m, \sigma)$ is such that $\mathsf{F.Eval}(pp_{\mathsf{F}}, sk) \neq pk$, then $\mathsf{R}((pp_{\mathsf{F}}, pk, m), sk) = \mathsf{false}$. ▌

6.3 Joining Signature to Encryption with No Public-Key Overhead

Let PKE be an arbitrary IND-CCA-secure public-key encryption scheme. As an example, it could be the Cramer-Shoup [49], the Kurosawa-Desmedt [82], or the DDN scheme [57], but it could be any other IND-CCA-secure scheme as well. Alice has already established a keypair $(sk_{\mathsf{PKE}}, pk_{\mathsf{PKE}})$ for this scheme, allowing anyone to send her ciphertexts computed under pk_{PKE} that she can decrypt under sk_{PKE}. She wants now to add signature capability. This is easily done. She can create a keypair $(sk_{\mathsf{DS}}, pk_{\mathsf{DS}})$ for her favourite signature scheme and sign an arbitrary message m under sk_{DS}, verification being possible given pk_{DS}. The difficulty is that her public key is now $pk = (pk_{\mathsf{PKE}}, pk_{\mathsf{DS}})$. It is not just larger but will require a new certificate. The question we ask is whether we can add signing capability in a way that is more parsimonious with regard to public key size. Technically, we seek a joint encryption and signature scheme where Alice has a single keypair (sk, pk), with sk used to decrypt and sign, and pk used to encrypt and verify, each usage secure in the face of the other, and we want pk smaller than that of the trivial solution $pk = (pk_{\mathsf{PKE}}, pk_{\mathsf{DS}})$. Perhaps surprisingly, we show how to construct a JES scheme with pk-overhead zero, meaning pk is unchanged, remaining pk_{PKE}. We not only manage to use sk_{PKE} to sign and pk_{PKE} to verify, but do so in such a way that the security of the encryption is not affected by the presence of the signature, and vice versa. Previous standard model JES schemes had been able to reduce the pk-overhead only for *specific* starting encryption schemes (see Chapter 3) while our result says the overhead can be zero regardless of the starting encryption scheme. The result is obtained by defining F as the function mapping sk_{PKE} to pk_{PKE} and using a simulatable and key-extractable F-keyed signature scheme with the keys remaining $(sk_{\mathsf{PKE}}, pk_{\mathsf{PKE}})$.

The base PKE scheme. We are given a public-key encryption scheme PKE meeting the usual notion of IND-CCA security. Let us say that PKE is *canonical* if the operation $(sk, pk) \leftarrow\$\ \mathsf{PKE.Kg}(pp_{\mathsf{PKE}})$ picks sk at random from a finite, non-empty set we denote $\mathsf{PKE.SKSp}(pp_{\mathsf{PKE}})$, and then applies to (pp_{PKE}, sk) a PT deterministic *public-key derivation function* we denote PK to get pk. Canonicity may seem like an extra assumption, but isn't. First, many (most) schemes are already canonical. This is true for the Cramer-Shoup scheme [49], the Kurosawa-Desmedt scheme [82] and for schemes obtained via the CHK transform [32] applied to the identity-based encryption schemes of Boneh-Boyen [29] or Waters [98]. Second, if by chance a scheme is not canonical, we can modify it be so. Crucially (for our purposes), the modification *does not change the public key.* (But it might change the secret key.) Briefly, the modification, which is standard, is to use the random coins of the key generation algorithm as the secret key. In some more detail, given PKE, the new key-generation algorithm, on input pp_{PKE}, picks random coins ω, lets $(sk, pk) \leftarrow \mathsf{PKE.Kg}(pp_{\mathsf{PKE}}; \omega)$, and returns (ω, pk), so that the new secret key is ω and the public key is still pk. Encryption is unchanged. The modified decryption algorithm, given $(pp_{\mathsf{PKE}}, \omega, c)$, lets $(sk, pk) \leftarrow \mathsf{PKE.Kg}(pp_{\mathsf{PKE}}; \omega)$ and outputs $m \leftarrow \mathsf{PKE.Dec}(pp_{\mathsf{PKE}}, sk, c)$. It is easy to see that the modified scheme is canonical and also inherits both the correctness and the IND-CCA security of the original scheme.

Construction. Given a canonical PKE scheme as above, we construct a JES scheme $\mathsf{JES}[\mathsf{PKE}]$. The first step is to construct from PKE a function family F as follows: let $\mathsf{F.Pg} = \mathsf{PKE.Pg}$, so the parameters of F are the same those of PKE; let $\mathsf{F.Dom} = \mathsf{PKE.SKSp}$, so the domain of F is the space of secret keys of PKE; and let $\mathsf{F.Eval} = \mathsf{PKE.PK}$, so the function defined by pp_{F} maps a secret key to a corresponding public key. Now let DS be an F-keyed signature scheme that is simulatable and key-extractable. (We can obtain DS via Theorem 6.2.2.) Now we define our JES scheme $\mathsf{JES}[\mathsf{PKE}]$. Let $\mathsf{JES}[\mathsf{PKE}].\mathsf{Pg} = \mathsf{DS.Pg}$, so parameters for $\mathsf{JES}[\mathsf{PKE}]$ have the form $pp_{\mathsf{JES[PKE]}} = (pp_{\mathsf{F}}, pp_{\mathrm{aux}})$, where pp_{F} are parameters for F, which by definition of F are also parameters for PKE. Let $\mathsf{JES}[\mathsf{PKE}].\mathsf{Kg} = \mathsf{DS.Kg}$. Recall that key compatibility requires $\mathsf{DS.Kg}$ first samples from $\mathsf{F.Dom}$ then applies $\mathsf{F.Eval}$, so that keys are those of PKE which are also those of DS. Let $\mathsf{JES}[\mathsf{PKE}].\mathsf{Sign} = \mathsf{DS.Sign}$ and $\mathsf{JES}[\mathsf{PKE}].\mathsf{Verify} = \mathsf{DS.Verify}$, so the signing and verifying algorithms of the joint scheme $\mathsf{JES}[\mathsf{PKE}]$ are inherited from the signature scheme DS. Let $\mathsf{JES}[\mathsf{PKE}].\mathsf{Enc}((pp_{\mathsf{F}}, pp_{\mathrm{aux}}), pk, m)$ return $\mathsf{PKE.Enc}(pp_{\mathsf{F}}, pk, m)$ and let $\mathsf{JES}[\mathsf{PKE}].\mathsf{Dec}((pp_{\mathsf{F}}, pp_{\mathrm{aux}}), sk, c)$ return $\mathsf{PKE.Dec}$ (pp_{F}, sk, c), so that the encryption and decryption algorithms of the joint scheme $\mathsf{JES}[\mathsf{PKE}]$ are inherited from the PKE scheme PKE. Note that the public key of the joint scheme $\mathsf{JES}[\mathsf{PKE}]$ is exactly that of PKE, so there is zero public-key overhead. (If PKE had been born canonical, there is also zero secret-key overhead. Had it undergone the transformation described above to make it canonical, the secret-key overhead might be non-zero but the public-key overhead would still be zero because the transformation did not change the public key.) The following says that $\mathsf{JES}[\mathsf{PKE}]$ is both IND-CCMA and EUF-CCMA secure.

Theorem 6.3.1 *Let* PKE *be an IND-CCA-secure canonical public-key encryption scheme. Let* F *be defined from it as above. Let* DS *be a simulatable and key-extractable* F*-keyed signature scheme, and let* $\mathsf{JES[PKE]}$ *be the corresponding joint encryption and signature scheme constructed above. Then (1)* $\mathsf{JES[PKE]}$ *is IND-CCMA secure, and (2)* $\mathsf{JES[PKE]}$ *is EUF-CCMA secure.*

First we give some intuition. For (1), given an adversary A against the IND-CCMA security of $\mathsf{JES[PKE]}$, we build an adversary D against the IND-CCA security of PKE. D will simply run A on simulated auxiliary parameters, using the simulator to answer A's Sign queries and using its own Dec oracle to answer A's Dec queries. An adversary against simulatability is built alongside, but key extraction is not needed. For (2), given an adversary A against the EUF-CCMA security of $\mathsf{JES[PKE]}$, we again build an adversary D against the IND-CCA security of PKE. It will run A with simulated auxiliary parameters, replying to A's oracle queries as before. From a forgery it extracts the secret key, using this to defeat IND-CCA security. Adversaries A_1 and A_2 against simulatability and key-extractability of DS are built alongside to show that D succeeds.

Proof: Part (1): IND-CCMA security

Let A be a PT adversary playing game IND-CCMA. We build PT adversaries A_1, D such that

$$\mathbf{Adv}^{\text{ind-ccma}}_{\mathsf{JES[PKE]},A}(\lambda) \leq 2\mathbf{Adv}^{\text{sim}}_{\mathsf{DS},\mathsf{F},A_1}(\lambda) + \mathbf{Adv}^{\text{ind-cca}}_{\mathsf{PKE},D}(\lambda)$$

for all $\lambda \in \mathbb{N}$, from which part (1) of the theorem follows.

The proof uses the games in Figure 6.4. Game G_0 switches to using simulated parameters and signatures. We will build A_1, D so that for all $\lambda \in \mathbb{N}$ we have

$$\Pr[\text{IND-CCMA}^A_{\mathsf{JES[PKE]}}(\lambda)] - \Pr[\mathrm{G}_0^A(\lambda)] \leq \mathbf{Adv}^{\text{sim}}_{\mathsf{DS},\mathsf{F},A_1}(\lambda) \tag{6.4}$$

$$2\Pr[\mathrm{G}_0^A(\lambda)] - 1 \leq \mathbf{Adv}^{\text{ind-cca}}_{\mathsf{PKE},D}(\lambda)\ . \tag{6.5}$$

Using this we have

$$\begin{aligned}
\mathbf{Adv}^{\text{ind-ccma}}_{\mathsf{JES[PKE]},A}(\lambda) &= 2\Pr[\text{IND-CCMA}^A_{\mathsf{JES[PKE]}}(\lambda)] - 1 \\
&= 2\Big(\Pr[\text{IND-CCMA}^A_{\mathsf{JES[PKE]}}(\lambda)] - \Pr[\mathrm{G}_0^A(\lambda)] + \Pr[\mathrm{G}_0^A(\lambda)]\Big) - 1 \\
&= 2\Big(\Pr[\text{IND-CCMA}^A_{\mathsf{JES[PKE]}}(\lambda)] - \Pr[\mathrm{G}_0^A(\lambda)]\Big) + 2\Pr[\mathrm{G}_0^A(\lambda)] - 1 \\
&\leq 2\mathbf{Adv}^{\text{sim}}_{\mathsf{DS},\mathsf{F},A_1}(\lambda) + \mathbf{Adv}^{\text{ind-cca}}_{\mathsf{PKE},D}(\lambda)
\end{aligned}$$

```
main  $\text{IND-CCMA}^{A}_{\mathsf{JES[PKE]}}(\lambda)$ / $\boxed{\mathrm{G}^{A}_{0}(\lambda)}$
$b \leftarrow_{\$} \{0,1\}$ ; $c^* \leftarrow \perp$
$(pp_{\mathsf{F}}, pp_{\mathrm{aux}}) \leftarrow_{\$} \mathsf{DS.Pg}(1^{\lambda})$
$\boxed{pp_{\mathsf{F}} \leftarrow_{\$} \mathsf{F.Pg}(1^{\lambda})\ ;\ (pp_{\mathrm{aux}}, std, xtd) \leftarrow_{\$} \mathsf{DS.SimPg}(1^{\lambda})}$
$pp_{\mathsf{JES[PKE]}} \leftarrow (pp_{\mathsf{F}}, pp_{\mathrm{aux}})$ ; $(sk, pk) \leftarrow_{\$} \mathsf{DS.Kg}(pp_{\mathsf{JES[PKE]}})$
$b' \leftarrow_{\$} A^{\mathrm{Dec,Sign,LR}}(pp_{\mathsf{JES[PKE]}}, pk)$
Return $(b = b')$

proc Dec$(c)$ // $\text{IND-CCMA}^{A}_{\mathsf{JES[PKE]}}(\lambda)$ / $\mathrm{G}^{A}_{0}(\lambda)$
If $(c = c^*)$ then Return $\perp$
Return $m \leftarrow \mathsf{JES[PKE].Dec}(pp_{\mathsf{JES[PKE]}}, sk, c)$

proc Sign$(m)$ // $\text{IND-CCMA}^{A}_{\mathsf{JES[PKE]}}(\lambda)$ / $\boxed{\mathrm{G}^{A}_{0}(\lambda)}$
Return $\mathsf{DS.Sign}(pp_{\mathsf{JES[PKE]}}, sk, m)$
$\boxed{\text{Return } \mathsf{DS.SimSign}(pp_{\mathsf{JES[PKE]}}, std, pk, m)}$

proc LR$(m_0, m_1)$ // $\text{IND-CCMA}^{A}_{\mathsf{JES[PKE]}}(\lambda)$ / $\mathrm{G}^{A}_{0}(\lambda)$
If $(c^* \neq \perp)$ then Return $\perp$
If $(|m_0| \neq |m_1|)$ then Return $\perp$
$c^* \leftarrow_{\$} \mathsf{JES[PKE].Enc}(pp_{\mathsf{JES[PKE]}}, pk, m_b)$
Return $c^*$
```

Figure 6.4: Games used in the proof of part (1) of Theorem 6.3.1.

as desired. We proceed to the constructions of A_1, D. Adversary A_1 against the simulatability of DS behaves as follows:

```
$A_1^{\mathrm{Sign}}(pp)$
$(sk, pk) \leftarrow_{\$} \mathsf{DS.Kg}(pp)$
$c^* \leftarrow \perp$ ; $d \leftarrow_{\$} \{0,1\}$
$d' \leftarrow_{\$} A^{\mathrm{Dec,SignSim,LR}}(pp, pk)$
If $(d' = d)$ then $b' \leftarrow 1$
Else $b' \leftarrow 0$
Return $b'$

SignSim$(m)$
$\sigma \leftarrow_{\$} \mathrm{Sign}(sk, m)$
Return $\sigma$

Dec$(c)$
If $(c = c^*)$ then $m \leftarrow \perp$
Else $m \leftarrow \mathsf{JES[PKE].Dec}(pp, sk, c)$
Return $m$

LR$(m_0, m_1)$
If $(c^* \neq \perp)$ then Return $\perp$
If $(|m_0| \neq |m_1|)$ then Return $\perp$
$c^* \leftarrow \mathsf{JES[PKE].Enc}(pp, pk, m_d)$
Return $c^*$
```

When the challenge bit b in game SIM is 0, adversary A_1 simulates for A game G_0, and if $b = 1$, adversary A_1 simulates game IND-CCMA. We thus have

$$\begin{aligned}\Pr[\text{IND-CCMA}^A_{\mathsf{JES[PKE]}}(\lambda)] - \Pr[\mathrm{G}_0^A(\lambda)] &= \Pr[b' = 1 \,|\, b = 1] - \Pr[b' = 1 \,|\, b = 0] \\ &\leq \mathbf{Adv}^{\text{sim}}_{\mathsf{DS},\mathsf{F},A_1}(\lambda) \;,\end{aligned}$$

establishing Equation (6.4). Adversary D against the IND-CCA security of PKE behaves as follows:

```
$D^{\text{DEC},\text{LR}}(pp_{\mathsf{F}}, pk)$
$(pp_{\text{aux}}, std, xtd) \leftarrow\$ \mathsf{DS.SimPg}(1^\lambda)$
$pp_{\mathsf{JES[PKE]}} \leftarrow (pp_{\mathsf{F}}, pp_{\text{aux}})$
$b' \leftarrow\$ A^{\text{DEC},\text{SIGN},\text{LR}}(pp_{\mathsf{JES[PKE]}}, pk)$
Return $b'$

$\text{SIGN}(m)$
$\sigma \leftarrow\$ \mathsf{DS.SimSign}(pp_{\mathsf{JES[PKE]}}, std, pk, m)$
Return $\sigma$
```

We omit the analysis establishing Equation (6.5).

Part (2): EUF-CCMA security

Let A be a PT adversary playing game EUF-CCMA. We build PT adversaries A_1, A_2, D such that

$$\mathbf{Adv}^{\text{euf-ccma}}_{\mathsf{JES[PKE]},A}(\lambda) \leq \mathbf{Adv}^{\text{sim}}_{\mathsf{DS},\mathsf{F},A_1}(\lambda) + \mathbf{Adv}^{\text{ext}}_{\mathsf{DS},\mathsf{F},A_2}(\lambda) + \mathbf{Adv}^{\text{ind-cca}}_{\mathsf{PKE},D}(\lambda)$$

for all $\lambda \in \mathbb{N}$, from which part (2) of the theorem follows.

The proof uses the games in Figure 6.5. Games G_0 and G_1 switch to using simulated parameters and signatures. We will build A_1, A_2, D so that for all $\lambda \in \mathbb{N}$ we have

$$\Pr[\text{EUF-CCMA}^A_{\mathsf{JES[PKE]}}(\lambda)] - \Pr[\mathrm{G}_0^A(\lambda)] \leq \mathbf{Adv}^{\text{sim}}_{\mathsf{DS},\mathsf{F},A_1}(\lambda) \qquad (6.6)$$

$$\Pr[\mathrm{G}_0^A(\lambda) \text{ sets } \mathsf{bad}] \leq \mathbf{Adv}^{\text{ext}}_{\mathsf{DS},\mathsf{F},A_2}(\lambda) \qquad (6.7)$$

$$\Pr[\mathrm{G}_1^A(\lambda)] \leq \mathbf{Adv}^{\text{ind-cca}}_{\mathsf{PKE},D}(\lambda) \;. \qquad (6.8)$$

```
main EUF-CCMA$^A_{\mathsf{JES[PKE]}}(\lambda)$
$Q \leftarrow \emptyset$ ; $d \leftarrow \mathsf{false}$
$(pp_{\mathsf{F}}, pp_{\mathrm{aux}}) \leftarrow\$ \mathsf{DS.Pg}(1^\lambda)$
$pp_{\mathsf{JES[PKE]}} \leftarrow (pp_{\mathsf{F}}, pp_{\mathrm{aux}})$ ; $(sk, pk) \leftarrow\$ \mathsf{DS.Kg}(pp_{\mathsf{JES[PKE]}})$
$(m, \sigma) \leftarrow\$ A^{\mathrm{SIGN},\mathrm{DEC}}(pp_{\mathsf{JES[PKE]}}, pk)$
$sk' \leftarrow\$ \mathsf{DS.Ext}(pp_{\mathsf{JES[PKE]}}, xtd, pk, m, \sigma)$
Return $(\mathsf{Verify}(pp_{\mathsf{JES[PKE]}}, pk, m, \sigma) \wedge (m \notin Q))$

main $\mathrm{G}_0^A(\lambda)$ / $\boxed{\mathrm{G}_1^A(\lambda)}$
$Q \leftarrow \emptyset$ ; $d \leftarrow \mathsf{false}$
$pp_{\mathsf{F}} \leftarrow\$ \mathsf{F.Pg}(1^\lambda)$ ; $(pp_{\mathrm{aux}}, std, xtd) \leftarrow\$ \mathsf{DS.SimPg}(1^\lambda)$
$pp_{\mathsf{JES[PKE]}} \leftarrow (pp_{\mathsf{F}}, pp_{\mathrm{aux}})$ ; $(sk, pk) \leftarrow\$ \mathsf{DS.Kg}(pp_{\mathsf{JES[PKE]}})$
$(m, \sigma) \leftarrow\$ A^{\mathrm{SIGN},\mathrm{DEC}}(pp_{\mathsf{JES[PKE]}}, pk)$
$sk' \leftarrow\$ \mathsf{DS.Ext}(pp_{\mathsf{JES[PKE]}}, xtd, pk, m, \sigma)$
If $(\mathsf{Verify}(pp_{\mathsf{JES[PKE]}}, pk, m, \sigma) \wedge (m \notin Q))$ then
    $d \leftarrow \mathsf{true}$
    If $(\mathsf{F.Eval}(pp_{\mathsf{F}}, sk') \neq pk)$ then $\mathsf{bad} \leftarrow \mathsf{true}$ ; $\boxed{d \leftarrow \mathsf{false}}$
Return $d$

proc SIGN($m$) // EUF-CCMA$^A_{\mathsf{JES[PKE]}}(\lambda)$ / $\boxed{\mathrm{G}_0^A(\lambda) \text{ / } \mathrm{G}_1^A(\lambda)}$
$\sigma \leftarrow\$ \mathsf{DS.Sign}(pp_{\mathsf{JES[PKE]}}, sk, m)$
$\boxed{\sigma \leftarrow\$ \mathsf{DS.SimSign}(pp_{\mathsf{JES[PKE]}}, std, pk, m)}$
$Q \leftarrow Q \cup \{m\}$
Return $\sigma$

proc DEC($c$) // EUF-CCMA$^A_{\mathsf{JES[PKE]}}(\lambda)$ / $\mathrm{G}_0^A(\lambda)$ / $\mathrm{G}_1^A(\lambda)$
Return $m \leftarrow \mathsf{Dec}(pp_{\mathsf{JES[PKE]}}, sk, c)$
```

Figure 6.5: Games used in the proof of part (2) of Theorem 6.3.1.

Games G_0 and G_1 are identical until bad, so by the Fundamental Lemma of Game-Playing [23] and the above, for all $\lambda \in \mathbb{N}$ we have:

$$\begin{aligned}
\mathbf{Adv}^{\text{euf-ccma}}_{\mathsf{JES[PKE]},A}(\lambda) &= \Pr[\text{EUF-CCMA}^A_{\mathsf{JES[PKE]}}(\lambda)] \\
&= (\Pr[\text{EUF-CCMA}^A_{\mathsf{JES[PKE]}}(\lambda)] - \Pr[\mathrm{G}_0^A(\lambda)]) \\
&\qquad + (\Pr[\mathrm{G}_0^A(\lambda)] - \Pr[\mathrm{G}_1^A(\lambda)]) + \Pr[\mathrm{G}_1^A(\lambda)] \\
&\leq (\Pr[\text{EUF-CCMA}^A_{\mathsf{JES[PKE]}}(\lambda)] - \Pr[\mathrm{G}_0^A(\lambda)]) \\
&\qquad + \Pr[\mathrm{G}_0^A(\lambda) \text{ sets } \mathsf{bad}] + \Pr[\mathrm{G}_1^A(\lambda)] \\
&\leq \mathbf{Adv}^{\text{sim}}_{\mathsf{DS},\mathsf{F},A_1}(\lambda) + \mathbf{Adv}^{\text{ext}}_{\mathsf{DS},\mathsf{F},A_2}(\lambda) + \mathbf{Adv}^{\text{ind-cca}}_{\mathsf{PKE},D}(\lambda)
\end{aligned}$$

as desired. We proceed to the constructions of A_1, A_2, D. Adversary A_1 against the simulatability of DS behaves as follows:

```
$A_1^{\text{SIGN}}(pp)$
$(sk, pk) \leftarrow\$ \mathsf{DS.Kg}(pp)$ ; $Q \leftarrow \emptyset$
$(m, \sigma) \leftarrow\$ A^{\text{DEC,SIGNSIM}}(pp, pk)$
If $(\mathsf{Verify}(pp, pk, m, \sigma) \wedge (m \notin Q))$
    then $b' \leftarrow 1$
Else $b' \leftarrow 0$
Return $b'$

$\text{DEC}(c)$
$m \leftarrow \mathsf{JES[PKE].Dec}(pp, sk, c)$
Return $m$

$\text{SIGNSIM}(m)$
$\sigma \leftarrow\$ \text{SIGN}(sk, m)$
$Q \leftarrow Q \cup \{m\}$
Return $\sigma$
```

When the challenge bit b in game SIM is 0, adversary A_1 simulates for A game G_0, and if $b = 1$, adversary A_1 simulates game EUF-CCMA. We thus have

$$\begin{aligned}\Pr[\text{EUF-CCMA}^A_{\mathsf{JES[PKE]}}(\lambda)] - \Pr[\mathrm{G}_0^A(\lambda)] &= \Pr[b' = 1 \,|\, b = 1] - \Pr[b' = 1 \,|\, b = 0] \\ &\leq \mathbf{Adv}^{\text{sim}}_{\mathsf{DS},\mathsf{F},A_1}(\lambda)\ ,\end{aligned}$$

establishing Equation (6.6). Adversary A_2 against the key-extractability of DS behaves as follows:

```
$A_2^{\text{SIGN}}(pp)$
$(sk, pk) \leftarrow\$ \mathsf{DS.Kg}(pp)$
$(m, \sigma) \leftarrow\$ A^{\text{DEC,SIGNSIM}}(pp, pk)$
Return $(pk, m, \sigma)$

$\text{DEC}(c)$
$m \leftarrow \mathsf{JES[PKE].Dec}(pp, sk, c)$
Return $m$

$\text{SIGNSIM}(m)$
$\sigma \leftarrow\$ \text{SIGN}(sk, m)$
Return $\sigma$
```

We omit the analysis establishing Equation (6.7). Adversary D against the IND-CCA security of PKE behaves as follows:

```
$D^{\text{DEC,LR}}(pp_{\mathsf{F}}, pk)$
$(pp_{\text{aux}}, std, xtd) \leftarrow\$ \mathsf{DS.SimPg}(1^\lambda)$
$pp_{\mathsf{JES[PKE]}} \leftarrow (pp_{\mathsf{F}}, pp_{\text{aux}})$
$(m, \sigma) \leftarrow\$ A^{\text{DEC,SIGN}}(pp_{\mathsf{JES[PKE]}}, pk)$
$sk' \leftarrow\$ \mathsf{DS.Ext}(pp, xtd, pk, m, \sigma)$
If $(\mathsf{F.Eval}(pp_{\mathsf{F}}, sk') \neq pk)$ then Return 0
$m_0 \leftarrow 0^\lambda$ ; $m_1 \leftarrow 1^\lambda$
$c^* \leftarrow\$ \mathrm{LR}(m_0, m_1)$
$m \leftarrow \mathsf{PKE.Dec}(pp_{\mathsf{F}}, sk', c^*)$
If $(m = m_1)$ then $b' \leftarrow 1$ else $b' \leftarrow 0$
Return $b'$

$\text{SIGN}(m)$
$\sigma \leftarrow\$ \mathsf{DS.SimSign}(pp_{\mathsf{JES[PKE]}}, std, pk, m)$
Return $\sigma$
```

When A wins G_1 we have that $\mathsf{F.Eval}(pp_{\mathsf{F}}, sk') = pk$, so sk' is a valid secret key for pk. By correctness of PKE, we then have $\mathsf{PKE.Dec}(pp_{\mathsf{F}}, sk', \mathsf{PKE.Enc}(pp_{\mathsf{F}}, pk, m_b)) = m_b$, where b is the challenge bit in game IND-CCA, so

$$\mathbf{Adv}^{\text{ind-cca}}_{\mathsf{PKE},D}(\lambda) = \Pr[b' = 1 | b = 1] - \Pr[b' = 1 | b = 0] \geq \Pr[\mathrm{G}_1^A(\lambda)] \ .$$

This is because the first term in the difference above is at least $\Pr[\mathrm{G}_1^A(\lambda)]$ and the second term is zero. The second term is zero because $b' = 1$ only when sk' is a valid secret key for pk and the challenge ciphertext decrypts under sk' to m_1, which cannot happen when $b = 0$ as this would violate the correctness of PKE. This establishes Equation (6.8). ▌

6.4 RKA-Secure Signatures from RKA-Secure OWFs

RKA security is notoriously hard to provably achieve. Recognising this, several authors [66, 14] have suggested a bootstrapping approach in which we build higher-level RKA-secure primitives from lower-level RKA-secure primitives. In this vein, a construction of RKA-secure signatures from RKA-secure PRFs was given in [14]. We improve on this via a construction of RKA-secure signatures from RKA-secure one-way functions. The result is simple: If F is a Φ-RKA-secure OWF then any F-keyed simulatable and key-extractable signature scheme is also Φ-RKA secure. The benefit is that (as we will show) many popular OWFs are already RKA secure and we immediately get new RKA-secure signatures.

RKA SECURITY. Let F be a function family. A class of RKD (related-key deriving) functions Φ for F is a set of PT-computable functions $\phi(pp_{\mathsf{F}}, \cdot) : \mathsf{F.Dom}(pp_{\mathsf{F}}) \to \mathsf{F.Dom}(pp_{\mathsf{F}})$. We say that F is Φ-RKA secure if $\mathbf{Adv}^{\text{ow-rka}}_{\mathsf{F},\Phi,A}(\cdot)$ is negligible for every PT adversary A, where $\mathbf{Adv}^{\text{ow-rka}}_{\mathsf{F},\Phi,A}(\lambda) = \Pr[\text{OWF-RKA}^A_{\mathsf{F},\Phi}(\lambda)]$ and game OWF-RKA is on the left-hand side of Figure 6.6. In this game, A, like in the basic one-wayness notion, is given $y = \mathsf{F.Eval}(pp_{\mathsf{F}}, x)$ and attempts to find x' such that $\mathsf{F.Eval}(pp_{\mathsf{F}}, x') = y$. Now, however, it has help. It can request that the hidden challenge input x be modified to $x' = \phi(pp_{\mathsf{F}}, x)$ for any ϕ of its choice, and obtain $y' = \mathsf{F.Eval}(pp_{\mathsf{F}}, x')$. This should not help it in its inversion task. The definition is from Goldenberg and Liskov [66], adapted to our notation, and represents a particularly simple and basic form of RKA security.

Let DS be an F-keyed signature scheme and let Φ be as above. We say that DS is Φ-RKA secure if $\mathbf{Adv}^{\text{suf-rka}}_{\mathsf{DS},\mathsf{F},\Phi,A}(\cdot)$ is negligible for every PT adversary A, where

```
MAIN OWF-RKA$^A_{\mathsf{F},\Phi}(\lambda)$
$pp_\mathsf{F} \leftarrow_\$ \mathsf{F.Pg}(1^\lambda)$
$x \leftarrow_\$ \mathsf{F.Dom}(pp_\mathsf{F})$
$y \leftarrow \mathsf{F.Eval}(pp_\mathsf{F}, x)$
$x' \leftarrow_\$ A^{\text{EVAL}}(pp_\mathsf{F}, y)$
Return $(\mathsf{F.Eval}(pp_\mathsf{F}, x') = y)$

EVAL$(\phi)$
If $(\phi \notin \Phi)$ then Return $\perp$
$x' \leftarrow \phi(pp_\mathsf{F}, x)$
$y' \leftarrow \mathsf{F.Eval}(pp_\mathsf{F}, x')$
Return $y'$
```

```
MAIN SUF-RKA$^A_{\mathsf{DS},\mathsf{F},\Phi}(\lambda)$
$Q \leftarrow \emptyset$
$(pp_\mathsf{F}, pp_{\text{aux}}) \leftarrow_\$ \mathsf{DS.Pg}(1^\lambda)$
$pp \leftarrow (pp_\mathsf{F}, pp_{\text{aux}})$
$(sk, pk) \leftarrow_\$ \mathsf{DS.Kg}(pp)$
$(m, \sigma) \leftarrow_\$ A^{\text{SIGN}}(pp, pk)$
Return $((\mathsf{Verify}(pp, pk, m, \sigma)) \wedge ((pk, m, \sigma) \notin Q))$

SIGN$(\phi, m)$
If $(\phi \notin \Phi)$ then Return $\perp$
$sk' \leftarrow \phi(pp_\mathsf{F}, sk)$
$pk' \leftarrow \mathsf{F.Eval}(pp_\mathsf{F}, sk')$
$\sigma \leftarrow_\$ \mathsf{Sign}(pp, sk', m)$
$Q \leftarrow Q \cup \{(pk', m, \sigma)\}$
Return $\sigma'$
```

Figure 6.6: Left: Game OWF-RKA defining Φ-RKA security of function family F. Right: Game SUF-RKA defining Φ-RKA security of F-keyed signature scheme DS.

$\mathbf{Adv}^{\text{suf-rka}}_{\mathsf{DS},\mathsf{F},\Phi,A}(\lambda) = \Pr[\text{SUF-RKA}^A_{\mathsf{DS},\mathsf{F},\Phi}(\lambda)]$ and game SUF-RKA is on the right-hand side of Figure 6.6. In this game, A, like in the basic (strong) unforgeability notion, is given public key pk and is attempting to forge a signature under it. Now, however, it has help beyond its usual signing oracle. It can request that the hidden secret key sk be modified to $sk' = \phi(pp_\mathsf{F}, sk)$ for any function ϕ of its choice, and obtain a signature under sk' of any message of its choice. This should not help it in its forgery task. Our definition adapts the one of Bellare, Cash and Miller [14] for Φ-RKA security of arbitrary signature schemes to the special case of F-keyed signature schemes.[1]

Construction. Suppose we are given a Φ-RKA-secure OWF F and want to build a Φ-RKA-secure signature scheme. For the question to even make sense, RKD functions specified by Φ must apply to the secret signing key. Thus, the secret key needs to be an input for the OWF and the public key needs to be the image of the secret key under the OWF. The main technical difficulty is, given F, finding a signature scheme with this property. But this is exactly what a key-versatile signing

[1] One change (strengthening the definition) is that we use a strong unforgeability formulation rather than an unforgeability one. On the other hand while the authors of [14] disallow A a victory from forgery m, σ when m was previously signed under $sk' = sk$, we disallow it when m was previously signed under $pk' = pk$ even if $sk' \neq sk$. In our setting this is more natural since the secret key determines the public key. In any case Theorem 6.4.1 extends to the definition of [14] assuming F is additionally injective or collision-resistant, which is true in most examples.

```
MAIN $\text{SUF-RKA}^{A}_{\mathsf{DS,F},\Phi}(\lambda)$
$Q \leftarrow \emptyset$ ; $d \leftarrow \mathsf{false}$
$(pp_{\mathsf{F}}, pp_{\text{aux}}) \leftarrow_{\$} \mathsf{DS.Pg}(1^{\lambda})$
$pp \leftarrow (pp_{\mathsf{F}}, pp_{\text{aux}})$ ; $(sk, pk) \leftarrow_{\$} \mathsf{DS.Kg}(pp)$
$(m, \sigma) \leftarrow_{\$} A^{\text{SIGN}}(pp, pk)$
Return $(\mathsf{Verify}(pp, pk, m, \sigma) \wedge ((pk, m, \sigma) \notin Q))$

MAIN $\text{G}^{A}_{0}(\lambda)$ / $\boxed{\text{G}^{A}_{1}(\lambda)}$
$Q \leftarrow \emptyset$ ; $d \leftarrow \mathsf{false}$
$pp_{\mathsf{F}} \leftarrow_{\$} \mathsf{F.Pg}(1^{\lambda})$
$(pp_{\text{aux}}, std, xtd) \leftarrow_{\$} \mathsf{DS.SimPg}(1^{\lambda})$
$pp \leftarrow (pp_{\mathsf{F}}, pp_{\text{aux}})$ ; $(sk, pk) \leftarrow_{\$} \mathsf{DS.Kg}(pp)$
$(m, \sigma) \leftarrow_{\$} A^{\text{SIGN}}(pp, pk)$
$sk' \leftarrow_{\$} \mathsf{DS.Ext}(pp, xtd, pk, m, \sigma)$
If $(\mathsf{Verify}(pp, pk, m, \sigma) \wedge ((pk, m, \sigma) \notin Q))$ then
    $d \leftarrow \mathsf{true}$
    If $(\mathsf{F.Eval}(pp_{\mathsf{F}}, sk') \neq pk)$ then $\mathsf{bad} \leftarrow \mathsf{true}$ ; $\boxed{d \leftarrow \mathsf{false}}$
Return $d$

SIGN$(\phi, m)$ // $\text{SUF-CMA}^{A}_{\mathsf{DS}}(\lambda)$ / $\boxed{\text{G}^{A}_{0}(\lambda) \text{ / } \text{G}^{A}_{1}(\lambda)}$
If $(\phi \notin \Phi)$ then Return $\perp$
$sk' \leftarrow \phi(pp_{\mathsf{F}}, sk)$ ; $pk' \leftarrow \mathsf{F.Eval}(pp_{\mathsf{F}}, sk')$
$\sigma \leftarrow_{\$} \mathsf{DS.Sign}(pp, sk', m)$
$\boxed{\sigma \leftarrow_{\$} \mathsf{DS.SimSign}(pp, std, pk', m)}$
$Q \leftarrow Q \cup \{(pk', m, \sigma)\}$
Return $\sigma$
```

Figure 6.7: Games used in the proof of Theorem 6.4.1.

schema gives us. The following says that if the signature scheme produced by this schema is simulatable and key-extractable then it inherits the Φ-RKA security of the OWF.

Theorem 6.4.1 *Let* F *be a Φ-RKA secure one-way function, and let* DS *be a simulatable and key-extractable* F*-keyed signature scheme. Then* DS *is Φ-RKA secure.*

The proof extends that of Theorem 6.2.1. The adversary I that we build uses its EVAL oracle to simulate the SIGN oracle of the given adversary A. In applying the simulation and key-extractability, we make crucial use of the fact that the adversaries can obtain signatures under secret keys of their choice.

Proof: Let A be a PT adversary playing game SUF-RKA. We build PT adversaries A_1, A_2, I such that

$$\mathbf{Adv}^{\text{suf-rka}}_{\mathsf{DS},\mathsf{F},\Phi,A}(\lambda) \leq \mathbf{Adv}^{\text{sim}}_{\mathsf{DS},\mathsf{F},A_1}(\lambda) + \mathbf{Adv}^{\text{ext}}_{\mathsf{DS},\mathsf{F},A_2}(\lambda) + \mathbf{Adv}^{\text{ow-rka}}_{\mathsf{F},\Phi,I}(\lambda)$$

for all $\lambda \in \mathbb{N}$, from which the theorem follows. The proof uses the games in Figure 6.7. Games G_0 and G_1 switch to using simulated parameters and signatures. We will build A_1, A_2, I so that for all $\lambda \in \mathbb{N}$ we have

$$\Pr[\text{SUF-RKA}^A_{\mathsf{DS},\mathsf{F},\Phi}(\lambda)] - \Pr[\mathrm{G}^A_0(\lambda)] \leq \mathbf{Adv}^{\text{sim}}_{\mathsf{DS},\mathsf{F},A_1}(\lambda) \tag{6.9}$$

$$\Pr[\mathrm{G}^A_0(\lambda) \text{ sets } \mathsf{bad}] \leq \mathbf{Adv}^{\text{ext}}_{\mathsf{DS},\mathsf{F},A_2}(\lambda) \tag{6.10}$$

$$\Pr[\mathrm{G}^A_1(\lambda)] \leq \mathbf{Adv}^{\text{ow-rka}}_{\mathsf{F},\Phi,I}(\lambda)\ . \tag{6.11}$$

Games G_0 and G_1 are identical until bad, so by the Fundamental Lemma of Game-Playing [23] and the above, for all $\lambda \in \mathbb{N}$ we have:

$$\begin{aligned}
\mathbf{Adv}^{\text{suf-rka}}_{\mathsf{DS},\mathsf{F},\Phi,A}(\lambda) &= \Pr[\text{SUF-RKA}^A_{\mathsf{DS},\mathsf{F},\Phi}(\lambda)] \\
&= (\Pr[\text{SUF-RKA}^A_{\mathsf{DS},\mathsf{F},\Phi}(\lambda)] - \Pr[\mathrm{G}^A_0(\lambda)]) \\
&\qquad + (\Pr[\mathrm{G}^A_0(\lambda)] - \Pr[\mathrm{G}^A_1(\lambda)]) + \Pr[\mathrm{G}^A_1(\lambda)] \\
&\leq (\Pr[\text{SUF-RKA}^A_{\mathsf{DS},\mathsf{F},\Phi}(\lambda)] - \Pr[\mathrm{G}^A_0(\lambda)]) \\
&\qquad + \Pr[\mathrm{G}^A_0(\lambda) \text{ sets } \mathsf{bad}] + \Pr[\mathrm{G}^A_1(\lambda)] \\
&\leq \mathbf{Adv}^{\text{sim}}_{\mathsf{DS},\mathsf{F},A_1}(\lambda) + \mathbf{Adv}^{\text{ext}}_{\mathsf{DS},\mathsf{F},A_2}(\lambda) + \mathbf{Adv}^{\text{ow-rka}}_{\mathsf{F},\Phi,I}(\lambda)
\end{aligned}$$

as desired. We proceed to the constructions of A_1, A_2, I. Adversary A_1 against the simulatability of DS behaves as follows:

```
$A_1^{\text{Sign}}(pp)$
$Q \leftarrow \emptyset\ ;\ (sk, pk) \leftarrow\$\ \mathsf{DS.Kg}(pp)$
$(m, \sigma) \leftarrow\$\ A^{\text{SignSim}}(pp, pk)$
If $(\mathsf{Verify}(pp, pk, m, \sigma) \wedge ((pk, m, \sigma) \notin Q))$
    then $b' \leftarrow 1$
Else $b' \leftarrow 0$
Return $b'$
```

```
$\text{SignSim}(\phi, m)$
If $(\phi \notin \Phi)$ then Return $\perp$
$sk' \leftarrow \phi(pp_{\mathsf{F}}, sk)$
$pk' \leftarrow \mathsf{F.Eval}(pp_{\mathsf{F}}, sk')$
$\sigma \leftarrow\$\ \text{Sign}(sk', m)$
$Q \leftarrow Q \cup \{(pk', m, \sigma)\}$
Return $\sigma$
```

When the challenge bit b in game SIM is 0, adversary A_1 simulates for A game G_0, and if $b = 1$, adversary A_1 simulates game SUF-RKA. We thus have

$$\begin{aligned}
\Pr[\text{SUF-RKA}^A_{\mathsf{DS},\mathsf{F},\Phi}(\lambda)] - \Pr[\mathrm{G}^A_0(\lambda)] &= \Pr[b' = 1 \mid b = 1] - \Pr[b' = 1 \mid b = 0] \\
&\leq \mathbf{Adv}^{\text{sim}}_{\mathsf{DS},\mathsf{F},A_1}(\lambda)\ ,
\end{aligned}$$

establishing Equation (6.9). Adversary A_2 against the key-extractability of DS behaves as follows:

```
$\underline{A_2^{\textsc{Sign}}(pp)}$
$Q \leftarrow \emptyset$
$(sk, pk) \leftarrow\$ \mathsf{DS.Kg}(pp)$
$(m, \sigma) \leftarrow\$ A^{\textsc{SignSim}}(pp, pk)$
Return $(pk, m, \sigma)$

$\underline{\textsc{SignSim}(\phi, m)}$
If $(\phi \notin \Phi)$ then Return $\perp$
$sk' \leftarrow \phi(pp_{\mathsf{F}}, sk)$
$pk' \leftarrow \mathsf{F.Eval}(pp_{\mathsf{F}}, sk')$
$\sigma \leftarrow\$ \textsc{Sign}(sk', m)$
$Q \leftarrow Q \cup \{(pk', m, \sigma)\}$
Return $\sigma$
```

If bad is set to true in game G_0 then we have: (1) $\mathsf{Verify}(pp, pk, m, \sigma)$ (2) $(pk, m, \sigma) \notin Q$, and (3) $\mathsf{F.Eval}(pp_{\mathsf{F}}, sk') \neq pk$. These are exactly the necessary conditions for A_2 to win game EXT, establishing Equation (6.10). Adversary I against the one-wayness of F behaves as follows:

```
$\underline{I^{\textsc{Eval}}(pp_{\mathsf{F}}, pk)}$
$(pp_{\mathrm{aux}}, std, xtd) \leftarrow\$ \mathsf{DS.SimPg}(1^\lambda)$
$(m, \sigma) \leftarrow\$ A^{\textsc{SignSim}}((pp_{\mathsf{F}}, pp_{\mathrm{aux}}), pk)$
$sk' \leftarrow\$ \mathsf{DS.Ext}(pp, xtd, pk, m, \sigma)$
Return $sk'$

$\underline{\textsc{SignSim}(\phi, m)}$
If $(\phi \notin \Phi)$ then Return $\perp$
$pk' \leftarrow\$ \textsc{Eval}(\phi)$
$\sigma \leftarrow\$ \mathsf{DS.SimSign}(pp, std, pk', m)$
Return $\sigma$
```

Adversary I uses the simulation trapdoor *std* to answer A's signing queries, generating the appropriate public key through a call to his EVAL oracle. Whenever A wins G_1, that is A outputs a new valid forgery from which a valid secret key can be extracted, I outputs this secret key and breaks the RKA security of F, establishing Equation (6.11). ▌

FINDING Φ-RKA OWFS. Theorem 6.4.1 motivates finding Φ-RKA-secure function families F. The merit of our approach is that there are many such families. To enable systematically identifying them, we adapt the definition of key-malleable PRFs of [13] to OWFs. We say that a function family F is *Φ-key-malleable* if there is a PT algorithm M, called a Φ-key-simulator, such that $M(pp_{\mathsf{F}}, \phi, \mathsf{F.Eval}(pp_{\mathsf{F}}, x)) = \mathsf{F.Eval}(pp_{\mathsf{F}}, \phi(pp_{\mathsf{F}}, x))$ for all $pp_{\mathsf{F}} \in [\mathsf{F.Pg}(1^\lambda)]$, all $\phi \in \Phi$ and all $x \in \mathsf{F.Dom}(pp_{\mathsf{F}})$.

Proposition 6.4.2 *Let* F *be a Φ-key-malleable, one-way function family. Then* F *is Φ-RKA secure.*

Proof: Let A be a PT adversary attacking the Φ-RKA security of F and let M be a Φ-key-simulator. We construct a PT adversary A_1 against the (regular) one-wayness

F	EXP	RSA	LWE
pp_{F}	$(\langle\mathbb{G}\rangle, g, m)$	(N, e)	(A, n, m, q)
x	$\in \mathbb{Z}_m$	$\in \mathbb{Z}_N^*$	$(s, e) \in \mathbb{Z}_q^m \times \mathbb{Z}_q^n$
$\mathsf{F.Eval}(pp_{\mathsf{F}}, x)$	g^x	$x^e \bmod N$	$(As+e) \bmod q$
$\phi(pp_{\mathsf{F}}, x)$	$(ax+b) \bmod m$,	$x^a \bmod N$,	$(s+s', e+e') \bmod q$,
	$(a, b) \in \mathbb{Z}_m \times \mathbb{Z}_m$	$a \in \mathbb{N}$	$(s', e') \in \mathbb{Z}_q^m \times \mathbb{Z}_q^n$
$M(pp_{\mathsf{F}}, \phi, y)$	$y^a g^b$	$y^a \bmod N$	$(y+As'+e') \bmod q$

Figure 6.8: Φ-RKA secure OWFs: We succinctly define the families and the Φ-key-simulator showing their Φ malleability and hence their Φ-RKA security.

of F such that $\mathbf{Adv}^{\text{ow-rka}}_{\mathsf{F},A}(\lambda) \le \mathbf{Adv}^{\text{ow}}_{\mathsf{F},A_1}(\lambda)$ for all $\lambda \in \mathbb{N}$. On input (pp_{F}, y), adversary A_1 runs $A(pp_{\mathsf{F}}, y)$. When A makes an EVAL query ϕ, adversary A_1 computes $y' \leftarrow M(pp_{\mathsf{F}}, \phi, y)$ and returns y' to A. Φ-key malleability says that $y' = \mathsf{F.Eval}(pp_{\mathsf{F}}, \phi(pp_{\mathsf{F}}, x))$ as A expects. When A eventually halts and outputs a value x', adversary A_1 does the same. ∎

Previous uses of key-malleability in [13] and in Chapter 4 for RKA security required additional conditions on the primitives, such as key-fingerprints in the first case and some form of collision-resistance in the second. For OWFs, it is considerably easier, key-malleability alone sufficing. We now exemplify how to leverage Proposition 6.4.2 to find Φ-RKA OWFs and thence, via Theorem 6.4.1, Φ-RKA signature schemes. Table 6.8 examines three popular one-way functions: discrete exponentiation in a cyclic group, RSA, and the LWE one-way function. It succinctly describes F, Φ, and the Φ-key-simulator M showing Φ-key-malleability. Briefly:

- EXP: The first column of Table 6.8 shows that exponentiation in any group with hard discrete logarithm problem is Φ-RKA secure for the class Φ of affine functions over the exponent space. Here $\mathbb{G}$ is a cyclic group of order m generated by $g \in \mathbb{G}$.
- RSA: The second column of Table 6.8 shows that the RSA function is Φ-RKA secure for the class Φ of functions raising the input to integer powers a, under the assumption that RSA is one-way. Here N is an RSA modulus and $e \in \mathbb{Z}^*_{\varphi(N)}$ is an encryption exponent. Notice that in this rendition of RSA the latter has no trapdoor.
- LWE: The third column of Table 6.8 shows that the LWE function is Φ-RKA secure for the class Φ of functions shown. Here A is an n by m matrix over $\mathbb{Z}_q$ and Φ-RKA-security relies on the standard LWE one-wayness assumption.

The summary is that standard, natural one-way functions are Φ-RKA secure, leading

```
main IND-KDM$^A_{\mathsf{PKE},\Phi}(\lambda)$
$b \leftarrow\!\!\$\ \{0,1\}$ ; $\mathbf{pk} \leftarrow \perp$
$pp \leftarrow\!\!\$\ \mathsf{PKE.Pg}(1^\lambda)$
$b' \leftarrow\!\!\$\ A^{\textsc{MkKey},\textsc{Enc}}(pp)$
Return $(b = b')$

$\textsc{MkKey}(1^n)$
If $(\mathbf{pk} \neq \perp)$ then Return $\perp$
For $i = 1, \ldots, n$ do $(\mathbf{sk}[i], \mathbf{pk}[i]) \leftarrow\!\!\$\ \mathsf{PKE.Kg}(pp)$
Return $\mathbf{pk}$

$\textsc{Enc}(\phi, i)$
If $(\mathbf{pk} = \perp)$ then Return $\perp$
If $(\phi \notin \Phi)$ then Return $\perp$
If $!(1 \leq i \leq n)$ then Return $\perp$
$m \leftarrow \phi(\mathbf{sk})$
If $(b = 1)$ then $c \leftarrow\!\!\$\ \mathsf{PKE.Enc}(pp, \mathbf{pk}[i], m)$
Else $c \leftarrow\!\!\$\ \mathsf{PKE.Enc}(pp, \mathbf{pk}[i], 0^{|m|})$
Return $c$
```

Figure 6.9: Game IND-KDM defining Φ-KDM security of public-key encryption scheme PKE.

to signature schemes that are Φ-RKA secure over standard and natural keyspaces.

6.5 KDM-Secure Storage

Services like Dropbox, Google Drive and Amazon S3 offer outsourced storage. Users see obvious benefits but equally obvious security concerns. We would like to secure this storage, even when messages (files needing to be stored) depend on the keys securing them. If privacy is the only concern, existing KDM-secure encryption schemes (e.g., [27, 36, 9, 7, 43, 44, 26, 11, 87, 6, 40, 41, 17, 62, 75]) will do the job. However, integrity is just as much of a concern, and adding it without losing KDM security is challenging. This is because conventional ways of adding integrity introduce new keys and create new ways for messages to depend on keys. Key-versatile signing, by leaving the keys unchanged, will provide a solution.

We begin below by formalising our goal of encrypted and authenticated outsourced storage secure for key-dependent messages. In our syntax, the user encrypts and authenticates under her secret key, and then verifies and decrypts under the same secret key, with the public key used by the server for verification. Our requirement for KDM security has two components: IND for privacy and SUF for integrity. With the definitions in hand, we take a base KDM-secure encryption scheme and show how, via a key-versatile signature, to obtain storage schemes meeting our goal. Our resulting storage schemes will achieve KDM security with respect to the same class of message-deriving functions Φ as the underlying encryption scheme. Also, we will assume only CPA KDM security of the base scheme, yet achieve CCA KDM privacy for the constructed storage scheme. Interestingly, our solution uses a *public-key* base encryption scheme, even though the privacy component of the goal is symmetric and nobody but the user will encrypt. This allows us to start with KDM privacy under keys permitting signatures through key-versatile signing. This represents a novel application for public-key KDM-secure encryption.

KDM SECURITY. A class of KDM (key-dependent message) functions Φ is a set of PT-computable message-deriving functions $\phi(\cdot)$. These functions take input a vector $\mathbf{sk}$ (of keys) and return a string (the message) of length $\ell(\phi)$, where $\ell(\phi) \in \mathbb{N}$, the output length of ϕ, is computable in polynomial time. We assume Φ always includes all constant functions. Formally, Φ contains the function $\phi_m(\mathbf{sk}) = m$ for all $m \in \mathsf{MSp}$. We say that public-key encryption scheme PKE is Φ-KDM secure if $\mathbf{Adv}^{\text{ind-kdm}}_{\mathsf{PKE},\Phi,A}(\cdot)$ is negligible for every PT adversary A, where $\mathbf{Adv}^{\text{ind-kdm}}_{\mathsf{PKE},\Phi,A}(\lambda) = 2\Pr[\text{IND-KDM}^{A}_{\mathsf{PKE},\Phi}(\lambda)] - 1$ and game IND-KDM is in Figure 6.9. The argument $n \geq 1$ to MkKey determines the number of keys and must be given in unary. The definition follows [27] except in parameterising security by the class Φ of allowed message-deriving functions. The parameterisation is important because many existing KDM-secure encryption schemes are for particular classes Φ, for example for encryption cycles, affine functions or cliques [36, 7, 40, 41, 75]. We aim to transfer whatever KDM security we have in the encryption to the secure storage, meaning we want to preserve Φ regardless of what it is. Of course a particularly interesting case is that of "full" security, but this is captured as the special case where Φ is all functions.

In the setting of direct practical interest, Alice arguably has just one key, corresponding to the vector $\mathbf{sk}$ above having just one component. However, as noted above, much of the literature on KDM security concerns itself with the encryption of cycles and cliques, which represent message-deriving functions on multiple keys, and so our definitions allow the latter.

SECURE STORAGE SCHEMES. A storage scheme ST specifies the following PT algorithms: via $pp \leftarrow\!\!\$\, \mathsf{ST.Pg}(1^\lambda)$ one generates public parameters pp common to all users; via $(sk, pk) \leftarrow\!\!\$\, \mathsf{ST.Kg}(pp)$ a user can generate a secret key sk and corresponding public key pk; via $D \leftarrow\!\!\$\, \mathsf{ST.Store}(pp, sk, m)$ a user can produce some data D based on $m \in \mathsf{ST.MSp}(pp)$ to store on the server; via $m \leftarrow \mathsf{ST.Retrieve}(pp, sk, D)$ a user can deterministically retrieve $m \in \mathsf{ST.MSp}(pp) \cup \{\perp\}$ from their stored data D; and via $d \leftarrow \mathsf{ST.Verify}(pp, pk, D)$ the server can deterministically produce a decision $d \in \{\mathsf{true}, \mathsf{false}\}$ regarding the validity of D. Correctness requires that $\mathsf{ST.Retrieve}(pp, sk, \mathsf{ST.Store}(pp, sk, m)) = m$ and $\mathsf{ST.Verify}(pp, pk, \mathsf{ST.Store}(pp, sk, m)) = \mathsf{true}$ for all $\lambda \in \mathbb{N}$, all $pp \in [\mathsf{ST.Pg}(1^\lambda)]$, all $(sk, pk) \in [\mathsf{ST.Kg}(pp)]$, and all messages $m \in \mathsf{ST.MSp}(pp)$.

We say that ST is Φ-IND secure if $\mathbf{Adv}^{\mathrm{ind}}_{\mathsf{ST},\Phi,A}(\cdot)$ is negligible for all PT adversaries A, where $\mathbf{Adv}^{\mathrm{ind}}_{\mathsf{ST},\Phi,A}(\lambda) = 2\Pr[\mathrm{IND}^{A}_{\mathsf{ST},\Phi}(\lambda)] - 1$ and game IND is on the left-hand side of Figure 6.10. The presence of the RETRIEVE oracle makes this a CCA KDM notion. We say that ST is Φ-SUF secure if $\mathbf{Adv}^{\mathrm{suf}}_{\mathsf{ST},\Phi,A}(\cdot)$ is negligible for all PT adversaries A, where $\mathbf{Adv}^{\mathrm{suf}}_{\mathsf{ST},\Phi,A}(\lambda) = \Pr[\mathrm{SUF}^{A}_{\mathsf{ST},\Phi}(\lambda)]$ and game SUF is on the right-hand side of Figure 6.10. In both cases, we require that the argument $n \geq 1$ given to MKKEY, indicating the number of keys, be given in unary.

CONSTRUCTION. The base scheme we take as given is a Φ-KDM secure, canonical public-key encryption scheme PKE. As in Section 6.3, we begin by constructing from PKE a function family F. We do not repeat this construction here, but refer the reader to Section 6.3. We then let DS be an F-keyed signature scheme that is simulatable and key-extractable. We construct our storage scheme ST through an "Encrypt then Sign" construction as follows:

- $\underline{\mathsf{ST.Pg}(\lambda)}$: Return $(pp_{\mathsf{F}}, pp_{\mathrm{aux}}) \leftarrow\!\!\$\, \mathsf{DS.Pg}(1^\lambda)$. Thus, parameters for ST have the form $pp = (pp_{\mathsf{F}}, pp_{\mathrm{aux}})$, where pp_{F} are parameters for both F and PKE.
- $\underline{\mathsf{ST.Kg}((pp_{\mathsf{F}}, pp_{\mathrm{aux}}))}$: Return $(sk, pk) \leftarrow\!\!\$\, \mathsf{DS.Kg}((pp_{\mathsf{F}}, pp_{\mathrm{aux}}))$. Thus, the keys are those of PKE and DS.
- $\underline{\mathsf{ST.Store}((pp_{\mathsf{F}}, pp_{\mathrm{aux}}), sk, m)}$: $pk \leftarrow \mathsf{F.Eval}(pp_{\mathsf{F}}, sk)$; $c \leftarrow\!\!\$\, \mathsf{PKE.Enc}(pp_{\mathsf{F}}, pk, m)$; $\sigma \leftarrow\!\!\$\, \mathsf{DS.Sign}((pp_{\mathsf{F}}, pp_{\mathrm{aux}}), sk, c)$; Return (c, σ).
- $\underline{\mathsf{ST.Retrieve}((pp_{\mathsf{F}}, pp_{\mathrm{aux}}), sk, (c, \sigma))}$: $pk \leftarrow \mathsf{F.Eval}(pp_{\mathsf{F}}, sk)$;
  If $\mathsf{DS.Verify}((pp_{\mathsf{F}}, pp_{\mathrm{aux}}), pk, c, \sigma) = \mathsf{false}$ then Return $\perp$
  Else Return $\mathsf{PKE.Dec}(pp_{\mathsf{F}}, sk, c)$.
- $\underline{\mathsf{ST.Verify}((pp_{\mathsf{F}}, pp_{\mathrm{aux}}), pk, (c, \sigma))}$: Return $\mathsf{DS.Verify}((pp_{\mathsf{F}}, pp_{\mathrm{aux}}), pk, c, \sigma)$.

The following says that our construction provides both privacy and integrity for

```
main $\mathrm{IND}^{A}_{\mathsf{ST},\Phi}(\lambda)$
$b \leftarrow_{\$} \{0,1\}$ ; $Q \leftarrow \emptyset$ ; $\mathbf{pk} \leftarrow \perp$
$pp \leftarrow_{\$} \mathsf{ST.Pg}(1^\lambda)$
$b' \leftarrow_{\$} A^{\textsc{MkKey},\textsc{Store},\textsc{Retrieve}}(pp)$
Return $(b = b')$

$\textsc{MkKey}(1^n)$
If $(\mathbf{pk} \neq \perp)$ then Return $\perp$
For $i = 1, \ldots, n$ do
    $(\mathbf{sk}[i], \mathbf{pk}[i]) \leftarrow_{\$} \mathsf{ST.Kg}(pp)$
Return $\mathbf{pk}$

$\textsc{Store}(\phi, i)$
If $(\mathbf{pk} = \perp)$ then Return $\perp$
If $(\phi \notin \Phi)$ then Return $\perp$
If $!(1 \leq i \leq n)$ then Return $\perp$
If $(b = 1)$
    then $m \leftarrow \phi(\mathbf{sk})$
    Else $m \leftarrow 0^{\ell(\phi)}$
$D \leftarrow_{\$} \mathsf{ST.Store}(pp, \mathbf{sk}[i], m)$
$Q \leftarrow Q \cup \{(D, i)\}$
Return $D$

$\textsc{Retrieve}(D, i)$
If $(\mathbf{pk} = \perp)$ then Return $\perp$
If $!(1 \leq i \leq n)$ then Return $\perp$
If $((D, i) \in Q)$ then Return $\perp$
$m \leftarrow \mathsf{ST.Retrieve}(pp, \mathbf{sk}[i], D)$
Return $m$
```

```
main $\mathrm{SUF}^{A}_{\mathsf{ST},\Phi}(\lambda)$
$Q \leftarrow \emptyset$ ; $\mathbf{pk} \leftarrow \perp$
$pp \leftarrow_{\$} \mathsf{ST.Pg}(1^\lambda)$
$(D, i) \leftarrow_{\$} A^{\textsc{MkKey},\textsc{Store},\textsc{Retrieve}}(pp)$
If $(D, i) \in Q$ then Return false
If $!(1 \leq i \leq n)$ then Return false
Return $\mathsf{ST.Verify}(pp, \mathbf{pk}[i], D)$

$\textsc{MkKey}(1^n)$
If $(\mathbf{pk} \neq \perp)$ then Return $\perp$
For $i = 1, \ldots, n$ do
    $(\mathbf{sk}[i], \mathbf{pk}[i]) \leftarrow_{\$} \mathsf{ST.Kg}(pp)$
Return $\mathbf{pk}$

$\textsc{Store}(\phi, i)$
If $(\mathbf{pk} = \perp)$ then Return $\perp$
If $(\phi \notin \Phi)$ then Return $\perp$
If $!(1 \leq i \leq n)$ then Return $\perp$
$m \leftarrow \phi(\mathbf{sk})$
$D \leftarrow_{\$} \mathsf{ST.Store}(pp, \mathbf{sk}[i], m)$
$Q \leftarrow Q \cup \{(D, i)\}$
Return $D$

$\textsc{Retrieve}(D, i)$
If $(\mathbf{pk} = \perp)$ then Return $\perp$
If $!(1 \leq i \leq n)$ then Return $\perp$
If $((D, i) \in Q)$ then Return $\perp$
$m \leftarrow \mathsf{ST.Retrieve}(pp, \mathbf{sk}[i], D)$
Return $m$
```

Figure 6.10: Left: Game IND defining Φ-IND security of storage scheme ST. Right: Game SUF defining Φ-SUF security.

key-dependent messages, assuming Φ-KDM security of the base encryption scheme and simulatability and key-extractability of the F-keyed signature scheme:

Theorem 6.5.1 *Let* PKE *be a* Φ*-KDM-secure canonical PKE scheme, and let* F *be defined from it as above. Let* DS *be a simulatable and key-extractable* F*-keyed signature scheme, and let* ST *be the corresponding storage scheme constructed above. Then (1)* ST *is* Φ*-IND secure and (2)* ST *is* Φ*-SUF secure.*

First we provide sketches to highlight some of the unusual difficulties. Taking first the proof of privacy, we would like, given an adversary A breaking the Φ-IND security of ST, to build an adversary D breaking the assumed Φ-KDM security of PKE. The first problem is how D can create the signatures needed to answer STORE queries of A, since these rely on secret keys hidden from D. We solve this by switching to simulation parameters, so that D can simulate signatures without a secret key. In answering RETRIEVE queries, however, we run into another problem: the assumed KDM security of PKE is only under CPA. To solve this, we use the extractor to extract the secret key from signatures and decrypt under it. The full proof involves building simulation and extractability adversaries in addition to D.

Turning next to the proof of unforgeability, we might at first expect that it relies on nothing more than the unforgeability of the signature scheme, so that given an adversary A breaking the Φ-SUF security of ST we could build an adversary breaking the SUF security of DS. However, we run into the basic issue that, since the same keys are used for signing, encryption, and decryption, an adversary against the unforgeability of the signature scheme cannot even construct the messages (ciphertexts) on which A would forge. Instead, we will build from A an adversary D breaking the Φ-KDM security of PKE. This adversary will extract a secret key from a forgery of A and use this to break privacy. To get D to work we must first, as above, switch to simulated signatures, and then use extractability to switch to a simpler RETRIEVE oracle.

Proof: Part (1): Φ-IND security

Let A be a PT adversary playing game IND. Let $q(\cdot)$ be a polynomial such that the number of RETRIEVE queries of A in game $\mathrm{IND}^{A}_{\mathsf{ST},\Phi}(\lambda)$ is $q(\lambda)$ for all $\lambda \in \mathbb{N}$. We provide PT adversaries A_1, A_2, D and a negligible function $\nu(\cdot)$ such that

$$\mathbf{Adv}^{\mathrm{ind}}_{\mathsf{ST},\Phi,A}(\lambda) \leq 2\mathbf{Adv}^{\mathrm{sim}}_{\mathsf{DS},\mathsf{F},A_1}(\lambda) + \mathbf{Adv}^{\mathrm{ind\text{-}kdm}}_{\mathsf{PKE},\Phi,D}(\lambda) + 2\nu(\lambda) + 2q(\lambda) \cdot \mathbf{Adv}^{\mathrm{ext}}_{\mathsf{DS},\mathsf{F},A_2}(\lambda)$$

for all $\lambda \in \mathbb{N}$, from which part (1) of the theorem follows.

The proof uses the games in Figure 6.11 and Figure 6.12. Game G_0 is as IND, but switches to simulated parameters and keeps track of the stored data objects (c, σ) together with the public keys they were computed under in a list Q'. The RETRIEVE oracle in game G_1 extracts a secret key sk' from σ and uses it to decrypt c. If the public key pk' computed from sk' is not equal to $\mathbf{pk}[i]$, it instead uses $\mathbf{sk}[i]$ to decrypt c. Correctness of PKE says that when $\mathsf{F.Eval}(pp_{\mathsf{F}}, sk') = \mathsf{F.Eval}(pp_{\mathsf{F}}, \mathbf{sk}[i]) = \mathbf{pk}[i]$, decryption under sk' and $\mathbf{sk}[i]$ return the same value, so the responses are identical to those of the RETRIEVE oracle in game G_0, and we have

$$\Pr[\mathrm{G}_0^A(\lambda)] = \Pr[\mathrm{G}_1^A(\lambda)] \ . \tag{6.12}$$

Game G_2 is as G_1, but if the RETRIEVE oracle sets the flag bad, that is if A submits $((c, \sigma), i)$ such that the public key computed from the secret key extracted from σ is not equal to $\mathbf{pk}[i]$, it decrypts c under the extracted key sk' instead of under $\mathbf{sk}[i]$. Games G_1 and G_2 are thus identical until bad, so by the Fundamental Lemma of Game-Playing [23] we have

$$\Pr[\mathrm{G}_1^A(\lambda)] - \Pr[\mathrm{G}_2^A(\lambda)] \leq \Pr[\mathrm{G}_1^A(\lambda) \text{ sets } \mathsf{bad}] \ . \tag{6.13}$$

Game G_3 is as G_1, but additionally sets bad if A submits $((c, \sigma), i)$ such that $(\mathbf{pk}[i], c, \sigma)$ is in the list Q', so we have

$$\Pr[\mathrm{G}_1^A(\lambda) \text{ sets } \mathsf{bad}] \leq \Pr[\mathrm{G}_3^A(\lambda) \text{ sets } \mathsf{bad}] \ . \tag{6.14}$$

We build A_1, A_2, D and $\nu(\cdot)$ such that for all $\lambda \in \mathbb{N}$ we have

$$\Pr[\mathrm{IND}_{\mathsf{ST},\Phi}^A(\lambda)] - \Pr[\mathrm{G}_0^A(\lambda)] \leq \mathbf{Adv}_{\mathsf{DS},\mathsf{F},A_1}^{\mathrm{sim}}(\lambda) \tag{6.15}$$

$$2\Pr[\mathrm{G}_2^A(\lambda)] - 1 \leq \mathbf{Adv}_{\mathsf{PKE},\Phi,D}^{\text{ind-kdm}}(\lambda) \tag{6.16}$$

$$\Pr[\mathrm{G}_3^A(\lambda) \text{ sets } \mathsf{bad}] \leq \nu(\lambda) + q(\lambda) \cdot \mathbf{Adv}_{\mathsf{DS},\mathsf{F},A_2}^{\mathrm{ext}}(\lambda) \ . \tag{6.17}$$

Combining Equations (6.12)-(6.17), for all $\lambda \in \mathbb{N}$ we have:

$$\begin{aligned}
\mathbf{Adv}_{\mathsf{ST},\Phi,A}^{\mathrm{ind}}(\lambda) &= 2\Pr[\mathrm{IND}_{\mathsf{ST},\Phi}^A(\lambda)] - 1 \\
&= 2\left(\Pr[\mathrm{IND}_{\mathsf{ST},\Phi}^A(\lambda) - \Pr[\mathrm{G}_0^A(\lambda)]\right) + 2\left(\Pr[\mathrm{G}_0^A(\lambda)] - \Pr[\mathrm{G}_1^A(\lambda)]\right) \\
&\qquad + 2\left(\Pr[\mathrm{G}_1^A(\lambda)] - \Pr[\mathrm{G}_2^A(\lambda)]\right) + 2\Pr[\mathrm{G}_2^A(\lambda)] - 1 \\
&\leq 2\left(\Pr[\mathrm{IND}_{\mathsf{ST},\Phi}^A(\lambda) - \Pr[\mathrm{G}_0^A(\lambda)]\right) \\
&\qquad + 2\Pr[\mathrm{G}_1^A(\lambda) \text{ sets } \mathsf{bad}] + 2\Pr[\mathrm{G}_2^A(\lambda)] - 1 \\
&\leq 2\mathbf{Adv}_{\mathsf{DS},\mathsf{F},A_1}^{\mathrm{sim}}(\lambda) + \mathbf{Adv}_{\mathsf{PKE},\Phi,D}^{\text{ind-kdm}}(\lambda) + 2\Pr[\mathrm{G}_1^A(\lambda) \text{ sets } \mathsf{bad}] \\
&\leq 2\mathbf{Adv}_{\mathsf{DS},\mathsf{F},A_1}^{\mathrm{sim}}(\lambda) + \mathbf{Adv}_{\mathsf{PKE},\Phi,D}^{\text{ind-kdm}}(\lambda) + 2\Pr[\mathrm{G}_3^A(\lambda) \text{ sets } \mathsf{bad}] \\
&\leq 2\mathbf{Adv}_{\mathsf{DS},\mathsf{F},A_1}^{\mathrm{sim}}(\lambda) + \mathbf{Adv}_{\mathsf{PKE},\Phi,D}^{\text{ind-kdm}}(\lambda) \\
&\qquad + 2\nu(\lambda) + 2q(\lambda) \cdot \mathbf{Adv}_{\mathsf{DS},\mathsf{F},A_2}^{\mathrm{ext}}(\lambda)
\end{aligned}$$

```
MAIN $\boxed{\mathrm{IND}^{A}_{\mathsf{ST},\Phi}(\lambda)}$ / $\boxed{\mathrm{G}^{A}_{0}(\lambda)\ /\ \mathrm{G}^{A}_{1}(\lambda)\ /\ \mathrm{G}^{A}_{2}(\lambda)\ /\ \mathrm{G}^{A}_{3}(\lambda)}$
$b \leftarrow_{\$} \{0,1\}$ ; $Q \leftarrow \emptyset$ ; $\mathbf{pk} \leftarrow \perp$ ; $\boxed{Q' \leftarrow \emptyset}$
$\boxed{(pp_{\mathsf{F}}, pp_{\mathrm{aux}}) \leftarrow_{\$} \mathsf{DS.Pg}(1^{\lambda})}$
$\boxed{pp_{\mathsf{F}} \leftarrow_{\$} \mathsf{F.Pg}(1^{\lambda})\ ;\ (pp_{\mathrm{aux}}, std, xtd) \leftarrow_{\$} \mathsf{DS.SimPg}(1^{\lambda})}$
$pp \leftarrow (pp_{\mathsf{F}}, pp_{\mathrm{aux}})$ ; $b' \leftarrow_{\$} A^{\text{MkKey},\text{Store},\text{Retrieve}}(pp)$
Return $(b = b')$

MkKey$(1^n)$   // $\mathrm{IND}^{A}_{\mathsf{ST},\Phi}(\lambda)$ / $\mathrm{G}^{A}_{0}(\lambda)$ / $\mathrm{G}^{A}_{1}(\lambda)$ / $\mathrm{G}^{A}_{2}(\lambda)$ / $\mathrm{G}^{A}_{3}(\lambda)$
If $(\mathbf{pk} \neq \perp)$ then Return $\perp$
For $i = 1, \ldots, n$ do $(\mathbf{sk}[i], \mathbf{pk}[i]) \leftarrow_{\$} \mathsf{DS.Kg}(pp)$
Return $\mathbf{pk}$

proc Store$(\phi, i)$   // $\boxed{\mathrm{IND}^{A}_{\mathsf{ST},\Phi}(\lambda)}$ / $\boxed{\mathrm{G}^{A}_{0}(\lambda)\ /\ \mathrm{G}^{A}_{1}(\lambda)\ /\ \mathrm{G}^{A}_{2}(\lambda)\ /\ \mathrm{G}^{A}_{3}(\lambda)}$
If $(\mathbf{pk} = \perp)$ then Return $\perp$
If $(\phi \notin \Phi)$ then Return $\perp$
If $!(1 \le i \le n)$ then Return $\perp$
If $(b = 1)$ then $m \leftarrow \phi(\mathbf{sk})$ Else $m \leftarrow 0^{\ell(\phi)}$
$c \leftarrow_{\$} \mathsf{PKE.Enc}(pp_{\mathsf{F}}, \mathbf{pk}[i], m)$
$\boxed{\sigma \leftarrow_{\$} \mathsf{DS.Sign}(pp, \mathbf{sk}[i], c)}$ $\boxed{\sigma \leftarrow_{\$} \mathsf{DS.SimSign}(pp, std, \mathbf{pk}[i], c)}$
$Q \leftarrow Q \cup \{((c,\sigma), i)\}$ ; $\boxed{Q' \leftarrow Q' \cup \{(\mathbf{pk}[i], c, \sigma)\}}$
Return $(c, \sigma)$

proc Retrieve$((c,\sigma), i)$   // $\mathrm{IND}^{A}_{\mathsf{ST},\Phi}(\lambda)$ / $\mathrm{G}^{A}_{0}(\lambda)$
If $(\mathbf{pk} = \perp)$ then Return $\perp$
If $!(1 \le i \le n)$ then Return $\perp$
If $(((c,\sigma), i) \in Q)$ then Return $\perp$
If $!(\mathsf{DS.Verify}(pp, \mathbf{pk}[i], c, \sigma))$ then Return $\perp$
$m \leftarrow \mathsf{PKE.Dec}(pp_{\mathsf{F}}, \mathbf{sk}[i], c)$
Return $m$
```

Figure 6.11: Games used in the proof of part (1) of Theorem 6.5.1.

```
$\underline{\text{proc Retrieve}((c,\sigma),i)}$   // $\boxed{G_1^A(\lambda)}$ / $G_2^A(\lambda)$
If $(\mathbf{pk} = \perp)$ then Return $\perp$
If $!(1 \le i \le n)$ then Return $\perp$
If $(((c,\sigma),i) \in Q)$ then Return $\perp$
If $!(\mathsf{DS.Verify}(pp, \mathbf{pk}[i], c, \sigma))$ then Return $\perp$
$sk' \leftarrow_{\$} \mathsf{DS.Ext}(pp, xtd, \mathbf{pk}[i], c, \sigma)$ ; $pk' \leftarrow \mathsf{F.Eval}(pp_{\mathsf{F}}, sk')$
$m \leftarrow \mathsf{PKE.Dec}(pp_{\mathsf{F}}, sk', c)$
If $(pk' \neq \mathbf{pk}[i])$ then $\mathsf{bad} \leftarrow \mathsf{true}$ ; $\boxed{m \leftarrow \mathsf{PKE.Dec}(pp_{\mathsf{F}}, \mathbf{sk}[i], c)}$
Return $m$

$\underline{\text{proc Retrieve}((c,\sigma),i)}$   // $G_3^A(\lambda)$
If $(\mathbf{pk} = \perp)$ Return $\perp$
If $!(1 \le i \le n)$ then Return $\perp$
If $(((c,\sigma),i) \in Q)$ then Return $\perp$
If $!(\mathsf{DS.Verify}(pp, \mathbf{pk}[i], c, \sigma))$ then Return $\perp$
$m \leftarrow \mathsf{PKE.Dec}(pp_{\mathsf{F}}, \mathbf{sk}[i], c)$
If $(\mathbf{pk}[i], c, \sigma) \in Q'$ then $\mathsf{bad} \leftarrow \mathsf{true}$ ; Return $m$
$sk' \leftarrow_{\$} \mathsf{DS.Ext}(pp, xtd, \mathbf{pk}[i], c, \sigma)$ ; $pk' \leftarrow \mathsf{F.Eval}(pp_{\mathsf{F}}, sk')$
If $(pk' \neq \mathbf{pk}[i])$ then $\mathsf{bad} \leftarrow \mathsf{true}$
Return $m$
```

Figure 6.12: Games used in the proof of part (1) of Theorem 6.5.1, continued.

as desired. We proceed to the constructions of A_1, A_2, D and ν.

Adversary A_1 against the simulatability of DS behaves as follows:

```
$\underline{A_1^{\text{SIGN}}(1^\lambda, pp)}$
$Q \leftarrow \emptyset$ ; $d \leftarrow\$ \{0,1\}$ ; $\mathbf{pk} \leftarrow \perp$
$(pp_{\mathsf{F}}, pp_{\text{aux}}) \leftarrow pp$
$d' \leftarrow\$ A^{\text{MkKey,Store,Retrieve}}(pp)$
If $(d' = d)$ then $b' \leftarrow 1$ Else $b' \leftarrow 0$
Return $b'$

$\underline{\text{MkKey}(1^n)}$
If $(\mathbf{pk} \neq \perp)$ Return $\perp$
For $i = 1, \ldots, n$ do
    $(\mathbf{sk}[i], \mathbf{pk}[i]) \leftarrow\$ \mathsf{DS.Kg}(pp)$
Return $\mathbf{pk}$
```

```
$\underline{\text{Store}(\phi, i)}$
If $(\mathbf{pk} = \perp)$ Return $\perp$
If $(\phi \notin \Phi)$ Return $\perp$
If $!(1 \leq i \leq n)$ then Return $\perp$
If $(d = 1)$ then $m \leftarrow \phi(\mathbf{sk})$
Else $m \leftarrow 0^{\ell(\phi)}$
$c \leftarrow\$ \mathsf{PKE.Enc}(pp_{\mathsf{F}}, \mathbf{pk}[i], m)$
$\sigma \leftarrow\$ \text{Sign}(\mathbf{sk}[i], c)$
$Q \leftarrow Q \cup \{((c, \sigma), i)\}$
Return $(c, \sigma)$

$\underline{\text{Retrieve}((c, \sigma), i)}$
If $(\mathbf{pk} = \perp)$ Return $\perp$
If $!(1 \leq i \leq n)$ then Return $\perp$
If $(((c, \sigma), i) \in Q)$ then Return $\perp$
If $(\mathsf{DS.Verify}(pp, \mathbf{pk}[i], c, \sigma))$
  then Return $\perp$
$m \leftarrow \mathsf{PKE.Dec}(pp_{\mathsf{F}}, \mathbf{sk}[i], c)$
Return $m$
```

When the challenge bit b in game SIM is 1, Sign returns real signatures and so adversary A_1 simulates IND, while when $b = 0$ Sign returns simulated signatures and so A_1 simulates G_0. We thus have

$$\Pr[\text{IND}_{\mathsf{ST},\Phi}^{A}(\lambda)] - \Pr[\text{G}_0^{A}(\lambda)] = \Pr[b' = 1 \mid b = 1] - \Pr[b' = 1 \mid b = 0] \leq \mathbf{Adv}_{\mathsf{DS},\mathsf{F},A_1}^{\text{sim}}(\lambda),$$

establishing Equation (6.15).

Adversary D against the Φ-KDM security of PKE behaves as follows:

```
$D^{\textsc{MkKey},\textsc{Enc}}(pp_{\mathsf{F}})$
$Q \leftarrow \emptyset$
$(pp_{\text{aux}}, std, xtd) \leftarrow\$ \mathsf{DS.SimPg}(1^\lambda)$
$pp \leftarrow (pp_{\mathsf{F}}, pp_{\text{aux}})$
$b' \leftarrow\$ A^{\textsc{MkKey},\textsc{Store},\textsc{Retrieve}}(pp)$
Return $b'$

$\textsc{Store}(\phi, i)$
If $(\mathbf{pk} = \bot)$ Return $\bot$
If $(\phi \notin \Phi)$ Return $\bot$
If $!(1 \leq i \leq n)$ then Return $\bot$
$c \leftarrow\$ \textsc{Enc}(\phi, i)$
$\sigma \leftarrow\$ \mathsf{DS.SimSign}(pp, std, \mathbf{pk}[i], c)$
$Q \leftarrow Q \cup \{((c,\sigma), i)\}$
Return $(c, \sigma)$

$\textsc{Retrieve}((c,\sigma), i)$
If $(\mathbf{pk} = \bot)$ Return $\bot$
If $!(1 \leq i \leq n)$ then Return $\bot$
If $(((c,\sigma), i) \in Q)$ then Return $\bot$
If $!(\mathsf{DS.Verify}(pp, \mathbf{pk}[i], c, \sigma))$
   then Return $\bot$
$sk' \leftarrow\$ \mathsf{DS.Ext}(pp, xtd, \mathbf{pk}[i], c, \sigma)$
$m \leftarrow \mathsf{PKE.Dec}(pp_{\mathsf{F}}, sk', c)$
Return $m$
```

D correctly simulates game G_2 for A, successfully determining b' whenever A does, establishing Equation (6.16).

Now we would like to show that G_1 (equivalently, G_2) sets bad with negligible probability. Intuitively, this should follow from extractability, since bad is set when extraction fails. A difficulty is that extraction is not required to succeed when $(pk[i], C, \sigma) \in Q'$. So first we show that the latter event is unlikely, and then, assuming it does not happen, that failure of extraction is unlikely. This is formalised via G_3, which breaks the setting of bad from G_1 into two parts corresponding to the two events of interest. To establish Equation (6.17), let E_1 be the event that bad is set by the line 6 "If" statement, and E_2 the event that bad is set by the line 8 "If" statement. We first show the existence of a negligible $\nu(\cdot)$ such that $\Pr[E_1] \leq \nu(\lambda)$. Then we build A_2 such that $\Pr[E_2 \wedge \overline{E}_1] \leq q(\lambda) \cdot \mathbf{Adv}^{\text{ext}}_{\mathsf{DS},\mathsf{F},A_2}(\lambda)$. This establishes Equation (6.17), as we have

$$\begin{aligned}\Pr[G_3^A(\lambda) \text{ sets } \mathsf{bad}] &= \Pr[E_1 \vee E_2] \\ &= \Pr[E_1] + \Pr[E_2 \wedge \overline{E}_1] \\ &\leq \nu(\lambda) + q(\lambda) \cdot \mathbf{Adv}^{\text{ext}}_{\mathsf{DS},\mathsf{F},A_2}(\lambda).\end{aligned}$$

For the first claim, let E be the event that there is a collision in the public keys chosen in $\textsc{MkKey}$, meaning there are distinct $i, j \in \{1, \ldots, n\}$ such that $\mathbf{pk}[i] = \mathbf{pk}[j]$. We claim that if this event does not happen, then neither will E_1. This is because setting bad requires that $(\mathbf{pk}[i], c, \sigma) \in Q'$ yet $((c, \sigma), i) \notin Q$, but this cannot happen if the public keys are all distinct. So $\Pr[E_1] \leq \Pr[E]$. However, if E does happen with probability that is not negligible, then it is easy to break KDM security of PKE. An adversary just has to itself sample keypairs, hoping to get one where the public key matches one of her challenge public keys. In that case, having the corresponding secret key, it is easy to defeat security. We omit the details because this argument is standard.

Adversary A_2 against the key-extractability of DS behaves as follows:

```
$\underline{A_2^{\text{SIGN}}(pp)}$
$Q \leftarrow \emptyset$ ; $b \leftarrow\$ \{0,1\}$ ; $\mathbf{pk} = \perp$ ; $j \leftarrow 0$
$(pp_{\mathsf{F}}, pp_{\text{aux}}) \leftarrow pp$
$b' \leftarrow\$ A^{\text{MkKey},\text{Store},\text{Retrieve}}(pp)$
$\ell \leftarrow\$ \{1, \ldots, j\}$
Return $(pk_\ell, c_\ell, \sigma_\ell)$

$\underline{\text{MkKey}(1^n)}$
If $(\mathbf{pk} \neq \perp)$ Return $\perp$
For $i = 1, \ldots, n$ do
    $(\mathbf{sk}[i], \mathbf{pk}[i]) \leftarrow\$ \mathsf{DS.Kg}(pp)$
Return $\mathbf{pk}$
```

```
$\underline{\text{Store}(\phi, i)}$
If $(\mathbf{pk} = \perp)$ Return $\perp$
If $(\phi \notin \Phi)$ Return $\perp$
If $!(1 \leq i \leq n)$ then Return $\perp$
If $(b = 1)$ then $m \leftarrow \phi(\mathbf{sk})$
Else $m \leftarrow 0^{\ell(\phi)}$
$c \leftarrow\$ \mathsf{PKE.Enc}(pp_{\mathsf{F}}, \mathbf{pk}[i], m)$
$\sigma \leftarrow\$ \text{Sign}(\mathbf{sk}[i], c)$
$Q \leftarrow Q \cup \{((c, \sigma), i)\}$
Return $(c, \sigma)$

$\underline{\text{Retrieve}((c, \sigma), i)}$
If $(\mathbf{pk} = \perp)$ Return $\perp$
If $!(1 \leq i \leq n)$ then Return $\perp$
If $(((c, \sigma), i) \in Q)$ then Return $\perp$
If $!(\mathsf{DS.Verify}(pp, \mathbf{pk}[i], c, \sigma))$
   then Return $\perp$
$m \leftarrow \mathsf{PKE.Dec}(pp_{\mathsf{F}}, \mathbf{sk}[i], c)$
$j \leftarrow j + 1$ ; $pk_j \leftarrow \mathbf{pk}[i]$
$c_j \leftarrow c$ ; $\sigma_j \leftarrow \sigma$
Return $m$
```

Adversary A_2 always performs correct decryptions when responding to Retrieve queries, following G_3. If bad is set at line 8 but not at line 6, then there is some tuple on which the extractor would succeed. Since a tuple is guessed at random we have $\Pr[E_2 \wedge \overline{E}_1] \leq q(\lambda) \cdot \mathbf{Adv}^{\text{ext}}_{\mathsf{DS},\mathsf{F},A_2}(\lambda)$ as desired. The importance of bad not being set at line 6 is that otherwise extraction is not required to succeed according to game EXT.

```
MAIN $\mathrm{H}_{\Phi}^{A}(\lambda)$
$Q \leftarrow \emptyset$ ; $\mathsf{win} \leftarrow \mathsf{false}$ ; $\mathbf{pk} \leftarrow \perp$ ; $(pp_{\mathsf{F}}, pp_{\mathrm{aux}}) \leftarrow\$ \mathsf{DS.Pg}(1^{\lambda})$ ; $pp \leftarrow (pp_{\mathsf{F}}, pp_{\mathrm{aux}})$
$\perp \leftarrow\$ A^{\textsc{MkKey},\textsc{Store},\textsc{Retrieve}}(pp)$
Return $\mathsf{win}$

$\textsc{MkKey}(1^{n})$
If $(\mathbf{pk} \neq \perp)$ then Return $\perp$
For $i = 1, \ldots, n$ do $(\mathbf{sk}[i], \mathbf{pk}[i]) \leftarrow\$ \mathsf{DS.Kg}(pp)$
Return $\mathbf{pk}$

$\textsc{Store}(\phi, i)$
If $(\mathbf{pk} = \perp)$ then Return $\perp$
If $(\phi \notin \Phi)$ then Return $\perp$
If $!(1 \leq i \leq n)$ then Return $\perp$
$m \leftarrow \phi(\mathbf{sk})$ ; $c \leftarrow\$ \mathsf{PKE.Enc}(pp_{\mathsf{F}}, \mathbf{pk}[i], m)$
$\sigma \leftarrow\$ \mathsf{DS.Sign}(pp, \mathbf{sk}[i], c)$ ; $Q \leftarrow Q \cup \{((c, \sigma), i)\}$
Return $(c, \sigma)$

$\textsc{Retrieve}((c, \sigma), i)$
If $(\mathbf{pk} = \perp)$ then Return $\perp$
If $!(1 \leq i \leq n)$ then Return $\perp$
If $(((c, \sigma), i) \in Q)$ then Return $\perp$
If $\mathsf{DS.Verify}(pp, \mathbf{pk}[i], c, \sigma)$ then $\mathsf{win} \leftarrow \mathsf{true}$ Else Return $\perp$
$m \leftarrow \mathsf{PKE.Dec}(pp_{\mathsf{F}}, \mathbf{sk}[i], c)$
Return $m$
```

Figure 6.13: Game H defining alternate form of SUF for the proof of part (2) of Theorem 6.5.1.

Part (2): Φ-SUF security

Let A' be a PT adversary playing game SUF. Our first step is to consider the simplified form of the SUF game shown in Figure 6.13. Here the adversary does not output a forgery but instead wins via RETRIEVE queries. We can easily transform A' into a PT adversary A such that $\mathbf{Adv}^{\mathrm{suf}}_{\mathsf{ST},\Phi,A'}(\lambda) \leq \Pr[\mathrm{H}_{\Phi}^{A}(\lambda)]$ for all $\lambda \in \mathbb{N}$. Adversary A simply runs A', answering all queries via its own oracles (the two adversaries have the same oracles). When A' halts with output $((c, \sigma), i)$, A makes query $\textsc{Retrieve}((c, \sigma), i)$ and halts with output $\perp$. The flag win is set to true with at least the probability that A' wins its game. We now proceed to upper bound $\Pr[\mathrm{H}_{\Phi}^{A}(\lambda)]$.

Let $q(\cdot)$ be a polynomial such that the number of RETRIEVE queries of A in game $\mathrm{H}_{\Phi}^{A}(\lambda)$ is $q(\lambda)$ for all $\lambda \in \mathbb{N}$. We provide PT adversaries A_1, A_2, D and a negligible

```
MAIN $\boxed{G_0^A(\lambda)}$ / $G_1^A(\lambda)$ / $G_2^A(\lambda)$
$Q \leftarrow \emptyset$ ; $Q' \leftarrow \emptyset$ ; $\mathsf{win} \leftarrow \mathsf{false}$ ; $\mathbf{pk} \leftarrow \perp$
$pp_{\mathsf{F}} \leftarrow_{\$} \mathsf{F.Pg}(1^\lambda)$ ; $(pp_{\mathrm{aux}}, std, xtd) \leftarrow_{\$} \mathsf{DS.SimPg}(1^\lambda)$ ; $pp \leftarrow (pp_{\mathsf{F}}, pp_{\mathrm{aux}})$
$\perp \leftarrow_{\$} A^{\textsc{MkKey},\textsc{Store},\textsc{Retrieve}}(pp)$
Return $\mathsf{win}$

$\textsc{MkKey}(1^n)$ // $G_0^A(\lambda)$ / $G_1^A(\lambda)$ / $G_2^A(\lambda)$
If ($\mathbf{pk} \neq \perp$) then Return $\perp$
For $i = 1, \ldots, n$ do $(\mathbf{sk}[i], \mathbf{pk}[i]) \leftarrow_{\$} \mathsf{DS.Kg}(pp)$
Return $\mathbf{pk}$

proc $\textsc{Store}(\phi)$ // $G_0^A(\lambda)$ / $G_1^A(\lambda)$ / $G_2^A(\lambda)$
If ($\mathbf{pk} = \perp$) then Return $\perp$
If ($\phi \notin \Phi$) then Return $\perp$
If $!(1 \leq i \leq n)$ then Return $\perp$
$m \leftarrow \phi(\mathbf{sk})$ ; $c \leftarrow_{\$} \mathsf{PKE.Enc}(pp_{\mathsf{F}}, \mathbf{pk}[i], m)$
$\sigma \leftarrow_{\$} \mathsf{DS.SimSign}(pp, std, \mathbf{pk}[i], c)$
$Q \leftarrow Q \cup \{((c, \sigma), i)\}$ ; $Q' \leftarrow Q' \cup \{(\mathbf{pk}[i], c, \sigma)\}$
Return $(c, \sigma)$

proc $\textsc{Retrieve}((c, \sigma), i)$ // $G_0^A(\lambda)$
If ($\mathbf{pk} = \perp$) then Return $\perp$
If $!(1 \leq i \leq n)$ then Return $\perp$
If ($((c, \sigma), i) \in Q$) then Return $\perp$
If ($\mathsf{DS.Verify}(pp, \mathbf{pk}[i], c, \sigma)$) then $\mathsf{win} \leftarrow \mathsf{true}$ Else Return $\perp$
$m \leftarrow \mathsf{PKE.Dec}(pp_{\mathsf{F}}, \mathbf{sk}[i], c)$
Return $m$
```

Figure 6.14: Games used in the proof of part (2) of Theorem 6.5.1.

function $\nu(\cdot)$ such that

$$\Pr[H_{\Phi}^A(\lambda)] \leq \mathbf{Adv}_{\mathsf{DS,F},A_1}^{\mathsf{sim}}(\lambda) + \mathbf{Adv}_{\mathsf{PKE},\Phi,D}^{\mathsf{ind\text{-}kdm}}(\lambda) + \nu(\lambda) + q(\lambda) \cdot \mathbf{Adv}_{\mathsf{DS,F},A_2}^{\mathrm{ext}}(\lambda)$$

for all $\lambda \in \mathbb{N}$, from which part (2) of the theorem follows.

The proof uses the games in Figure 6.14 and Figure 6.15. The modifications between games mirror those made in the proof of part (1) above. Game G_0 is as H, but switches to simulated parameters and keeps track of the stored data objects (c, σ) together with the public keys they were computed under in a list Q'. The RETRIEVE oracle in game G_1 extracts a secret key sk' from σ and uses it to decrypt c. If the public key pk' computed from sk' is not equal to $\mathbf{pk}[i]$, it instead uses $\mathbf{sk}[i]$ to decrypt c. Correctness of PKE says that when $\mathsf{F.Eval}(pp_{\mathsf{F}}, sk') = \mathsf{F.Eval}(pp_{\mathsf{F}}, \mathbf{sk}[i]) = \mathbf{pk}[i]$, decryption under sk' and $\mathbf{sk}[i]$ return the same value, so the responses are identical

```
proc RETRIEVE((c, σ), i)   // $\boxed{G_1^A(\lambda)}$ / $G_2^A(\lambda)$
If ($\mathbf{pk} = \perp$) then Return $\perp$
If $!(1 \le i \le n)$ then Return $\perp$
If $(((c,\sigma),i) \in Q)$ then Return $\perp$
If ($\mathsf{DS.Verify}(pp, \mathbf{pk}[i], c, \sigma)$) then $\mathsf{win} \leftarrow \mathsf{true}$ Else Return $\perp$
$sk' \leftarrow\$ \mathsf{DS.Ext}(pp, xtd, \mathbf{pk}[i], c, \sigma)$ ; $pk' \leftarrow \mathsf{F.Eval}(pp_{\mathsf{F}}, sk')$
$m \leftarrow pk.\mathsf{Dec}(pp_{\mathsf{F}}, sk', c)$
If $(pk' \neq \mathbf{pk}[i])$ then $\mathsf{bad} \leftarrow \mathsf{true}$ ; $\boxed{m \leftarrow \mathsf{PKE.Dec}(pp_{\mathsf{F}}, \mathbf{sk}[i], c)}$
Return $m$

proc RETRIEVE((c, σ), i)   // $G_3^A(\lambda)$
If ($\mathbf{pk} = \perp$) then Return $\perp$
If $!(1 \le i \le n)$ then Return $\perp$
If $(((c,\sigma),i) \in Q)$ then Return $\perp$
If ($\mathsf{DS.Verify}(pp, \mathbf{pk}[i], c, \sigma)$) then $\mathsf{win} \leftarrow \mathsf{true}$ Else Return $\perp$
$m \leftarrow \mathsf{PKE.Dec}(pp_{\mathsf{F}}, \mathbf{sk}[i], c)$
If $(\mathbf{pk}[i], c, \sigma) \in Q'$ then $\mathsf{bad} \leftarrow \mathsf{true}$ ; Return $m$
$sk' \leftarrow\$ \mathsf{DS.Ext}(pp, xtd, \mathbf{pk}[i], c, \sigma)$ ; $pk' \leftarrow \mathsf{F.Eval}(pp_{\mathsf{F}}, sk')$
If $(pk' \neq \mathbf{pk}[i])$ then $\mathsf{bad} \leftarrow \mathsf{true}$
Return $m$
```

Figure 6.15: Games used in the proof of part (2) of Theorem 6.5.1, continued.

to those of the RETRIEVE oracle in game G_0, and we have

$$\Pr[G_0^A(\lambda)] = \Pr[G_1^A(\lambda)] . \tag{6.18}$$

Game G_2 is as G_1, but if the RETRIEVE oracle sets the flag bad, that is if A submits $((c,\sigma),i)$ such that the public key computed from the secret key extracted from σ is not equal to $\mathbf{pk}[i]$, it decrypts c under the extracted key sk' instead of under $\mathbf{sk}[i]$. Games G_1 and G_2 are thus identical until bad, so by a variant of the Fundamental Lemma of Game-Playing [23] we have

$$\Pr[G_1^A(\lambda) \wedge \overline{\mathsf{bad}}] = \Pr[G_2^A(\lambda) \wedge \overline{\mathsf{bad}}] , \tag{6.19}$$

where the notation in Equation (6.20) means that we are considering the event that the game returns true and also bad is not set. Game G_3 is as G_1, but additionally sets bad if A submits $((c,\sigma),i)$ such that $(\mathbf{pk}[i], c, \sigma)$ is in the list Q', so we have

$$\Pr[G_1^A(\lambda) \text{ sets } \mathsf{bad}] \le \Pr[G_3^A(\lambda) \text{ sets } \mathsf{bad}] . \tag{6.20}$$

We build A_1, A_2, D and ν such that for all $\lambda \in \mathbb{N}$ we have

$$\Pr[H_\Phi^A(\lambda)] - \Pr[G_0^A(\lambda)] \le \mathbf{Adv}^{\mathrm{sim}}_{\mathsf{DS},\mathsf{F},A_1}(\lambda) \tag{6.21}$$

$$\Pr[G_2^A(\lambda) \wedge \overline{\mathsf{bad}}] \le \mathbf{Adv}^{\mathrm{ind\text{-}kdm}}_{\mathsf{PKE},\Phi,D}(\lambda) \tag{6.22}$$

$$\Pr[G_3^A(\lambda) \text{ sets } \mathsf{bad}] \le \nu(\lambda) + q(\lambda) \cdot \mathbf{Adv}^{\mathrm{ext}}_{\mathsf{DS},\mathsf{F},A_2}(\lambda) . \tag{6.23}$$

The notation in Equation (6.22) means that we are considering the event that the game returns true and also bad is not set. Now games $\mathrm{G}_1, \mathrm{G}_2$ are identical until bad so a variant of the Fundamental Lemma of Game-Playing [23] says that $\Pr[\mathrm{G}_1^A(\lambda) \wedge \overline{\mathsf{bad}}] = \Pr[\mathrm{G}_2^A(\lambda) \wedge \overline{\mathsf{bad}}]$. We also observe that $\Pr[\mathrm{G}_1^A(\lambda) \text{ sets } \mathsf{bad}] \leq \Pr[\mathrm{G}_3^A(\lambda) \text{ sets } \mathsf{bad}]$. (In both games, decryption in Retrieve is always done correctly.) Combining Equations (6.18)-(6.23), for all $\lambda \in \mathbb{N}$ we have:

$$\begin{aligned}
\mathbf{Adv}^{\mathrm{suf}}_{\mathsf{ST},A',\Phi}(\lambda) &\leq \Pr[\mathrm{H}^A(\lambda)] \\
&= \Pr[\mathrm{H}^A(\lambda)] - \Pr[\mathrm{G}_0^A(\lambda)] + (\Pr[\mathrm{G}_0^A(\lambda)] - \Pr[\mathrm{G}_1^A(\lambda)]) + \Pr[\mathrm{G}_1^A(\lambda)] \\
&\leq \Pr[\mathrm{H}^A(\lambda)] - \Pr[\mathrm{G}_0^A(\lambda)] + \Pr[\mathrm{G}_1^A(\lambda) \wedge \overline{\mathsf{bad}}] + \Pr[\mathrm{G}_1^A(\lambda) \text{ sets } \mathsf{bad}] \\
&\leq \Pr[\mathrm{H}^A(\lambda)] - \Pr[\mathrm{G}_0^A(\lambda)] + \Pr[\mathrm{G}_2^A(\lambda) \wedge \overline{\mathsf{bad}}] + \Pr[\mathrm{G}_1^A(\lambda) \text{ sets } \mathsf{bad}] \\
&\leq \Pr[\mathrm{H}^A(\lambda)] - \Pr[\mathrm{G}_0^A(\lambda)] + \Pr[\mathrm{G}_2^A(\lambda) \wedge \overline{\mathsf{bad}}] + \Pr[\mathrm{G}_3^A(\lambda) \text{ sets } \mathsf{bad}] \\
&\leq \mathbf{Adv}^{\mathrm{sim}}_{\mathsf{DS},\mathsf{F},A_1}(\lambda) + \mathbf{Adv}^{\mathrm{ind\text{-}kdm}}_{\mathsf{PKE},\Phi,D}(\lambda) + \nu(\lambda) + q(\lambda) \cdot \mathbf{Adv}^{\mathrm{ext}}_{\mathsf{DS},\mathsf{F},A_2}(\lambda)
\end{aligned}$$

as desired. We proceed to the constructions of A_1, A_2, D and ν.

Adversary A_1 against the simulatability of DS behaves as follows:

```
$\underline{A_1^{\textsc{Sign}}(pp)}$
$Q \leftarrow \emptyset$ ; $\mathbf{pk} \leftarrow \perp$
$\perp \leftarrow\$ A^{\textsc{MkKey},\textsc{Store},\textsc{Retrieve}}(pp)$
If win then $b' \leftarrow 1$ Else $b' \leftarrow 0$
Return $b'$

$\underline{\textsc{Store}(\phi, i)}$
If $(\mathbf{pk} = \perp)$ then Return $\perp$
If $(\phi \notin \Phi)$ then Return $\perp$
If $!(1 \leq i \leq n)$ then Return $\perp$
$m \leftarrow \phi(\mathbf{sk})$
$c \leftarrow\$ \mathsf{PKE.Enc}(pp_{\mathsf{F}}, \mathbf{pk}[i], m)$
$\sigma \leftarrow\$ \textsc{Sign}(\mathbf{sk}[i], c)$
$Q \leftarrow Q \cup \{((c,\sigma), i)\}$
Return $(c, \sigma)$
```

```
$\underline{\textsc{MkKey}(1^n)}$
If $(\mathbf{pk} \neq \perp)$ then Return $\perp$
For $i = 1, \ldots, n$ do
    $(\mathbf{sk}[i], \mathbf{pk}[i]) \leftarrow\$ \mathsf{DS.Kg}(pp)$
Return $\mathbf{pk}$

$\underline{\textsc{Retrieve}((c,\sigma), i)}$
If $(\mathbf{pk} = \perp)$ then Return $\perp$
If $!(1 \leq i \leq n)$ then Return $\perp$
If $(((c,\sigma), i) \in Q)$ then Return $\perp$
If $(\mathsf{DS.Verify}(pp, \mathbf{pk}[i], c, \sigma))$
    then win $\leftarrow$ true ;
Else Return $\perp$
$m \leftarrow \mathsf{PKE.Dec}(pp_{\mathsf{F}}, \mathbf{sk}[i], c)$
Return $m$
```

When the challenge bit b in game SIM is 1, Sign returns real signatures and so adversary A_1 simulates H, while when $b = 0$ Sign returns simulated signatures and so A_1 simulates G_0. This establishes Equation (6.21).

Adversary D against the Φ-KDM security of PKE behaves as follows:

```
$\underline{D^{\text{MkKey,Enc}}(pp_{\mathsf{F}})}$
$Q \leftarrow \emptyset$ ; $sk^* \leftarrow \perp$ ; $j \leftarrow \perp$ ; $\mathbf{pk} \leftarrow \perp$
$(pp_{\text{aux}}, std, xtd) \leftarrow\$ \mathsf{DS.SimPg}(1^\lambda)$
$pp \leftarrow (pp_{\mathsf{F}}, pp_{\text{aux}})$
$\perp \leftarrow\$ A^{\text{MkKey,Store,Retrieve}}(pp)$
If $(sk^*, j) = (\perp, \perp)$ then Return 0
$m_1 \leftarrow 1^\lambda$ ; $c^* \leftarrow\$ \text{Enc}(\phi_{m_1}, j)$
$m \leftarrow \mathsf{PKE.Dec}(pp_{\mathsf{F}}, sk^*, c^*)$
If $(m = m_1)$ then $b' \leftarrow 1$ Else $b' \leftarrow 0$
Return $b'$

$\underline{\text{Store}(\phi, i)}$
If $(\mathbf{pk} = \perp)$ then Return $\perp$
If $(\phi \notin \Phi)$ then Return $\perp$
If $!(1 \leq i \leq n)$ then Return $\perp$
$c \leftarrow\$ \text{Enc}(\phi, i)$
$\sigma \leftarrow\$ \mathsf{DS.SimSign}(pp, std, \mathbf{pk}[i], c)$
$Q \leftarrow Q \cup \{((c, \sigma), i)\}$
Return $(c, \sigma)$

$\underline{\text{Retrieve}((c, \sigma), i)}$
If $(\mathbf{pk} = \perp)$ then Return $\perp$
If $!(1 \leq i \leq n)$ then Return $\perp$
If $(((c, \sigma), i) \in Q)$ then Return $\perp$
If $(\mathsf{DS.Verify}(pp, \mathbf{pk}[i], c, \sigma))$
    then $\mathsf{win} \leftarrow \mathsf{true}$ ;
Else Return $\perp$
$sk' \leftarrow\$ \mathsf{DS.Ext}(pp, xtd, \mathbf{pk}[i], c, \sigma)$
If $(\mathsf{F.Eval}(pp_{\mathsf{F}}, sk') \neq \mathbf{pk}[i])$
    then $\mathsf{bad} \leftarrow \mathsf{true}$
Else $(sk^*, j) \leftarrow (sk', i)$
$m \leftarrow \mathsf{PKE.Dec}(pp_{\mathsf{F}}, sk', c)$
Return $m$
```

Recall that ϕ_{m_1} denotes the constant function that always returns m_1. If win is set in G_2 then we are assured that there is at least one Retrieve query which leads to extraction being performed. If additionally bad is not set then this extraction succeeds, which means decryption under sk^* will be correct. That in turn means that m is the correct decryption of c^* and hence D succeeds if $\mathsf{win} \wedge \overline{\mathsf{bad}}$. This establishes Equation (6.22).

Now we would like to show that G_1 (equivalently, G_2) sets bad with negligible probability. Intuitively, this should follow from extractability, since bad is set when extraction fails. A difficulty is that extraction is not required to succeed when $(pk[i], c, \sigma) \in Q'$. So first we show that the latter event is unlikely, and then, assuming it does not happen, that failure of extraction is unlikely. This is formalised via G_3, which breaks the setting of bad from G_1 into two parts corresponding to the two events of interest. To establish Equation (6.23), let E_1 be the event that bad is set by the line 6 "If" statement, and E_2 the event that bad is set by the line 8 "If" statement. We show the existence of a negligible $\nu(\cdot)$ such that $\Pr[E_1] \leq \nu(\lambda)$ as in the proof of part (1) above, first arguing that $\Pr[E_1]$ is at most the probability of a collision in public keys, and then arguing that this is negligible by the assumed security of PKE. To establish Equation (6.23), we now build an adversary A_2 against the key-extractability of DS so that $\Pr[E_2 \wedge \overline{E}_1] \leq q(\lambda) \cdot \mathbf{Adv}^{\text{ext}}_{\mathsf{DS},\mathsf{F},A_2}(\lambda)$:

```
$\underline{A_2^{\text{SIGN}}(pp)}$
$Q \leftarrow \emptyset$ ; $j \leftarrow 0$ ; $\mathbf{pk} \leftarrow \perp$
$\perp \leftarrow\$ A^{\text{MKKEY,STORE,RETRIEVE}}(pp)$
$\ell \leftarrow\$ \{1, \ldots, j\}$
Return $(pk_\ell, c_\ell, \sigma_\ell)$

$\underline{\text{MKKEY}(1^n)}$
If $(\mathbf{pk} \neq \perp)$ then Return $\perp$
For $i = 1, \ldots, n$ do
    $(\mathbf{sk}[i], \mathbf{pk}[i]) \leftarrow\$ \mathsf{DS.Kg}(pp)$
Return $\mathbf{pk}$

$\underline{\text{STORE}(\phi, i)}$
If $(\mathbf{pk} = \perp)$ then Return $\perp$
f $(\phi \notin \Phi)$ then Return $\perp$
If $!(1 \leq i \leq n)$ then Return $\perp$
$m \leftarrow \phi(\mathbf{sk})$
$c \leftarrow\$ \mathsf{PKE.Enc}(pp_{\mathsf{F}}, \mathbf{pk}[i], m)$
$\sigma \leftarrow\$ \text{SIGN}(\mathbf{sk}[i], c)$ ; $Q \leftarrow Q \cup \{((c, \sigma), i)\}$
Return $(c, \sigma)$

$\underline{\text{RETRIEVE}((c, \sigma), i)}$
If $(\mathbf{pk} = \perp)$ then Return $\perp$
If $!(1 \leq i \leq n)$ then Return $\perp$
If $(((c, \sigma), i) \in Q)$ then Return $\perp$
If $!(\mathsf{DS.Verify}(pp, \mathbf{pk}[i], c, \sigma))$ then Return $\perp$
$m \leftarrow \mathsf{PKE.Dec}(pp_{\mathsf{F}}, \mathbf{sk}[i], c)$
$j \leftarrow j + 1$ ; $pk_j \leftarrow \mathbf{pk}[i]$ ; $c_j \leftarrow c$ ; $\sigma_j \leftarrow \sigma$
Return $m$
```

Adversary A_2 always returns correct decryptions in responding to RETRIEVE queries, following G_3. If bad is set at line 8 but not at line 6, then there is some tuple on which the extractor would succeed. Since a tuple is guessed at random we have $\Pr[E_2 \wedge \overline{E}_1] \leq q(\lambda) \cdot \mathbf{Adv}^{\text{ext}}_{\mathsf{DS,F},A_2}(\lambda)$ as desired. The importance of bad not being set at line 6 is that otherwise extraction is not required to succeed according to game EXT. ▮

INSTANTIATION. We require our base scheme PKE to be canonical. In Section 6.3 we showed how to modify an encryption scheme to be canonical while preserving IND-CCA security, but the transformation does not in general preserve KDM security. Instead, we would use a KDM-secure scheme that is already canonical. One possibility is the scheme of [87]. The schemes of [36, 11, 7] are not canonical.

6.6 Conclusion

This chapter introduced key-versatile signatures, signatures where public keys are computed from private keys through an arbitrary one-way function family F. Key-versatile signatures allow us to sign with keys already in use for another purpose, without changing the keys and without impacting the security of the original purpose.

6.6 Conclusion

We required strong security properties for key-versatile signatures, namely simulatability and key-extractability, conditions that are significantly stronger than (and together imply) the usual unforgeability. These properties allowed us to obtain advances across a collection of challenging domains including joint encryption and signature, security against related-key attack and security for key-dependent messages.

We constructed key-versatile signatures from NIZKs with strong security properties. An obvious area for further research is finding more efficient constructions, as the current construction can be seen more as a proof of concept than a practical candidate for implementation.

Stepping away from the versatility aspect of the signatures and focusing on the applications of signatures with such strong security properties, it is likely easier to find efficient signatures meeting simulatability and key-extractability for specific function families F, rather than generic constructions for any F as we have provided.

In the joint security setting, we showed that it is possible to add signing capability to an existing encryption scheme with no overhead in the size of the public key. This construction leaves the public key untouched, meaning no need for further certificate generation or dissemination. An open question in this area is that of finding the lowest possible cost in terms of public key overhead of adding encryption to existing signatures. This is a more difficult problem, however we might look to witness encryption [63, 16] for ideas.

We showed that key-versatile signatures inherit RKA security of the underlying one-way function family F. The problem of RKA security of the signatures can then be reduced to finding RKA-secure one-way functions We show that several known one-way functions are already RKA-secure. On the other hand, there are not many known one-way functions to analyse and our approach would be more valuable if there were more candidates available.

In our final application of key-versatile signatures we added integrity to encryption while maintaining key-dependent message security, creating a new primitive for authenticated and encrypted storage. With the rise of outsourced storage this is an important area in which to determine users' needs and build appropriate tools. This provides a novel application for KDM-secure PKE, further constructions of which would lead to more instantiations of our secure storage primitive. However here we are limited by the current techniques for achieving KDM security, which often involve storing a secret key in a specific way, meaning that the public key is not a one-way function of the secret key. Constructing new canonical PKE schemes with KDM security is therefore an interesting open problem.

Through these applications, key-versatile signatures have been shown to be a effective tool, likely to be of use in further applications.

Bibliography

[1] Masayuki Abe, Georg Fuchsbauer, Jens Groth, Kristiyan Haralambiev, and Miyako Ohkubo. “Structure-Preserving Signatures and Commitments to Group Elements”. In: *CRYPTO 2010*. Ed. by Tal Rabin. Vol. 6223. LNCS. Santa Barbara, CA, USA: Springer, Berlin, Germany, 2010, pp. 209–236.

[2] Tolga Acar, Mira Belenkiy, Mihir Bellare, and David Cash. “Cryptographic Agility and Its Relation to Circular Encryption”. In: *EUROCRYPT 2010*. Ed. by Henri Gilbert. Vol. 6110. LNCS. French Riviera: Springer, Berlin, Germany, 2010, pp. 403–422.

[3] Jee Hea An, Yevgeniy Dodis, and Tal Rabin. “On the Security of Joint Signature and Encryption”. In: *EUROCRYPT 2002*. Ed. by Lars R. Knudsen. Vol. 2332. LNCS. Amsterdam, The Netherlands: Springer, Berlin, Germany, 2002, pp. 83–107.

[4] Ross Anderson and Markus Kuhn. “Tamper Resistance – a Cautionary Note”. In: *Proceedings of the 2nd USENIX Workshop on Electronic Commerce*. Oakland, CA, USA: USENIX Association, 1996, pp. 1–11.

[5] Benny Applebaum. “Garbling XOR Gates “For Free” in the Standard Model”. In: *TCC 2013*. Ed. by Amit Sahai. Vol. 7785. LNCS. Tokyo, Japan: Springer, Berlin, Germany, 2013, pp. 162–181.

[6] Benny Applebaum. “Key-Dependent Message Security: Generic Amplification and Completeness”. In: *EUROCRYPT 2011*. Ed. by Kenneth G. Paterson. Vol. 6632. LNCS. Tallinn, Estonia: Springer, Berlin, Germany, 2011, pp. 527–546.

[7] Benny Applebaum, David Cash, Chris Peikert, and Amit Sahai. “Fast Cryptographic Primitives and Circular-Secure Encryption Based on Hard Learning Problems”. In: *CRYPTO 2009*. Ed. by Shai Halevi. Vol. 5677. LNCS. Santa Barbara, CA, USA: Springer, Berlin, Germany, 2009, pp. 595–618.

[8] Benny Applebaum, Danny Harnik, and Yuval Ishai. “Semantic Security under Related-Key Attacks and Applications”. In: *ICS 2011*. Ed. by Andrew Chi-Chih Yao. Tsinghua University Press, 2011.

[9] Michael Backes, Markus Dürmuth, and Dominique Unruh. “OAEP Is Secure under Key-Dependent Messages”. In: *ASIACRYPT 2008*. Ed. by Josef Pieprzyk. Vol. 5350. LNCS. Melbourne, Australia: Springer, Berlin, Germany, 2008, pp. 506–523.

[10] Michael Backes, Birgit Pfitzmann, and Andre Scedrov. “Key-dependent message security under active attacks - BRSIM/UC-soundness of Dolev-Yao-style encryption with key cycles”. In: *Journal of Computer Security* 16.5 (2008), pp. 497–530.

[11] Boaz Barak, Iftach Haitner, Dennis Hofheinz, and Yuval Ishai. “Bounded Key-Dependent Message Security”. In: *EUROCRYPT 2010*. Ed. by Henri Gilbert. Vol. 6110. LNCS. French Riviera: Springer, Berlin, Germany, 2010, pp. 423–444.

[12] Paulo S. L. M. Barreto and Michael Naehrig. “Pairing-Friendly Elliptic Curves of Prime Order”. In: *SAC 2005*. Ed. by Bart Preneel and Stafford Tavares. Vol. 3897. LNCS. Kingston, Ontario, Canada: Springer, Berlin, Germany, 2005, pp. 319–331.

[13] Mihir Bellare and David Cash. “Pseudorandom Functions and Permutations Provably Secure against Related-Key Attacks”. In: *CRYPTO 2010*. Ed. by Tal Rabin. Vol. 6223. LNCS. Santa Barbara, CA, USA: Springer, Berlin, Germany, 2010, pp. 666–684.

[14] Mihir Bellare, David Cash, and Rachel Miller. “Cryptography Secure against Related-Key Attacks and Tampering”. In: *ASIACRYPT 2011*. Ed. by Dong Hoon Lee and Xiaoyun Wang. Vol. 7073. LNCS. Seoul, South Korea: Springer, Berlin, Germany, 2011, pp. 486–503.

[15] Mihir Bellare and Shafi Goldwasser. “New Paradigms for Digital Signatures and Message Authentication Based on Non-Interative Zero Knowledge Proofs”. In: *CRYPTO’89*. Ed. by Gilles Brassard. Vol. 435. LNCS. Santa Barbara, CA, USA: Springer, Berlin, Germany, 1990, pp. 194–211.

[16] Mihir Bellare and Viet Tung Hoang. “Adaptive Witness Encryption and Asymmetric Password-based Cryptography”. Cryptology ePrint Archive, Report 2013/704. 2013.

[17] Mihir Bellare and Sriram Keelveedhi. "Authenticated and Misuse-Resistant Encryption of Key-Dependent Data". In: *CRYPTO 2011*. Ed. by Phillip Rogaway. Vol. 6841. LNCS. Santa Barbara, CA, USA: Springer, Berlin, Germany, 2011, pp. 610–629.

[18] Mihir Bellare and Tadayoshi Kohno. "A Theoretical Treatment of Related-Key Attacks: RKA-PRPs, RKA-PRFs, and Applications". In: *EUROCRYPT 2003*. Ed. by Eli Biham. Vol. 2656. LNCS. Warsaw, Poland: Springer, Berlin, Germany, 2003, pp. 491–506.

[19] Mihir Bellare, Sarah Meiklejohn, and Susan Thomson. "Key-Versatile Signatures and Applications: RKA, KDM and Joint Enc/Sig". In: *EUROCRYPT 2014*. Ed. by Phong Q. Nguyen and Elisabeth Oswald. Vol. 8441. LNCS. Copenhagen, Denmark: Springer, Berlin, Germany, 2014, pp. 496–513.

[20] Mihir Bellare and Chanathip Namprempre. "Authenticated Encryption: Relations among Notions and Analysis of the Generic Composition Paradigm". In: *Journal of Cryptology* 21.4 (Oct. 2008), pp. 469–491.

[21] Mihir Bellare, Kenneth G. Paterson, and Susan Thomson. "RKA Security beyond the Linear Barrier: IBE, Encryption and Signatures". In: *ASIACRYPT 2012*. Ed. by Xiaoyun Wang and Kazue Sako. Vol. 7658. LNCS. Beijing, China: Springer, Berlin, Germany, 2012, pp. 331–348.

[22] Mihir Bellare and Thomas Ristenpart. "Simulation without the Artificial Abort: Simplified Proof and Improved Concrete Security for Waters' IBE Scheme". In: *EUROCRYPT 2009*. Ed. by Antoine Joux. Vol. 5479. LNCS. Cologne, Germany: Springer, Berlin, Germany, 2009, pp. 407–424.

[23] Mihir Bellare and Phillip Rogaway. "The Security of Triple Encryption and a Framework for Code-Based Game-Playing Proofs". In: *EUROCRYPT 2006*. Ed. by Serge Vaudenay. Vol. 4004. LNCS. St. Petersburg, Russia: Springer, Berlin, Germany, 2006, pp. 409–426.

[24] Eli Biham. "New Types of Cryptoanalytic Attacks Using related Keys (Extended Abstract)". In: *EUROCRYPT'93*. Ed. by Tor Helleseth. Vol. 765. LNCS. Lofthus, Norway: Springer, Berlin, Germany, 1993, pp. 398–409.

[25] Eli Biham and Adi Shamir. "Differential Fault Analysis of Secret Key Cryptosystems". In: *CRYPTO'97*. Ed. by Burton S. Kaliski Jr. Vol. 1294. LNCS. Santa Barbara, CA, USA: Springer, Berlin, Germany, 1997, pp. 513–525.

[26] Nir Bitansky and Ran Canetti. "On Strong Simulation and Composable Point Obfuscation". In: *CRYPTO 2010*. Ed. by Tal Rabin. Vol. 6223. LNCS. Santa Barbara, CA, USA: Springer, Berlin, Germany, 2010, pp. 520–537.

[27] John Black, Phillip Rogaway, and Thomas Shrimpton. "Encryption-Scheme Security in the Presence of Key-Dependent Messages". In: *SAC 2002*. Ed. by Kaisa Nyberg and Howard M. Heys. Vol. 2595. LNCS. St. John's, Newfoundland, Canada: Springer, Berlin, Germany, 2003, pp. 62–75.

[28] Manuel Blum, Alfredo De Santis, Silvio Micali, and Giuseppe Persiano. "Noninteractive zero-knowledge". In: *SIAM Journal on Computing* 20.6 (1991), pp. 1084–1118.

[29] Dan Boneh and Xavier Boyen. "Efficient Selective Identity-Based Encryption Without Random Oracles". In: *Journal of Cryptology* 24.4 (Oct. 2011), pp. 659–693.

[30] Dan Boneh and Xavier Boyen. "Short Signatures Without Random Oracles and the SDH Assumption in Bilinear Groups". In: *Journal of Cryptology* 21.2 (Apr. 2008), pp. 149–177.

[31] Dan Boneh, Xavier Boyen, and Eu-Jin Goh. "Hierarchical Identity Based Encryption with Constant Size Ciphertext". In: *EUROCRYPT 2005*. Ed. by Ronald Cramer. Vol. 3494. LNCS. Aarhus, Denmark: Springer, Berlin, Germany, 2005, pp. 440–456.

[32] Dan Boneh, Ran Canetti, Shai Halevi, and Jonathan Katz. "Chosen-Ciphertext Security from Identity-Based Encryption". In: *SIAM J. Comput.* 36.5 (2007), pp. 1301–1328.

[33] Dan Boneh, Richard A. DeMillo, and Richard J. Lipton. "On the Importance of Checking Cryptographic Protocols for Faults (Extended Abstract)". In: *EUROCRYPT'97*. Ed. by Walter Fumy. Vol. 1233. LNCS. Konstanz, Germany: Springer, Berlin, Germany, 1997, pp. 37–51.

[34] Dan Boneh and Matthew Franklin. "Identity-Based Encryption from the Weil Pairing". In: *SIAM J. Comput.* 32.3 (2003), pp. 586–615.

[35] Dan Boneh and Matthew K. Franklin. "Identity Based Encryption from the Weil Pairing". In: *SIAM Journal on Computing* 32.3 (2003), pp. 586–615.

[36] Dan Boneh, Shai Halevi, Michael Hamburg, and Rafail Ostrovsky. "Circular-Secure Encryption from Decision Diffie-Hellman". In: *CRYPTO 2008*. Ed. by David Wagner. Vol. 5157. LNCS. Santa Barbara, CA, USA: Springer, Berlin, Germany, 2008, pp. 108–125.

[37] Dan Boneh and Jonathan Katz. "Improved Efficiency for CCA-Secure Cryptosystems Built Using Identity-Based Encryption". In: *CT-RSA 2005*. Ed. by Alfred Menezes. Vol. 3376. LNCS. San Francisco, CA, USA: Springer, Berlin, Germany, 2005, pp. 87–103.

[38] Dan Boneh, Ben Lynn, and Hovav Shacham. “Short Signatures from the Weil Pairing”. In: *ASIACRYPT 2001*. Ed. by Colin Boyd. Vol. 2248. LNCS. Gold Coast, Australia: Springer, Berlin, Germany, 2001, pp. 514–532.

[39] Xavier Boyen, Qixiang Mei, and Brent Waters. “Direct Chosen Ciphertext Security from Identity-Based Techniques”. In: *ACM CCS 05*. Ed. by Vijayalakshmi Atluri, Catherine Meadows, and Ari Juels. Alexandria, Virginia, USA: ACM Press, 2005, pp. 320–329.

[40] Zvika Brakerski, Shafi Goldwasser, and Yael Tauman Kalai. “Black-Box Circular-Secure Encryption beyond Affine Functions”. In: *TCC 2011*. Ed. by Yuval Ishai. Vol. 6597. LNCS. Providence, RI, USA: Springer, Berlin, Germany, 2011, pp. 201–218.

[41] Zvika Brakerski and Vinod Vaikuntanathan. “Fully Homomorphic Encryption from Ring-LWE and Security for Key Dependent Messages”. In: *CRYPTO 2011*. Ed. by Phillip Rogaway. Vol. 6841. LNCS. Santa Barbara, CA, USA: Springer, Berlin, Germany, 2011, pp. 505–524.

[42] Jon Callas, Lutz Donnerhacke, Hal Finney, David Shaw, and Rodney Thayer. OpenPGP Message Format. RFC 4880. http://www.ietf.org/rfc/rfc4880.txt. 2007.

[43] Jan Camenisch, Nishanth Chandran, and Victor Shoup. “A Public Key Encryption Scheme Secure against Key Dependent Chosen Plaintext and Adaptive Chosen Ciphertext Attacks”. In: *EUROCRYPT 2009*. Ed. by Antoine Joux. Vol. 5479. LNCS. Cologne, Germany: Springer, Berlin, Germany, 2009, pp. 351–368.

[44] Ran Canetti, Yael Tauman Kalai, Mayank Varia, and Daniel Wichs. “On Symmetric Encryption and Point Obfuscation”. In: *TCC 2010*. Ed. by Daniele Micciancio. Vol. 5978. LNCS. Zurich, Switzerland: Springer, Berlin, Germany, 2010, pp. 52–71.

[45] Melissa Chase, Markulf Kohlweiss, Anna Lysyanskaya, and Sarah Meiklejohn. “Malleable Signatures: Complex Unary Transformations and Delegatable Anonymous Credentials”. Cryptology ePrint Archive, Report 2013/179. `http://eprint.iacr.org/`. 2013.

[46] Melissa Chase and Anna Lysyanskaya. “On Signatures of Knowledge”. In: *CRYPTO 2006*. Ed. by Cynthia Dwork. Vol. 4117. LNCS. Santa Barbara, CA, USA: Springer, Berlin, Germany, 2006, pp. 78–96.

[47] Jean-Sébastien Coron, Marc Joye, David Naccache, and Pascal Paillier. “Universal Padding Schemes for RSA”. In: *CRYPTO 2002*. Ed. by Moti Yung. Vol. 2442. LNCS. Santa Barbara, CA, USA: Springer, Berlin, Germany, 2002, pp. 226–241.

[48] Ronald Cramer, Yevgeniy Dodis, Serge Fehr, Carles Padró, and Daniel Wichs. “Detection of Algebraic Manipulation with Applications to Robust Secret Sharing and Fuzzy Extractors”. In: *EUROCRYPT 2008*. Ed. by Nigel P. Smart. Vol. 4965. LNCS. Istanbul, Turkey: Springer, Berlin, Germany, 2008, pp. 471–488.

[49] Ronald Cramer and Victor Shoup. “Design and Analysis of Practical Public-Key Encryption Schemes Secure against Adaptive Chosen Ciphertext Attack”. In: *SIAM Journal on Computing* 33.1 (2003), pp. 167–226.

[50] Alfredo De Santis, Giovanni Di Crescenzo, Rafail Ostrovsky, Giuseppe Persiano, and Amit Sahai. “Robust Non-interactive Zero Knowledge”. In: *CRYPTO 2011*. Ed. by Joe Kilian. Vol. 2139. LNCS. Santa Barbara, CA, USA: Springer, Berlin, Germany, 2001, pp. 566–598.

[51] Alfredo De Santis, Giovanni Di Crescenzo, and Giuseppe Persiano. “Necessary and sufficient assumptions for non-interactive zero-knowledge proofs of knowledge for all np relations”. In: *Automata, Languages and Programming*. Springer. 2000, pp. 451–462.

[52] Alfredo De Santis and Giuseppe Persiano. “Zero-knowledge proofs of knowledge without interaction”. In: *Foundations of Computer Science, 1992. Proceedings., 33rd Annual Symposium on*. IEEE. 1992, pp. 427–436.

[53] Jean Paul Degabriele, Anja Lehmann, Kenneth G. Paterson, Nigel P. Smart, and Mario Strefler. “On the Joint Security of Encryption and Signature in EMV”. In: *CT-RSA 2012*. Ed. by Orr Dunkelman. Vol. 7178. LNCS. San Francisco, CA, USA: Springer, Berlin, Germany, 2012, pp. 116–135.

[54] Yevgeniy Dodis, Michael J. Freedman, Stanislaw Jarecki, and Shabsi Walfish. “Optimal Signcryption from Any Trapdoor Permutation”. Cryptology ePrint Archive, Report 2004/020. `http://eprint.iacr.org/`. 2004.

[55] Yevgeniy Dodis, Michael J. Freedman, Stanislaw Jarecki, and Shabsi Walfish. “Versatile Padding Schemes for Joint Signature and Encryption”. In: *ACM CCS 04*. Ed. by Vijayalakshmi Atluri, Birgit Pfitzmann, and Patrick McDaniel. Washington D.C., USA: ACM Press, 2004, pp. 344–353.

[56] Yevgeniy Dodis, Kristiyan Haralambiev, Adriana López-Alt, and Daniel Wichs. “Efficient Public-Key Cryptography in the Presence of Key Leakage”. In: *ASIACRYPT 2010*. Ed. by Masayuki Abe. Vol. 6477. LNCS. Singapore: Springer, Berlin, Germany, 2010, pp. 613–631.

[57] Danny Dolev, Cynthia Dwork, and Moni Naor. “Nonmalleable Cryptography”. In: *SIAM Journal on Computing* 30.2 (2000), pp. 391–437.

[58] *EMV Specifications, Version 4.2, Books 1–4.* http://www.emvco.com/. 2008.

[59] Jia Fan, Yuliang Zheng, and Xiaohu Tang. “A Single Key Pair is Adequate for the Zheng Signcryption”. In: *ACISP 11*. Ed. by Udaya Parampalli and Philip Hawkes. Vol. 6812. LNCS. Melbourne, Australia: Springer, Berlin, Germany, 2011, pp. 371–388.

[60] David Freeman, Michael Scott, and Edlyn Teske. “A Taxonomy of Pairing-Friendly Elliptic Curves”. In: *Journal of Cryptology* 23.2 (Apr. 2010), pp. 224–280.

[61] Steven D. Galbraith, Kenneth G. Paterson, and Nigel P. Smart. “Pairings for cryptographers”. In: *Discrete Applied Mathematics* 156.16 (2008), pp. 3113–3121.

[62] David Galindo, Javier Herranz, and Jorge L. Villar. “Identity-Based Encryption with Master Key-Dependent Message Security and Leakage-Resilience”. In: *ESORICS 2012*. Ed. by Sara Foresti, Moti Yung, and Fabio Martinelli. Vol. 7459. LNCS. Pisa, Italy: Springer, Berlin, Germany, 2012, pp. 627–642.

[63] Sanjam Garg, Craig Gentry, Amit Sahai, and Brent Waters. “Witness encryption and its applications”. In: *Proceedings of the 45th annual ACM symposium on Symposium on theory of computing*. ACM. 2013, pp. 467–476.

[64] Rosario Gennaro, Anna Lysyanskaya, Tal Malkin, Silvio Micali, and Tal Rabin. “Algorithmic Tamper-Proof (ATP) Security: Theoretical Foundations for Security against Hardware Tampering”. In: *TCC 2004*. Ed. by Moni Naor. Vol. 2951. LNCS. Cambridge, MA, USA: Springer, Berlin, Germany, 2004, pp. 258–277.

[65] Craig Gentry. “Practical Identity-Based Encryption Without Random Oracles”. In: *EUROCRYPT 2006*. Ed. by Serge Vaudenay. Vol. 4004. LNCS. St. Petersburg, Russia: Springer, Berlin, Germany, 2006, pp. 445–464.

[66] David Goldenberg and Moses Liskov. “On Related-Secret Pseudorandomness”. In: *TCC 2010*. Ed. by Daniele Micciancio. Vol. 5978. LNCS. Zurich, Switzerland: Springer, Berlin, Germany, 2010, pp. 255–272.

[67] Shafi Goldwasser, Silvio Micali, and Ronald L. Rivest. “A Digital Signature Scheme Secure Against Adaptive Chosen-message Attacks”. In: *SIAM Journal on Computing* 17.2 (Apr. 1988), pp. 281–308.

[68] Maria Isabel Gonzalez Vasco, Florian Hess, and Rainer Steinwandt. “Combined (Identity-Based) Public Key Schemes”. Cryptology ePrint Archive, Report 2008/466. `http://eprint.iacr.org/`. 2008.

[69] Vipul Goyal, Adam O’Neill, and Vanishree Rao. “Correlated-Input Secure Hash Functions”. In: *TCC 2011*. Ed. by Yuval Ishai. Vol. 6597. LNCS. Providence, RI, USA: Springer, Berlin, Germany, 2011, pp. 182–200.

[70] Jens Groth. “Simulation-Sound NIZK Proofs for a Practical Language and Constant Size Group Signatures”. In: *ASIACRYPT 2006*. Ed. by Xuejia Lai and Kefei Chen. Vol. 4284. LNCS. Shanghai, China: Springer, Berlin, Germany, 2006, pp. 444–459.

[71] Jens Groth and Rafail Ostrovsky. “Cryptography in the Multi-string Model”. In: *CRYPTO 2007*. Ed. by Alfred Menezes. Vol. 4622. LNCS. Santa Barbara, CA, USA: Springer, Berlin, Germany, 2007, pp. 323–341.

[72] Stuart Haber and Benny Pinkas. “Securely Combining Public-Key Cryptosystems”. In: *ACM CCS 01*. Philadelphia, PA, USA: ACM Press, 2001, pp. 215–224.

[73] Kristiyan Haralambiev. “Efficient Cryptographic Primitives for Non-Interactive Zero-Knowledge Proofs and Applications”. PhD thesis. New York University, May 2011.

[74] Florian Hess. “Efficient Identity Based Signature Schemes Based on Pairings”. In: *SAC 2002*. Ed. by Kaisa Nyberg and Howard M. Heys. Vol. 2595. LNCS. St. John’s, Newfoundland, Canada: Springer, Berlin, Germany, 2003, pp. 310–324.

[75] Dennis Hofheinz. “Circular Chosen-Ciphertext Security with Compact Ciphertexts”. In: *EUROCRYPT*. Vol. 7881. Lecture Notes in Computer Science. To appear. Springer, 2013, pp. 520–536.

[76] Dennis Hofheinz and Eike Kiltz. “Programmable Hash Functions and Their Applications”. In: *CRYPTO 2008*. Ed. by David Wagner. Vol. 5157. LNCS. Santa Barbara, CA, USA: Springer, Berlin, Germany, 2008, pp. 21–38.

[77] John Kelsey, Bruce Schneier, and David Wagner. “Protocol Interactions and the Chosen Protocol Attack”. In: *Security Protocols Workshop*. Ed. by Bruce Christianson, Bruno Crispo, T. Mark A. Lomas, and Michael Roe. Vol. 1361. LNCS. Springer, Berlin, Germany, 1997, pp. 91–104.

[78] Eike Kiltz. “Chosen-Ciphertext Secure Key-Encapsulation Based on Gap Hashed Diffie-Hellman”. In: *PKC 2007*. Ed. by Tatsuaki Okamoto and Xiaoyun Wang. Vol. 4450. LNCS. Beijing, China: Springer, Berlin, Germany, 2007, pp. 282–297.

[79] Vlastimil Klíma and Tomás Rosa. “Further Results and Considerations on Side Channel Attacks on RSA”. In: *CHES 2002*. Ed. by Burton S. Kaliski Jr., Çetin Kaya Koç, and Christof Paar. Vol. 2523. LNCS. Redwood Shores, California, USA: Springer, Berlin, Germany, 2002, pp. 244–259.

[80] Lars R. Knudsen. “Cryptanalysis of LOKI91”. In: *AUSCRYPT’92*. Ed. by Jennifer Seberry and Yuliang Zheng. Vol. 718. LNCS. Gold Coast, Queensland, Australia: Springer, Berlin, Germany, 1992, pp. 196–208.

[81] Yuichi Komano and Kazuo Ohta. “Efficient Universal Padding Techniques for Multiplicative Trapdoor One-Way Permutation”. In: *CRYPTO 2003*. Ed. by Dan Boneh. Vol. 2729. LNCS. Santa Barbara, CA, USA: Springer, Berlin, Germany, 2003, pp. 366–382.

[82] Kaoru Kurosawa and Yvo Desmedt. “A New Paradigm of Hybrid Encryption Scheme”. In: *CRYPTO 2004*. Ed. by Matthew Franklin. Vol. 3152. LNCS. Santa Barbara, CA, USA: Springer, Berlin, Germany, 2004, pp. 426–442.

[83] Chung Ki Li, Guomin Yang, Duncan S. Wong, Xiaotie Deng, and Sherman S. M. Chow. “An Efficient Signcryption Scheme with Key Privacy”. In: *EuroPKI 2007*. Ed. by Javier Lopez, Pierangela Samarati, and Josep L. Ferrer. Vol. 4582. LNCS. Springer, Berlin, Germany, 2007, pp. 78–93.

[84] Benoît Libert and Jean-Jacques Quisquater. “Efficient Signcryption with Key Privacy from Gap Diffie-Hellman Groups”. In: *PKC 2004*. Ed. by Feng Bao, Robert Deng, and Jianying Zhou. Vol. 2947. LNCS. Singapore: Springer, Berlin, Germany, 2004, pp. 187–200.

[85] Benoît Libert and Jean-Jacques Quisquater. “Improved Signcryption from q-Diffie-Hellman Problems”. In: *SCN 04*. Ed. by Carlo Blundo and Stelvio Cimato. Vol. 3352. LNCS. Amalfi, Italy: Springer, Berlin, Germany, 2004, pp. 220–234.

[86] Stefan Lucks. “Ciphers Secure against Related-Key Attacks”. In: *FSE 2004*. Ed. by Bimal K. Roy and Willi Meier. Vol. 3017. LNCS. New Delhi, India: Springer, Berlin, Germany, 2004, pp. 359–370.

[87] Tal Malkin, Isamu Teranishi, and Moti Yung. “Efficient Circuit-Size Independent Public Key Encryption with KDM Security”. In: *EUROCRYPT 2011*. Ed. by Kenneth G. Paterson. Vol. 6632. LNCS. Tallinn, Estonia: Springer, Berlin, Germany, 2011, pp. 507–526.

[88] John Malone-Lee. “Signcryption with Non-interactive Non-repudiation”. In: *Des. Codes Cryptography* 37.1 (2005), pp. 81–109.

[89] Takahiro Matsuda, Kanta Matsuura, and Jacob C. N. Schuldt. “Efficient Constructions of Signcryption Schemes and Signcryption Composability”. In: *INDOCRYPT 2009*. Ed. by Bimal K. Roy and Nicolas Sendrier. Vol. 5922. LNCS. New Delhi, India: Springer, Berlin, Germany, 2009, pp. 321–342.

[90] M. González Muñiz and R. Steinwandt. “Security of signature schemes in the presence of key-dependent messages”. In: *Tatra Mt. Math. Publ.* 47 (2010), pp. 15–29.

[91] Kenneth G. Paterson, Jacob C. N. Schuldt, Martijn Stam, and Susan Thomson. “On the Joint Security of Encryption and Signature, Revisited”. In: *ASIACRYPT 2011*. Ed. by Dong Hoon Lee and Xiaoyun Wang. Vol. 7073. LNCS. Seoul, South Korea: Springer, Berlin, Germany, 2011, pp. 161–178.

[92] Duong Hieu Phan and David Pointcheval. “About the Security of Ciphers (Semantic Security and Pseudo-Random Permutations)”. In: *SAC 2004*. Ed. by Helena Handschuh and Anwar Hasan. Vol. 3357. LNCS. Waterloo, Ontario, Canada: Springer, Berlin, Germany, 2004, pp. 182–197.

[93] Phillip Rogaway. “Authenticated-Encryption With Associated-Data”. In: *ACM CCS 02*. Ed. by Vijayalakshmi Atluri. Washington D.C., USA: ACM Press, 2002, pp. 98–107.

[94] John Rompel. “One-way functions are necessary and sufficient for secure signatures”. In: *22nd ACM STOC*. Baltimore, Maryland, USA: ACM Press, 1990, pp. 387–394.

[95] Karl Rubin and Alice Silverberg. “Compression in Finite Fields and Torus-Based Cryptography”. In: *SIAM J. Comput.* 37.5 (2008), pp. 1401–1428.

[96] Amit Sahai. “Non-Malleable Non-Interactive Zero Knowledge and Adaptive Chosen-Ciphertext Security”. In: *40th FOCS*. New York, New York, USA: IEEE Computer Society Press, 1999, pp. 543–553.

[97] Sergei P. Skorobogatov and Ross J. Anderson. “Optical Fault Induction Attacks”. In: *CHES 2002*. Ed. by Burton S. Kaliski Jr., Çetin Kaya Koç, and Christof Paar. Vol. 2523. LNCS. Redwood Shores, California, USA: Springer, Berlin, Germany, 2002, pp. 2–12.

[98] Brent R. Waters. “Efficient Identity-Based Encryption Without Random Oracles”. In: *EUROCRYPT 2005*. Ed. by Ronald Cramer. Vol. 3494. LNCS. Aarhus, Denmark: Springer, Berlin, Germany, 2005, pp. 114–127.

[99] Hoeteck Wee. “Public Key Encryption against Related Key Attacks”. In: *PKC 2012*. Ed. by Marc Fischlin, Johannes Buchmann, and Mark Manulis. Vol. 7293. LNCS. Darmstadt, Germany: Springer, Berlin, Germany, 2012, pp. 262–279.

[100] Yuliang Zheng. “Digital Signcryption or How to Achieve Cost(Signature & Encryption) $<<$ Cost(Signature) + Cost(Encryption)”. In: *CRYPTO’97*. Ed. by Burton S. Kaliski Jr. Vol. 1294. LNCS. Santa Barbara, CA, USA: Springer, Berlin, Germany, 1997, pp. 165–179.