Key-Versatile Signatures and Applications: RKA, KDM and Joint Enc/Sig

Mihir Bellare[1] Sarah Meiklejohn[2] Susan Thomson[3]

May 2013

Abstract

Given an *arbitrary* one-way function F, is it possible to design a signature scheme where the secret key is an input x to F and the public key is $y = F(x)$? We show that signatures that are "key-versatile" in this sense, while also meeting stronger-than-usual security conditions we define, enable us to add signature-based integrity that is "for-free" in terms of key material, meaning we can sign with keys already in use for another purpose without impacting the security of the original purpose or in turn being impacted by it. We show applications across diverse areas including (1) security against related-key attack (RKA) (2) security for key-dependent messages (KDM), and (3) joint encryption and signing. We show how to build key-versatile signature schemes and then obtain new results in all these application domains in a modular way.

[1] Department of Computer Science & Engineering, University of California San Diego, 9500 Gilman Drive, La Jolla, California 92093, USA. Email: `mihir@eng.ucsd.edu`. URL: `cseweb.ucsd.edu/~mihir/`. Supported in part by NSF grants CNS-0904380, CCF-0915675, CNS-1116800 and CNS-1228890.

[2] Department of Computer Science & Engineering, University of California San Diego, 9500 Gilman Drive, La Jolla, California 92093, USA. Email: `smeiklej@eng.ucsd.edu`. URL: `cseweb.ucsd.edu/~smeiklejohn/`. Supported in part by NSF grant NSF-1237264.

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

Contents

1 Introduction

Let F be a one-way function. Let us say that a signature scheme is *F-keyed* if the secret signing key is a random domain point x for F and the public verification key is its image $y = F(x)$. Many known signature schemes are F-keyed for some *particular* F chosen to make the scheme work. For example, Schnorr signatures [59] are F-keyed for $F(x) = g^x$ where g generates the underlying cyclic group. In this paper, we consider the converse question. Namely, given an *arbitrary* one-way function F, can we construct an F-keyed signature scheme? Further, raising the bar, we will ask our "key-versatile" signatures to meet simulatability and key-extractability conditions that are significantly stronger than the usual unforgeability.

Why do we want this? Briefly, it gives us signing that is "for free" in terms of key material. We can take keys, already in use for some purpose and not created or structured for signing, and sign with them, *without impacting the security of the original use of the keys.* This ability to add authenticity without changing the key structure allows us in particular to retain security against related-key attack (RKA) or for key-dependent messages (KDM), so that we can expand our capabilities in these challenging areas in a modular way, leveraging previous work rather than starting from scratch. We begin, however, with a more direct application, namely adding signing to encryption with *zero* overhead in the size of the public key.

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.

Joining signatures to encryption with zero public-key overhead. Suppose Alice has keys (sk_e, pk_e) for a public-key encryption scheme and wants to also have signing capability. Certainly, she could pick new and separate keys (sk_s, pk_s) enabling her to use her favorite signature scheme. However, it means that Alice's public key, now $pk = (pk_e, pk_s)$, has doubled in size. Practitioners ask if one can do better. We want a joint encryption and signature (JES) scheme [47, 55], where there is a single key-pair (sk, pk) used for both encryption and signing. We aim to minimize the public-key overhead, (loosely) defined as the size of pk minus the size of the public key pk_e of the underlying encryption scheme.

Haber and Pinkas [47] initiated an investigation of JES. They note that the key re-use requires defining and achieving new notions of security particular to JES: signatures should remain unforgeable even in the presence of a decryption oracle, and encryption should retain IND-CCA privacy even in the presence of a signing oracle. In the random oracle model [16], *specific* IND-CCA-secure public-key encryption schemes have been presented where signing can be added with no public-key overhead [47, 34, 51]. In the standard model, encryption schemes have been presented that allow signing with a public-key overhead lower than that of the "Cartesian product" solution of just adding a separate signing key [47, 55], with the best results, from [55], using IBE or combining encryption and signature schemes of [28, 24].

All these results, however, 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.[1] We emphasize 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.[2] The assumed security of the encryption scheme means this function is one-way. Now, we

[1] 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.

[2] Not all encryption schemes will directly derive the public key as a deterministic function of the secret key, although

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 giving 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. See Section 4.

New RKA-secure signatures. In a related-key attack (RKA) [50, 18, 13, 9] an adversary can modify a stored secret key and observe outcomes of the cryptographic primitive under the modified key. Such attacks may be mounted by tampering [26, 19, 42], so RKA security improves resistance to side-channel attacks. Achieving proven security against RKAs, however, is broadly recognized as very challenging. This has lead several authors [43, 9] to suggest that we "bootstrap," building higher-level Φ-RKA-secure[3] primitives from lower-level Φ-RKA-secure primitives. In this vein, [9] 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 [8]. This has lead to direct (non-bootstrapping) constructions of Φ-RKA signatures for classes Φ of polynomials over certain specific pairing groups [15].

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.[4] The difference is significant because building Φ-RKA one-way functions under standard assumptions is easy. Adapting the key-malleability technique of [8], we show that many natural one-way functions are Φ-RKA secure *assuming nothing more than their standard one-wayness*. In particular this is true for discrete exponentiation over an arbitrary group and for the one-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. See Section 5.

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., [21, 27, 5, 4, 31, 32, 20, 7, 53, 3, 29, 30, 12, 41, 49]). 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. 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 we carefully formalize the corresponding primitive, providing both syntax and notions of security for key-dependent messages. Briefly, Alice will use a secret key sk to turn her message M into an encrypted and authenticated "data" object that she stores on the server. The server will be 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 will explain in more depth below, this goal is different from signcryption [61], authenticated

many, including Cramer-Shoup [36], 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.

[3] As per the framework of [13, 9], security is parameterized by the class of functions Φ that the adversary is allowed to apply to the key. Security is never possible for the class of all functions [13], so we seek results for specific Φ.

[4] 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 [43].

public-key encryption [2] and authenticated symmetric encryption [14, 56], 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 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.

GETTING F-KEYED SIGNATURES. In Section 3 we define F-keyed signature schemes and show how to construct them for arbitrary one-way F. This enables us to realize the above applications.

Our simulatability condition, adapting [1, 33], 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 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 3.1, showing these conditions imply not just standard but strong unforgeability, functions 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 [38], in turn a simplification of the NIZK-based signature scheme of [11] that is made possible by the significant advances in NIZK technology since [11]. While [38] prove leakage-resilience of their scheme, we prove simulatability and key-extractability. The underlying SE NIZKs are a variant of simulation-sound extractable NIZKs [37, 45, 46] introduced by [38] (they called them tSE NIZKs) and shown by [38, 48] to be achievable for all of **NP** under standard assumptions.

DISCUSSION AND RELATED WORK. Signcryption [61], authenticated public-key encryption [2], JES [47, 55] and our secure storage goal all have in common that both encryption and signature are involved. However, in signcryption and authenticated public-key encryption, 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 and authenticated public-key encryption, 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 [40] is however similar to secure storage, minus the key-dependent message aspect. Authenticated symmetric encryption [14, 56] 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 [12, 6].

KDM-secure signatures were studied in [54], 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 modeling 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.

2 Notation

The empty string is denoted by ε. If x is a (binary) string then $|x|$ is its length. If S is a finite set then $|S|$ denotes its size and $s \leftarrow\$ S$ denotes picking an element uniformly from S and assigning it to s. We denote by $\lambda \in \mathbb{N}$ the security parameter and by 1^λ its unary representation.

Algorithms are randomized unless otherwise indicated. "PT" stands for "polynomial-time," whether for randomized algorithms or deterministic. By $y \leftarrow A(x_1, \ldots; R)$, we denote the operation of running algorithm A on inputs $x_1, \ldots$ and coins R and letting y denote the output. By $y \leftarrow\$ A(x_1, \ldots)$, we denote the operation of letting $y \leftarrow A(x_1, \ldots; R)$ for random R. We denote by $[A(x_1, \ldots)]$ the set of points 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 (eg. Figure 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. The boolean flag bad, if used in a game, is assumed initialized to false.

3 Key-versatile signing signatures

We define F-keyed signature schemes, for F a family of functions rather than the single function F used for simplicity in Section 1. The requirement is that the secret key sk is an input for an instance fp of the family and the public key $pk = \mathsf{F.Ev}(1^\lambda, fp, sk)$ is the corresponding image under this instance, the instance fp 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 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 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 [57], 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.

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 $(sk, pk) \leftarrow\$ \mathsf{DS.Kg}(1^\lambda, pp)$ a user can generate a secret signing key sk and corresponding public verification key pk; via $\sigma \leftarrow\$ \mathsf{DS.Sig}(1^\lambda, pp, sk, M)$ the signer can generate a signature σ on a message $M \in \{0,1\}^*$; via $d \leftarrow \mathsf{DS.Ver}(1^\lambda, pp, pk, M, \sigma)$ a

```
main $\mathrm{SIM}^{A}_{\mathsf{DS,F}}(\lambda)$
$b \leftarrow_{\$} \{0,1\}$
$(fp, ap_1) \leftarrow_{\$} \mathsf{DS.Pg}(1^\lambda)$ ; $pp_1 \leftarrow (fp, ap_1)$
$(ap_0, std, xtd) \leftarrow_{\$} \mathsf{DS.SimPg}(1^\lambda)$ ; $pp_0 \leftarrow (fp, ap_0)$
$b' \leftarrow_{\$} A^{\mathrm{SIGN}}(1^\lambda, pp_b)$
Ret $(b = b')$

$\mathrm{SIGN}(sk, M)$
If $sk \notin \mathsf{F.Dom}(1^\lambda, fp)$ then return $\perp$
$pk \leftarrow \mathsf{F.Ev}(1^\lambda, fp, sk)$
If $b = 1$ then $\sigma \leftarrow_{\$} \mathsf{DS.Sig}(1^\lambda, pp_1, sk, M)$
Else $\sigma \leftarrow_{\$} \mathsf{DS.SimSig}(1^\lambda, pp_0, std, pk, M)$
Ret $\sigma$
```

```
main $\mathrm{EXT}^{A}_{\mathsf{DS,F}}(\lambda)$
$fp \leftarrow_{\$} \mathsf{F.Pg}(1^\lambda)$ ; $Q \leftarrow \emptyset$
$(ap, std, xtd) \leftarrow_{\$} \mathsf{DS.SimPg}(1^\lambda)$ ; $pp \leftarrow (fp, ap)$
$(pk, M, \sigma) \leftarrow_{\$} A^{\mathrm{SIGN}}(1^\lambda, pp)$
If $pk \notin \mathsf{F.Rng}(1^\lambda, fp)$ then Ret false
If not $\mathsf{DS.Ver}(1^\lambda, pp, pk, M, \sigma)$ then Ret false
If $(pk, M, \sigma) \in Q$ then Ret false
$sk \leftarrow_{\$} \mathsf{DS.Ext}(1^\lambda, pp, xtd, pk, M, \sigma)$
Ret $(\mathsf{F.Ev}(1^\lambda, fp, sk) \neq pk)$

$\mathrm{SIGN}(sk, M)$
If $sk \notin \mathsf{F.Dom}(1^\lambda, fp)$ then return $\perp$
$pk \leftarrow \mathsf{F.Ev}(1^\lambda, fp, sk)$
$\sigma \leftarrow_{\$} \mathsf{DS.SimSig}(1^\lambda, pp, std, pk, M)$ ; $Q \leftarrow Q \cup \{(pk, M, \sigma)\}$
Ret $\sigma$
```

Figure 1: **Games defining security of F-keyed signature scheme DS.** Left: Game defining simulatability. Right: Game defining key-extractability.

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 $\mathsf{DS.Ver}(1^\lambda, pp, pk, M, \mathsf{DS.Sig}(1^\lambda, pp, sk, M)) = \mathsf{true}$ for all $\lambda \in \mathbb{N}$, all $pp \in [\mathsf{DS.Pg}(1^\lambda)]$, all $(sk, pk) \in [\mathsf{DS.Kg}(1^\lambda, pp)]$, and all M.

Function families. A function family F specifies the following. Via $fp \leftarrow_{\$} \mathsf{F.Pg}(1^\lambda)$ one can in PT generate a description fp of a function $\mathsf{F.Ev}(1^\lambda, fp, \cdot)\colon \mathsf{F.Dom}(1^\lambda, fp) \to \mathsf{F.Rng}(1^\lambda, fp)$. We assume that membership of x in the non-empty domain $\mathsf{F.Dom}(1^\lambda, fp)$ can be tested in time polynomial in $1^\lambda, fp, x$ and one can in time polynomial in $1^\lambda, fp$ sample a point $x \leftarrow_{\$} \mathsf{F.Dom}(1^\lambda, fp)$ from the domain $\mathsf{F.Dom}(1^\lambda, fp)$. The deterministic evaluation algorithm F.Ev is PT. The range is defined by $\mathsf{F.Rng}(1^\lambda, fp) = \{\ \mathsf{F.Ev}(1^\lambda, fp, x)\ :\ x \in \mathsf{F.Dom}(1^\lambda, fp)\ \}$. 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 OWF if $\mathbf{Adv}^{\mathrm{ow}}_{\mathsf{F},I}(\cdot)$ is negligible for all PT I, where $\mathbf{Adv}^{\mathrm{ow}}_{\mathsf{F},I}(\lambda) = \Pr[\mathsf{F.Ev}(1^\lambda, fp, x') = y]$ under the experiment $fp \leftarrow_{\$} \mathsf{F.Pg}(1^\lambda)$; $x \leftarrow_{\$} \mathsf{F.Dom}(1^\lambda, fp)$; $y \leftarrow \mathsf{F.Ev}(1^\lambda, fp, x)$; $x' \leftarrow_{\$} I(1^\lambda, fp, y)$.

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 $pp = (fp, ap)$ consisting of parameters fp for F and auxiliary parameters ap, these independently generated. Formally, there is a PT *auxiliary parameter generation* algorithm APg such that $\mathsf{DS.Pg}(1^\lambda)$ picks $fp \leftarrow_{\$} \mathsf{F.Pg}(1^\lambda)$; $ap \leftarrow_{\$} \mathsf{APg}(1^\lambda)$ and returns (fp, ap).
- Key compatibility: The signing key sk is a random point in the domain of F.Ev and the verifying key pk is its image under F.Ev. Formally, $\mathsf{DS.Kg}(1^\lambda, (fp, ap))$ picks $sk \leftarrow_{\$} \mathsf{F.Dom}(1^\lambda, fp)$, lets $pk \leftarrow \mathsf{F.Ev}(1^\lambda, fp, sk)$ and returns (sk, pk). (DS.Kg ignores the auxiliary parameters ap, meaning the keys do not depend on it.)

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 $pk = \mathsf{F.Ev}(1^\lambda, fp, sk)$ 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 DS.SimPg (the auxiliary parameter simulator) and $\mathsf{DS.SimSig}$ (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 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 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 that of [33] which in turn strengthened that of [1]. 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 *fp*.

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 [44]. 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 3.1 *Let* DS *be an* F*-keyed signature scheme that is simulatable and key-extractable. If* F *is one-way then* DS *is strongly unforgeable.*

In Appendix A we recall the definition of strong unforgeability and formally prove the above. Here we sketch the intuition. Given an adversary A against unforgeability, we build an inverter I for F. On input $1^\lambda, \mathit{fp}, \mathit{pk}$, adversary I generates simulated auxiliary parameters *ap* together with simulation and extraction trapdoors. It now runs A with parameters $(\mathit{fp}, \mathit{ap})$, 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*, a pre-image of *pk* under $\mathsf{F.Ev}(1^\lambda, \mathit{fp}, \cdot)$. The simulation and key-extractability conditions are invoked to show that I succeeds with almost the same probability as A. This involves the construction of adversaries A_1, A_2 for the two conditions. A question here is that these adversaries can only make signing queries under secret keys that they know, so how do they proceed not knowing *sk*? The answer is that they will themselves run key-generation to get $(\mathit{sk}, \mathit{pk})$ and then run A on the latter. Now, when A makes a signing query M, adversaries A_1, A_2 can answer by invoking their own signing oracles on sk, M.

A 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 paper.

CONSTRUCTION. A *key-versatile signing schema* is a transform $\mathbf{KvS}$ 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 possible under standard assumptions.

Theorem 3.2 *Assume there exist SE NIZK systems for all of* $\mathbf{NP}$*. Then there is a key-versatile signing schema* $\mathbf{KvS}$ *such that if* F *is any family of functions then the signature scheme* $\mathsf{DS} = \mathbf{KvS}[\mathsf{F}]$ *is simulatable and key-extractable.*

In Appendix B we recall the definition of a SE (Simulation Extractable) NIZK system. SE was called tSE in [38] and is a variant of NIZK-security notions from [45, 37, 58]. We then specify the construction and prove it has the claimed properties. Here we sketch the construction and its history.

The scheme is simple. We define the relation $\mathsf{R}((1^\lambda, \mathit{fp}, \mathit{pk}, M), \mathit{sk})$ to return true iff $\mathsf{F.Ev}(1^\lambda, \mathit{fp}, \mathit{sk}) = \mathit{pk}$. 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 $(1^\lambda, fp, pk, M)$. 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 $(1^\lambda, fp, pk, M)$, 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. The formal proof of Theorem 3.2 in Appendix B shows simulatability of the F-keyed signature scheme based on the zero-knowledge property of the NIZK and key-extractability of the F-keyed signature scheme based on extractability of the NIZK, both in a natural and simple way.

A similar construction of signatures was given in [38] starting from a leakage-resilient hard relations 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 labeled 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 [38] establishing leakage resilience and unforgeability of the signature and our work showing simulatability and key-extractability. The technique of [38] was also used by [33] to construct malleable signatures. Going back further, the first NIZK-based signature scheme was that of [11]. 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 [37, 45, 46, 38].

4 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 scheme [36], the Kurosawa-Desmedt scheme [52], or the DDN scheme [39], but it could be any other IND-CCA-secure scheme as well. Alice has already established a key-pair (sk_e, pk_e) for this scheme, allowing anyone to send her ciphertexts computed under pk_e that she can decrypt under sk_e. She wants now to add signature capability. This is easily done. She can create a key-pair (sk_s, pk_s) for her favorite signature scheme and sign an arbitrary message M under sk_s, verification being possible given pk_s. The difficulty is that her public key is now $pk = (pk_e, pk_s)$. 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 (JES) scheme where Alice has a single key-pair (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 $pk = (pk_e, pk_s)$. Perhaps surprisingly, we show how to construct a JES scheme with pk-overhead zero, meaning pk is unchanged, remaining pk_e. We not only manage to use sk_e to sign and pk_e 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 [47, 55]. 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_e to pk_e and using a simulatable and key-extractable F-keyed signature scheme with the keys remaining (sk_e, pk_e).

JES SCHEMES. A joint encryption and signature (JES) scheme JES specifies the following PT algorithms: via $jp \leftarrow\$ \mathsf{JES.Pg}(1^\lambda)$ one generates public parameters jp common to all users; via $(sk, pk) \leftarrow\$ \mathsf{JES.Kg}(1^\lambda, jp)$ a user can generate a secret (signing and decryption) key sk and corresponding public (verification and encryption) key pk; via $\sigma \leftarrow\$ \mathsf{JES.Sig}(1^\lambda, jp, sk, M)$ the user can generate a signature σ on a message $M \in \{0,1\}^*$; via $d \leftarrow \mathsf{JES.Ver}(1^\lambda, jp, pk, M, \sigma)$ a verifier can deterministi-

```
main $\mathrm{IND}^A_{\mathsf{JES}}(\lambda)$
$b \leftarrow_{\$} \{0,1\}$ ; $C^* \leftarrow \bot$
$jp \leftarrow_{\$} \mathsf{JES.Pg}(1^\lambda)$ ; $(pk, sk) \leftarrow_{\$} \mathsf{JES.Kg}(1^\lambda, jp)$
$b' \leftarrow_{\$} A^{\mathrm{Dec},\mathrm{Sign},\mathrm{LR}}(1^\lambda, jp, pk)$
Ret $(b = b')$

proc $\mathrm{Dec}(C)$
If $(C = C^*)$ then Ret $\bot$
Ret $M \leftarrow \mathsf{JES.Dec}(1^\lambda, jp, sk, C)$

proc $\mathrm{Sign}(M)$
Ret $\sigma \leftarrow_{\$} \mathsf{JES.Sig}(1^\lambda, jp, sk, M)$

proc $\mathrm{LR}(M_0, M_1)$
If $(|M_0| \neq |M_1|)$ then Ret $\bot$
$C^* \leftarrow_{\$} \mathsf{JES.Enc}(1^\lambda, jp, pk, M_b)$
Ret $C^*$
```

```
main $\mathrm{SUF}^A_{\mathsf{JES}}(\lambda)$
$Q \leftarrow \emptyset$
$jp \leftarrow_{\$} \mathsf{JES.Pg}(1^\lambda)$ ; $(pk, sk) \leftarrow_{\$} \mathsf{JES.Kg}(1^\lambda, jp)$
$(M, \sigma) \leftarrow_{\$} A^{\mathrm{Sign},\mathrm{Dec}}(1^\lambda, jp, pk)$
Ret $(\mathsf{JES.Ver}(1^\lambda, jp, pk, M, \sigma)$ and $(M, \sigma) \notin Q)$

proc $\mathrm{Sign}(M)$
$\sigma \leftarrow_{\$} \mathsf{JES.Sig}(1^\lambda, jp, sk, M)$
$Q \leftarrow Q \cup \{(M, \sigma)\}$
Ret $\sigma$

proc $\mathrm{Dec}(C)$
Ret $M \leftarrow \mathsf{Dec}(1^\lambda, jp, sk, C)$
```

Figure 2: **Games defining security of joint encryption and signature scheme JES.** Left: Game IND defining privacy against chosen-ciphertext attack in the presence of a signing oracle. Right: Game SUF defining strong unforgeability in the presence of a decryption oracle.

cally produce a decision $d \in \{\mathsf{true}, \mathsf{false}\}$ regarding whether σ is a valid signature of M under pk; via $C \leftarrow_{\$} \mathsf{JES.Enc}(1^\lambda, jp, pk, M)$ anyone can generate a ciphertext C encrypting message M under pk; via $M \leftarrow \mathsf{JES.Dec}(1^\lambda, jp, sk, C)$ the user can deterministically decrypt ciphertext C to get a value $M \in \{0,1\}^* \cup \{\bot\}$. Correctness requires that $\mathsf{JES.Ver}(1^\lambda, jp, pk, M, \mathsf{JES.Sig}(1^\lambda, jp, sk, M)) = \mathsf{true}$ and that $\mathsf{JES.Dec}(1^\lambda, jp, sk, \mathsf{JES.Enc}(1^\lambda, jp, pk, M)) = M$ for all $\lambda \in \mathbb{N}$, all $jp \in [\mathsf{JES.Pg}(1^\lambda)]$, all $(sk, pk) \in [\mathsf{JES.Kg}(1^\lambda, jp)]$, and all $M \in \{0,1\}^*$. We say that JES is SUF-secure if $\mathbf{Adv}^{\mathrm{suf}}_{\mathsf{JES},A}(\cdot)$ is negligible for all PT adversaries A, where $\mathbf{Adv}^{\mathrm{suf}}_{\mathsf{JES},A}(\lambda) = \Pr[\mathrm{SUF}^A_{\mathsf{JES}}(\lambda)]$ and game SUF is on the right-hand side of Figure 2. This represents (strong) unforgeability of the signature in the presence of a decryption oracle. We say that JES is IND-secure if $\mathbf{Adv}^{\mathrm{ind}}_{\mathsf{JES},A}(\cdot)$ is negligible for all PT adversaries A, where $\mathbf{Adv}^{\mathrm{ind}}_{\mathsf{JES},A}(\lambda) = 2\Pr[\mathrm{IND}^A_{\mathsf{JES}}(\lambda)] - 1$ and game IND is on the left-hand side of Figure 2. Here the adversary is allowed only one query to LR. This represents privacy under chosen-ciphertext attack in the presence of a signing oracle. These definitions are from [47, 55].

The base PKE scheme. We are given a public-key encryption scheme PKE, specifying the following PT algorithms: via $fp \leftarrow_{\$} \mathsf{PKE.Pg}(1^\lambda)$ one generates public parameters; via $(sk, pk) \leftarrow_{\$} \mathsf{PKE.Kg}(1^\lambda, fp)$ a user generates a decryption key sk and encryption key pk; via $C \leftarrow_{\$} \mathsf{PKE.Enc}(1^\lambda, fp, pk, M)$ anyone can generate a ciphertext C encrypting a message M under pk; and via $M \leftarrow \mathsf{PKE.Dec}(1^\lambda, fp, sk, C)$ a user can deterministically decrypt a ciphertext C to get a value $M \in \{0,1\}^* \cup \{\bot\}$. Correctness requires that $\mathsf{PKE.Dec}(1^\lambda, fp, sk, \mathsf{PKE.Enc}(1^\lambda, fp, pk, M)) = M$ for all $\lambda \in \mathbb{N}$, all $fp \in [\mathsf{PKE.Pg}(1^\lambda)]$, all $(sk, pk) \in [\mathsf{PKE.Kg}(1^\lambda, fp)]$, and all $M \in \{0,1\}^*$. We assume that PKE meets the usual notion of IND-CCA security.

Let us say that PKE is *canonical* if the operation $(sk, pk) \leftarrow_{\$} \mathsf{PKE.Kg}(1^\lambda, fp)$ is done by picking sk at random from a finite, non-empty set we denote $\mathsf{PKE.SK}(1^\lambda, fp)$, and then applying to $(1^\lambda, fp, sk)$ a PT deterministic *public-key derivation function* we denote $\mathsf{PKE.PK}$ 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 [36], the Kurosawa-Desmedt scheme [52] and for schemes obtained via the BCHK transform [25] applied to the identity-based encryption schemes of Boneh-Boyen [23] or Waters [60]. 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 inputs $1^\lambda, fp$, picks random coins ω, lets $(sk, pk) \leftarrow \mathsf{PKE.Kg}(1^\lambda, fp; \omega)$, 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 $1^\lambda, fp, \omega, C$, lets $(sk, pk) \leftarrow_{\$} \mathsf{PKE.Kg}(1^\lambda, fp; \omega)$ and outputs $M \leftarrow \mathsf{PKE.Dec}(1^\lambda, fp, sk, C)$. 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 canonical PKE as above, we construct a JES scheme JES. 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 $\mathsf{F.Dom} = \mathsf{PKE.SK}$, so the domain of F is the space of secret keys of PKE; and let $\mathsf{F.Ev} = \mathsf{PKE.PK}$, so that the function defined by fp 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 3.2.) Now we define our JES scheme JES. Let $\mathsf{JES.Pg} = \mathsf{DS.Pg}$, so that parameters for JES have the form $jp = (fp, ap)$, where fp are parameters for F, which by definition of F are also parameters for PKE. Let $\mathsf{JES.Kg} = \mathsf{DS.Kg}$. (Keys are those of PKE which are also those of DS.) Let $\mathsf{JES.Sig} = \mathsf{DS.Sig}$ and $\mathsf{JES.Ver} = \mathsf{DS.Ver}$, meaning the signing and verifying algorithms of the joint scheme JES are inherited from the signature scheme DS. Let $\mathsf{JES.Enc}(1^\lambda, (fp, ap), pk, M)$ return $\mathsf{PKE.Enc}(1^\lambda, fp, pk, M)$ and let $\mathsf{JES.Dec}(1^\lambda, (fp, ap), sk, C)$ return $\mathsf{PKE.Dec}(1^\lambda, fp, sk, C)$, so that the encryption and decryption algorithms of the joint scheme JES are inherited from the PKE scheme PKE. Note that the public key of the joint scheme JES 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 would 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 JES is both IND and SUF secure.

Theorem 4.1 *Let* PKE *be a canonical public-key encryption scheme. Let* F *be defined from it as above. Let* DS *be an* F*-keyed signature scheme, and let* JES *be the corresponding joint encryption and signature scheme constructed above. Assume* PKE *is IND-CCA secure. Assume* DS *is simulatable and key-extractable. Then (1)* JES *is IND secure, and (2)* JES *is SUF secure.*

A full proof of this theorem is in Appendix C. Here we sketch the main ideas. For (1), given an adversary A against the IND security of JES, 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. A simulation adversary is built alongside, but key extraction is not needed. For (2), given an adversary A against the SUF security of JES, 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.

5 RKA-secure signatures from RKA-secure OWFs

RKA security is notoriously hard to provably achieve. Recognizing this, several authors [43, 9] 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 [9]. 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.

```
MAIN RKAOWF$^A_{\mathsf{F},\Phi}(\lambda)$
$fp \leftarrow\$ \mathsf{F.Pg}(1^\lambda)$
$x \leftarrow\$ \mathsf{F.Dom}(1^\lambda, fp)$ ; $y \leftarrow \mathsf{F.Ev}(1^\lambda, fp, x)$
$x' \leftarrow\$ A^{\text{EVAL}}(1^\lambda, fp, y)$
Ret $(\mathsf{F.Ev}(1^\lambda, fp, x') = y)$

EVAL($\phi$)
$x' \leftarrow \Phi(1^\lambda, fp, \phi, x)$ ; $y' \leftarrow \mathsf{F.Ev}(1^\lambda, fp, x')$
Ret $y'$
```

```
MAIN RKASIG$^A_{\mathsf{DS},\mathsf{F},\Phi}(\lambda)$
$Q \leftarrow \emptyset$ ; $(fp, ap) \leftarrow\$ \mathsf{DS.Pg}(1^\lambda)$ ; $pp \leftarrow (fp, ap)$
$(sk, pk) \leftarrow\$ \mathsf{DS.Kg}(1^\lambda, pp)$ ; $(M, \sigma) \leftarrow\$ A^{\text{SIGN}}(1^\lambda, pp, pk)$
Ret $(\mathsf{DS.Ver}(1^\lambda, pp, pk, M, \sigma)$ and $(pk, M, \sigma) \notin Q)$

SIGN($\phi, M$)
$sk' \leftarrow \Phi(1^\lambda, fp, \phi, sk)$ ; $pk' \leftarrow \mathsf{F.Ev}(1^\lambda, fp, sk')$
$\sigma \leftarrow\$ \mathsf{DS.Sig}(1^\lambda, pp, sk', M)$ ; $Q \leftarrow Q \cup \{(pk', M, \sigma)\}$
Ret $\sigma'$
```

Figure 3: **Games defining Φ-RKA security of a function family F (left) and an F-keyed signature scheme DS (right).**

RKA SECURITY. Let F be a function family. A class of RKD (related-key deriving) functions Φ for F is a PT-computable function that specifies for each $\lambda \in \mathbb{N}$, each $fp \in [\mathsf{F.Pg}(1^\lambda)]$ and each $\phi \in \{0,1\}^*$ a map $\Phi(1^\lambda, fp, \phi, \cdot) : \mathsf{F.Dom}(1^\lambda, fp) \to \mathsf{F.Dom}(1^\lambda, fp)$ called the RKD function described by ϕ. We say that F is Φ-RKA secure if $\mathbf{Adv}^{\mathrm{rka}}_{\mathsf{F},A,\Phi}(\cdot)$ is negligible for every PT adversary A, where $\mathbf{Adv}^{\mathrm{rka}}_{\mathsf{F},A,\Phi}(\lambda) = \Pr[\mathrm{RKAOWF}^A_{\mathsf{F},\Phi}(\lambda)]$ and game RKAOWF is on the left-hand side of Figure 3. In this game, A, like in the basic one-wayness notion, is given $y = \mathsf{F.Ev}(1^\lambda, fp, x)$ and is attempting to find x' such that $\mathsf{F.Ev}(1^\lambda, fp, x') = y$. Now, however, it has help. It can request that the hidden challenge input x be modified to $x' = \Phi(1^\lambda, fp, \phi, x)$ for any description ϕ of its choice, and obtain $y' = \mathsf{F.Ev}(1^\lambda, fp, x')$. This should not help it in its inversion task. The definition is from Goldenberg and Liskov [43], 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 $\mathbf{Adv}^{\mathrm{rka}}_{\mathsf{DS},A,\Phi}(\cdot)$ is negligible for every PT adversary A, where $\mathbf{Adv}^{\mathrm{rka}}_{\mathsf{DS},A,\Phi}(\lambda) = \Pr[\mathrm{RKASIG}^A_{\mathsf{DS},\Phi}(\lambda)]$ and game RKASIG is on the right-hand side of Figure 3. 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 $sk' = \Phi(1^\lambda, fp, \phi, sk)$ for any description 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 [9] for Φ-RKA security of arbitrary signature schemes to the special case of F-keyed signature schemes.[5]

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 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 5.1 *Let DS be an F-keyed signature scheme that is simulatable and key-extractable. Let Φ be a class of RKD functions. If F is Φ-RKA secure then DS is also Φ-RKA secure.*

A formal proof of this theorem is in Appendix D. The proof extends that of Theorem 3.1. The adversary I that we build uses its EVAL oracle to simulate the SIGN oracle of the given adversary A. In applying

[5] One change (strengthening the definition) is that we use a strong unforgeability formulation rather than an unforgeability one. On the other hand while [9] 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 5.1 extends to the definition of [9] assuming F is additionally injective or collision-resistant, which is true in most examples.

F	fp	x	$\mathsf{F.Ev}(1^\lambda, fp, x)$	ϕ	$\Phi(1^\lambda, fp, \phi, x)$	$M(1^\lambda, fp, \phi, y)$
EXP	$(\langle\mathbb{G}\rangle, g, m)$	$\in \mathbb{Z}_m$	g^x	$(a,b) \in \mathbb{Z}_m \times \mathbb{Z}_m$	$(ax+b) \bmod m$	$y^a g^b$
RSA	(N, e)	$\in \mathbb{Z}_N^*$	$x^e \bmod N$	$a \in \mathbb{N}$	$x^a \bmod N$	$y^a \bmod N$
LWE	(A, n, m, q)	$(s,e) \in \mathbb{Z}_q^m \times \mathbb{Z}_q^n$	$(As+e) \bmod q$	$(s', e') \in \mathbb{Z}_q^m \times \mathbb{Z}_q^n$	$(s+s', e+e') \bmod q$	$(y+As'+e') \bmod q$

Table 1: **Φ-RKA secure OWFs:** We succinctly define the families and the Φ-key-simulator showing their Φ malleability and hence their Φ-RKA security.

the simulation and key-extractability, we make crucial use of the fact that the adversaries can obtain signatures under secret keys of their choice.

FINDING Φ-RKA OWFS. Theorem 5.1 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 [8] 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 $M(1^\lambda, fp, \phi, \mathsf{F.Ev}(1^\lambda, fp, x)) = \mathsf{F.Ev}(1^\lambda, fp, \Phi(1^\lambda, fp, \phi, x))$ for all $\lambda \in \mathbb{N}$, all $fp \in [\mathsf{F.Pg}(1^\lambda)]$, all $\phi \in \{0,1\}^*$ and all $x \in \mathsf{F.Dom}(1^\lambda, fp)$. Then we have:

Proposition 5.2 *Let F be a function family and Φ a class of RKD functions. If F is Φ-key-malleable and one-way 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 B against the (regular) one-wayness of F such that $\mathbf{Adv}^{\mathrm{rka}}_{\mathsf{F},A}(\lambda) \leq \mathbf{Adv}^{\mathrm{ow}}_{\mathsf{F},B}(\lambda)$ for all $\lambda \in \mathbb{N}$. On input $(1^\lambda, fp, y)$, adversary B runs $A(1^\lambda, fp, y)$. When A makes a EVAL query ϕ, adversary B computes $y' \leftarrow M(1^\lambda, fp, \phi, y)$ and returns y' to A. Φ-key malleability says that $y' = \mathsf{F.Ev}(1^\lambda, fp, \Phi(1^\lambda, fp, \phi, x))$ as A expects. When A eventually halts and outputs a value x', adversary B does the same. ▌

Previous uses of key-malleability [8, 15] 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 5.2 to find Φ-RKA OWFs and thence, via Theorem 5.1, Φ-RKA signature schemes. Table 1 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 row of Table 1 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 row of Table 1 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 row of Table 1 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 to Φ-RKA security over standard and natural keyspaces.

```
main $\mathrm{KDM}^{A}_{\mathsf{PKE},\Phi}(\lambda)$
$b \leftarrow_{\$} \{0,1\}$ ; $Q \leftarrow \emptyset$ ; $pp_e \leftarrow_{\$} \mathsf{PKE.Pg}(1^\lambda)$ ; $b' \leftarrow_{\$} A^{\textsc{MkKey},\textsc{Enc},\textsc{Dec}}(1^\lambda, pp_e)$ ; Ret $(b = b')$

$\textsc{MkKey}(1^n)$
For $i = 1, \ldots, n$ do $(\mathbf{sk}[i], \mathbf{pk}[i]) \leftarrow_{\$} \mathsf{PKE.Kg}(1^\lambda, pp)$
Ret $\mathbf{pk}$

$\textsc{Enc}(\phi, i)$
If not $(1 \le i \le n)$ then Ret $\perp$
$M \leftarrow \Phi(1^\lambda, \phi, \mathbf{sk})$
If $(b = 1)$ then $C \leftarrow_{\$} \mathsf{PKE.Enc}(1^\lambda, pp_e, \mathbf{pk}[i], M)$ else $C \leftarrow_{\$} \mathsf{PKE.Enc}(1^\lambda, pp_e, \mathbf{pk}[i], 0^{|M|})$
$Q \leftarrow Q \cup \{(C, i)\}$ ; Ret $C$
```

Figure 4: **Game defining Φ-KDM security of a public-key encryption scheme PKE.**

6 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., [21, 27, 5, 4, 31, 32, 20, 7, 53, 3, 29, 30, 12, 41, 49]) 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 formalizing 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 utilized 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 PT-computable function specifying for each $\lambda \in \mathbb{N}$ and each $\phi \in \{0,1\}^*$ a map $\Phi(1^\lambda, \phi, \cdot)$ called the message-deriving function described by ϕ. This map takes input a vector $\mathbf{sk}$ (of keys) and returns a string (the message) of length $\phi.m$, where $\phi.m \in \mathbb{N}$, the output length of ϕ, is computable in polynomial time given $1^\lambda, \phi$. We assume Φ always includes all constant functions. Formally, $\langle M \rangle$ describes the function defined by $\Phi(1^\lambda, \langle M \rangle, \mathbf{sk}) = M$ for all $M \in \{0,1\}^*$. We say that public-key encryption scheme PKE is Φ-KDM secure if $\mathbf{Adv}^{\mathrm{kdm}}_{\mathsf{PKE},A,\Phi}(\cdot)$ is negligible for every PT adversary A, where $\mathbf{Adv}^{\mathrm{kdm}}_{\mathsf{PKE},A,\Phi}(\lambda) = 2\Pr[\mathrm{KDM}^{A}_{\mathsf{PKE},\Phi}(\lambda)] - 1$ and game KDM is in Figure 4. We require that A make exactly one query to MkKey and that this be its first oracle query. The argument $n \ge 1$ determines the number of keys and must be given in unary.

The definition follows [21] except in parameterizing security by the class Φ of allowed message-deriving functions. The parameterization is important because many existing KDM-secure encryption schemes are for particular classes Φ, for example for encryption cycles, affine functions or cliques [27, 4, 29, 30, 49]. 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 $(sk, pk) \leftarrow\$ \mathsf{ST.Kg}(1^\lambda, pp)$ a user can generate a secret key sk and corresponding public key pk; via $D \leftarrow\$ \mathsf{ST.Store}(1^\lambda, pp, sk, M)$ a user can produce some data D based on $M \in \{0,1\}^*$ to store on the server; via $M \leftarrow \mathsf{ST.Retrieve}(1^\lambda, pp, sk, D)$ a user can deterministically retrieve $M \in \{0,1\}^* \cup \{\bot\}$ from their stored data D; and via $d \leftarrow \mathsf{ST.Verify}(1^\lambda, pp, pk, D)$ the server can deterministically produce a decision $d \in \{\mathsf{true}, \mathsf{false}\}$ regarding the validity of D. Correctness requires that $\mathsf{ST.Retrieve}(1^\lambda, pp, sk, \mathsf{ST.Store}(1^\lambda, pp, sk, M)) = M$ and $\mathsf{ST.Verify}(1^\lambda, pp, pk, \mathsf{ST.Store}(1^\lambda, pp, sk, M)) = \mathsf{true}$ for all $\lambda \in \mathbb{N}$, all $pp \in [\mathsf{ST.Pg}(1^\lambda)]$, all $(sk, pk) \in [\mathsf{ST.Kg}(1^\lambda, pp)]$, and all messages $M \in \{0,1\}^*$. Let Φ be a class of KDM functions as above. We say that ST is Φ-IND-secure if $\mathbf{Adv}^{\mathrm{ind}}_{\mathsf{ST},A,\Phi}(\cdot)$ is negligible for all PT adversaries A, where $\mathbf{Adv}^{\mathrm{ind}}_{\mathsf{ST},A,\Phi}(\lambda) = 2\Pr[\mathrm{IND}^A_{\mathsf{ST},\Phi}(\lambda)] - 1$ and game IND is on the left-hand side of Figure 5. The presence of the Retrieve oracle makes this a CCA KDM notion. We say that ST is Φ-SUF-secure if $\mathbf{Adv}^{\mathrm{suf}}_{\mathsf{ST},A,\Phi}(\cdot)$ is negligible for all PT adversaries A, where $\mathbf{Adv}^{\mathrm{suf}}_{\mathsf{ST},A,\Phi}(\lambda) = \Pr[\mathrm{SUF}^A_{\mathsf{ST},\Phi}(\lambda)]$ and game SUF is on the right-hand side of Figure 5. In both cases, we require that A make exactly one query to MkKey and that this be its first oracle query, and again the argument $n \geq 1$, indicating the number of keys, must 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 4, we begin by constructing from PKE a function family F. We do not repeat this construction here, but refer the reader to Section 4. We then let DS be an F-keyed signature scheme that is simulatable and key-extractable. We construct our storage scheme ST as follows:

- $\underline{\mathsf{ST.Pg}(\lambda)}$: Ret $(fp, ap) \leftarrow\$ \mathsf{DS.Pg}(1^\lambda)$. Thus, parameters for ST have the form $pp = (fp, ap)$, where fp are parameters for both F and PKE.
- $\underline{\mathsf{ST.Kg}(1^\lambda, (fp, ap))}$: Ret $(sk, pk) \leftarrow\$ \mathsf{DS.Kg}(1^\lambda, (fp, ap))$. Thus, keys are those of PKE, which are also those of DS.
- $\underline{\mathsf{ST.Store}(1^\lambda, (fp, ap), sk, M)}$: $pk \leftarrow \mathsf{F.Ev}(1^\lambda, fp, sk)$; $C \leftarrow\$ \mathsf{PKE.Enc}(1^\lambda, fp, pk, M)$; $\sigma \leftarrow\$ \mathsf{DS.Sig}(1^\lambda, pp, sk, C)$; Ret (C, σ).
- $\underline{\mathsf{ST.Retrieve}(1^\lambda, (fp, ap), sk, (C, \sigma))}$: $pk \leftarrow \mathsf{F.Ev}(1^\lambda, fp, sk)$. If $\mathsf{DS.Ver}(1^\lambda, pp, pk, C, \sigma) = \mathsf{false}$ then Ret $\bot$. Else Ret $\mathsf{PKE.Dec}(1^\lambda, fp, sk, C)$.
- $\underline{\mathsf{ST.Verify}(1^\lambda, (fp, ap), pk, (C, \sigma))}$: Ret $\mathsf{DS.Ver}(1^\lambda, (fp, ap), pk, C, \sigma)$.

The following says that our construction provides both privacy and integrity for key-dependent messages, assuming privacy for key-dependent messages of the base encryption scheme and simulatability and key-extractability of the F-keyed signature scheme:

Theorem 6.1 *Let* PKE *be a canonical public-key encryption scheme, and let* F *be defined from it as above. Let* DS *be an* F*-keyed signature scheme, and let* ST *be the corresponding storage scheme constructed above. Let* Φ *be a class of message-deriving functions. Assume* PKE *is* Φ*-KDM secure. Assume* DS *is simulatable and key-extractable. Then (1)* ST *is* Φ*-IND secure and (2)* ST *is* Φ*-SUF secure.*

A full proof of Theorem 6.1 is in Appendix E. Here 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

```
main IND$^A_{\mathsf{ST},\Phi}(\lambda)$
$b \leftarrow\!\!\$ \{0,1\}$ ; $Q \leftarrow \emptyset$ ; $pp \leftarrow\!\!\$ \mathsf{ST.Pg}(1^\lambda)$
$b' \leftarrow\!\!\$ A^{\text{MkKey,Store,Retrieve}}(1^\lambda, pp)$
Ret $(b = b')$

MkKey$(1^n)$
For $i = 1, \ldots, n$ do $(\mathbf{sk}[i], \mathbf{pk}[i]) \leftarrow\!\!\$ \mathsf{ST.Kg}(1^\lambda, pp)$
Ret $\mathbf{pk}$

Store$(\phi, i)$
If not $(1 \le i \le n)$ then Ret $\perp$
If $(b = 1)$ then $M \leftarrow \Phi(1^\lambda, \phi, \mathbf{sk})$
Else $M \leftarrow 0^{\phi.m}$
$D \leftarrow\!\!\$ \mathsf{ST.Store}(1^\lambda, pp, \mathbf{sk}[i], M)$
$Q \leftarrow Q \cup \{(D, i)\}$
Ret $D$

Retrieve$(D, i)$
If not $(1 \le i \le n)$ then Ret $\perp$
If $((D, i) \in Q)$ then Ret $\perp$
$M \leftarrow \mathsf{ST.Retrieve}(1^\lambda, pp, \mathbf{sk}[i], D)$
Ret $M$
```

```
main SUF$^A_{\mathsf{ST},\Phi}(\lambda)$
$Q \leftarrow \emptyset$ ; $pp \leftarrow\!\!\$ \mathsf{ST.Pg}(1^\lambda)$
$(D, i) \leftarrow\!\!\$ A^{\text{MkKey,Store,Retrieve}}(1^\lambda, pp)$
If $(D, i) \in Q$ then Ret false
If not $(1 \le i \le n)$ then Ret false
Ret $\mathsf{ST.Verify}(1^\lambda, pp, \mathbf{pk}[i], D)$

MkKey$(1^n)$
For $i = 1, \ldots, n$ do $(\mathbf{sk}[i], \mathbf{pk}[i]) \leftarrow\!\!\$ \mathsf{ST.Kg}(1^\lambda, pp)$
Ret $\mathbf{pk}$

Store$(\phi, i)$
If not $(1 \le i \le n)$ then Ret $\perp$
$M \leftarrow \Phi(1^\lambda, \phi, \mathbf{sk})$ ; $D \leftarrow\!\!\$ \mathsf{ST.Store}(1^\lambda, pp, \mathbf{sk}[i], M)$
$Q \leftarrow Q \cup \{(D, i)\}$
Ret $D$

Retrieve$(D, i)$
If not $(1 \le i \le n)$ then Ret $\perp$
If $((D, i) \in Q)$ then Ret $\perp$
$M \leftarrow \mathsf{ST.Retrieve}(1^\lambda, pp, \mathbf{sk}[i], D)$
Ret $M$
```

Figure 5: **Games defining KDM-security of storage scheme ST:** Privacy (left) and unforgeability (right).

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.

Instantiation. We require our base scheme PKE to be canonical. In Section 4 we showed how to modify an encryption scheme to be canonical while preserving IND-CCA, but the transformation does not in general preserve KDM-security. Instead, we would use KDM-secure schemes that are already canonical. One possibility is the scheme of [53]. The schemes of [27, 7, 4] are not canonical.

References

[1] M. Abe, G. Fuchsbauer, J. Groth, K. Haralambiev, and M. Ohkubo. Structure-preserving signatures and commitments to group elements. In T. Rabin, editor, *CRYPTO 2010*, volume 6223 of *LNCS*, pages 209–236. Springer, Aug. 2010. (Cited on page 5, 8.)

[2] J. H. An, Y. Dodis, and T. Rabin. On the security of joint signature and encryption. In L. R. Knudsen, editor, *EUROCRYPT 2002*, volume 2332 of *LNCS*, pages 83–107. Springer, Apr. / May 2002. (Cited on page 5.)

[3] B. Applebaum. Key-dependent message security: Generic amplification and completeness. In K. G. Paterson, editor, *EUROCRYPT 2011*, volume 6632 of *LNCS*, pages 527–546. Springer, May 2011. (Cited on page 4, 14.)

[4] B. Applebaum, D. Cash, C. Peikert, and A. Sahai. Fast cryptographic primitives and circular-secure encryption based on hard learning problems. In S. Halevi, editor, *CRYPTO 2009*, volume 5677 of *LNCS*, pages 595–618. Springer, Aug. 2009. (Cited on page 4, 14, 15, 16.)

[5] M. Backes, M. Dürmuth, and D. Unruh. OAEP is secure under key-dependent messages. In J. Pieprzyk, editor, *ASIACRYPT 2008*, volume 5350 of *LNCS*, pages 506–523. Springer, Dec. 2008. (Cited on page 4, 14.)

[6] M. Backes, B. Pfitzmann, and A. Scedrov. Key-dependent message security under active attacks - brsim/uc-soundness of dolev-yao-style encryption with key cycles. *Journal of Computer Security*, 16(5):497–530, 2008. (Cited on page 5.)

[7] B. Barak, I. Haitner, D. Hofheinz, and Y. Ishai. Bounded key-dependent message security. In H. Gilbert, editor, *EUROCRYPT 2010*, volume 6110 of *LNCS*, pages 423–444. Springer, May 2010. (Cited on page 4, 14, 16.)

[8] M. Bellare and D. Cash. Pseudorandom functions and permutations provably secure against related-key attacks. In T. Rabin, editor, *CRYPTO 2010*, volume 6223 of *LNCS*, pages 666–684. Springer, Aug. 2010. (Cited on page 4, 13.)

[9] M. Bellare, D. Cash, and R. Miller. Cryptography secure against related-key attacks and tampering. In D. H. Lee and X. Wang, editors, *ASIACRYPT 2011*, volume 7073 of *LNCS*, pages 486–503. Springer, Dec. 2011. (Cited on page 4, 11, 12.)

[10] M. Bellare, A. Desai, D. Pointcheval, and P. Rogaway. Relations among notions of security for public-key encryption schemes. In H. Krawczyk, editor, *CRYPTO'98*, volume 1462 of *LNCS*, pages 26–45. Springer, Aug. 1998. (Cited on page 23.)

[11] M. Bellare and S. Goldwasser. New paradigms for digital signatures and message authentication based on non-interative zero knowledge proofs. In G. Brassard, editor, *CRYPTO'89*, volume 435 of *LNCS*, pages 194–211. Springer, Aug. 1990. (Cited on page 5, 9.)

[12] M. Bellare and S. Keelveedhi. Authenticated and misuse-resistant encryption of key-dependent data. In P. Rogaway, editor, *CRYPTO 2011*, volume 6841 of *LNCS*, pages 610–629. Springer, Aug. 2011. (Cited on page 4, 5, 14.)

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

[14] M. Bellare and C. Namprempre. Authenticated encryption: Relations among notions and analysis of the generic composition paradigm. *Journal of Cryptology*, 21(4):469–491, Oct. 2008. (Cited on page 5.)

[15] M. Bellare, K. G. Paterson, and S. Thomson. RKA security beyond the linear barrier: IBE, encryption and signatures. In *ASIACRYPT*, pages 331–348, 2012. (Cited on page 4, 13.)

[16] M. Bellare and P. Rogaway. Random oracles are practical: A paradigm for designing efficient protocols. In V. Ashby, editor, *ACM CCS 93*, pages 62–73. ACM Press, Nov. 1993. (Cited on page 3.)

[17] M. Bellare and P. Rogaway. The security of triple encryption and a framework for code-based game-playing proofs. In S. Vaudenay, editor, *EUROCRYPT 2006*, volume 4004 of *LNCS*, pages 409–426. Springer, May / June 2006. (Cited on page 20, 26, 27, 29, 32.)

[18] E. Biham. New types of cryptoanalytic attacks using related keys (extended abstract). In T. Helleseth, editor, *EUROCRYPT'93*, volume 765 of *LNCS*, pages 398–409. Springer, May 1993. (Cited on page 4.)

[19] E. Biham and A. Shamir. Differential fault analysis of secret key cryptosystems. In B. S. Kaliski Jr., editor, *CRYPTO'97*, volume 1294 of *LNCS*, pages 513–525. Springer, Aug. 1997. (Cited on page 4.)

[20] N. Bitansky and R. Canetti. On strong simulation and composable point obfuscation. In T. Rabin, editor, *CRYPTO 2010*, volume 6223 of *LNCS*, pages 520–537. Springer, Aug. 2010. (Cited on page 4, 14.)

[21] J. Black, P. Rogaway, and T. Shrimpton. Encryption-scheme security in the presence of key-dependent messages. In K. Nyberg and H. M. Heys, editors, *SAC 2002*, volume 2595 of *LNCS*, pages 62–75. Springer, Aug. 2003. (Cited on page 4, 14.)

[22] M. Blum, A. De Santis, S. Micali, and G. Persiano. Noninteractive zero-knowledge. *SIAM Journal on Computing*, 20(6):1084–1118, 1991. (Cited on page 22.)

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

[24] D. Boneh and X. Boyen. Short signatures without random oracles. In C. Cachin and J. Camenisch, editors, *EUROCRYPT 2004*, volume 3027 of *LNCS*, pages 56–73. Springer, May 2004. (Cited on page 3.)

[25] D. Boneh, R. Canetti, S. Halevi, and J. Katz. Chosen-ciphertext security from identity-based encryption. *SIAM Journal on Computing*, 36(5):1301–1328, 2007. (Cited on page 10.)

[26] D. Boneh, R. A. DeMillo, and R. J. Lipton. On the importance of checking cryptographic protocols for faults (extended abstract). In W. Fumy, editor, *EUROCRYPT'97*, volume 1233 of *LNCS*, pages 37–51. Springer, May 1997. (Cited on page 4.)

[27] D. Boneh, S. Halevi, M. Hamburg, and R. Ostrovsky. Circular-secure encryption from decision diffie-hellman. In D. Wagner, editor, *CRYPTO 2008*, volume 5157 of *LNCS*, pages 108–125. Springer, Aug. 2008. (Cited on page 4, 14, 15, 16.)

[28] X. Boyen, Q. Mei, and B. Waters. Direct chosen ciphertext security from identity-based techniques. In V. Atluri, C. Meadows, and A. Juels, editors, *ACM CCS 05*, pages 320–329. ACM Press, Nov. 2005. (Cited on page 3.)

[29] Z. Brakerski, S. Goldwasser, and Y. T. Kalai. Black-box circular-secure encryption beyond affine functions. In Y. Ishai, editor, *TCC 2011*, volume 6597 of *LNCS*, pages 201–218. Springer, Mar. 2011. (Cited on page 4, 14, 15.)

[30] Z. Brakerski and V. Vaikuntanathan. Fully homomorphic encryption from ring-LWE and security for key dependent messages. In P. Rogaway, editor, *CRYPTO 2011*, volume 6841 of *LNCS*, pages 505–524. Springer, Aug. 2011. (Cited on page 4, 14, 15.)

[31] J. Camenisch, N. Chandran, and V. Shoup. A public key encryption scheme secure against key dependent chosen plaintext and adaptive chosen ciphertext attacks. In A. Joux, editor, *EUROCRYPT 2009*, volume 5479 of *LNCS*, pages 351–368. Springer, Apr. 2009. (Cited on page 4, 14.)

[32] R. Canetti, Y. T. Kalai, M. Varia, and D. Wichs. On symmetric encryption and point obfuscation. In D. Micciancio, editor, *TCC 2010*, volume 5978 of *LNCS*, pages 52–71. Springer, Feb. 2010. (Cited on page 4, 14.)

[33] M. Chase, M. Kohlweiss, A. Lysyanskaya, and S. Meiklejohn. Malleable signatures: Complex unary transformations and delegatable anonymous credentials. Cryptology ePrint Archive, Report 2013/179, 2013. `http://eprint.iacr.org/`. (Cited on page 5, 8, 9.)

[34] J.-S. Coron, M. Joye, D. Naccache, and P. Paillier. Universal padding schemes for RSA. In M. Yung, editor, *CRYPTO 2002*, volume 2442 of *LNCS*, pages 226–241. Springer, Aug. 2002. (Cited on page 3.)

[35] R. Cramer and V. Shoup. A practical public key cryptosystem provably secure against adaptive chosen ciphertext attack. In H. Krawczyk, editor, *CRYPTO'98*, volume 1462 of *LNCS*, pages 13–25. Springer, Aug. 1998. (Cited on page 23.)

[36] R. Cramer and V. Shoup. Design and analysis of practical public-key encryption schemes secure against adaptive chosen ciphertext attack. *SIAM Journal on Computing*, 33(1):167–226, 2003. (Cited on page 4, 9, 10.)

[37] A. De Santis, G. Di Crescenzo, R. Ostrovsky, G. Persiano, and A. Sahai. Robust non-interactive zero knowledge. In J. Kilian, editor, *CRYPTO 2001*, volume 2139 of *LNCS*, pages 566–598. Springer, Aug. 2001. (Cited on page 5, 8, 9, 22.)

[38] Y. Dodis, K. Haralambiev, A. López-Alt, and D. Wichs. Efficient public-key cryptography in the presence of key leakage. In M. Abe, editor, *ASIACRYPT 2010*, volume 6477 of *LNCS*, pages 613–631. Springer, Dec. 2010. (Cited on page 5, 8, 9, 22.)

[39] D. Dolev, C. Dwork, and M. Naor. Nonmalleable cryptography. *SIAM Journal on Computing*, 30(2):391–437, 2000. (Cited on page 9.)

[40] J. Fan, Y. Zheng, and X. Tang. A single key pair is adequate for the zheng signcryption. In U. Parampalli and P. Hawkes, editors, *ACISP 11*, volume 6812 of *LNCS*, pages 371–388. Springer, July 2011. (Cited on page 5.)

[41] D. Galindo, J. Herranz, and J. L. Villar. Identity-based encryption with master key-dependent message security and leakage-resilience. In S. Foresti, M. Yung, and F. Martinelli, editors, *ESORICS 2012*, volume 7459 of *LNCS*, pages 627–642. Springer, Sept. 2012. (Cited on page 4, 14.)

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

[43] D. Goldenberg and M. Liskov. On related-secret pseudorandomness. In D. Micciancio, editor, *TCC 2010*, volume 5978 of *LNCS*, pages 255–272. Springer, Feb. 2010. (Cited on page 4, 11, 12.)

[44] S. Goldwasser, S. Micali, and R. L. Rivest. A digital signature scheme secure against adaptive chosen-message attacks. *SIAM Journal on Computing*, 17(2):281–308, Apr. 1988. (Cited on page 8.)

[45] J. Groth. Simulation-sound NIZK proofs for a practical language and constant size group signatures. In X. Lai and K. Chen, editors, *ASIACRYPT 2006*, volume 4284 of *LNCS*, pages 444–459. Springer, Dec. 2006. (Cited on page 5, 8, 9, 22.)

[46] J. Groth and R. Ostrovsky. Cryptography in the multi-string model. In A. Menezes, editor, *CRYPTO 2007*, volume 4622 of *LNCS*, pages 323–341. Springer, Aug. 2007. (Cited on page 5, 9, 22.)

[47] S. Haber and B. Pinkas. Securely combining public-key cryptosystems. In *ACM CCS 01*, pages 215–224. ACM Press, Nov. 2001. (Cited on page 3, 5, 9, 10.)

[48] K. Haralambiev. *Efficient Cryptographic Primitives for Non-Interactive Zero-Knowledge Proofs and Applications*. PhD thesis, New York University, May 2011. (Cited on page 5, 22.)

[49] D. Hofheinz. Circular chosen-ciphertext security with compact ciphertexts. In *EUROCRYPT*, volume 7881 of *Lecture Notes in Computer Science*, pages 520–536. Springer, 2013. To appear. (Cited on page 4, 14, 15.)

[50] L. R. Knudsen. Cryptanalysis of LOKI91. In J. Seberry and Y. Zheng, editors, *AUSCRYPT'92*, volume 718 of *LNCS*, pages 196–208. Springer, Dec. 1992. (Cited on page 4.)

[51] Y. Komano and K. Ohta. Efficient universal padding techniques for multiplicative trapdoor one-way permutation. In D. Boneh, editor, *CRYPTO 2003*, volume 2729 of *LNCS*, pages 366–382. Springer, Aug. 2003. (Cited on page 3.)

[52] K. Kurosawa and Y. Desmedt. A new paradigm of hybrid encryption scheme. In M. Franklin, editor, *CRYPTO 2004*, volume 3152 of *LNCS*, pages 426–442. Springer, Aug. 2004. (Cited on page 9, 10.)

[53] T. Malkin, I. Teranishi, and M. Yung. Efficient circuit-size independent public key encryption with KDM security. In K. G. Paterson, editor, *EUROCRYPT 2011*, volume 6632 of *LNCS*, pages 507–526. Springer, May 2011. (Cited on page 4, 14, 16.)

[54] M. G. Muñiz and R. Steinwandt. Security of signature schemes in the presence of key-dependent messages. *Tatra Mt. Math. Publ.*, 47:15–29, 2010. (Cited on page 5.)

```
$\underline{\text{MAIN SUF}^A_{\mathsf{DS}}(\lambda)}$
$Q \leftarrow \emptyset$ ; $pp \leftarrow_\$ \mathsf{DS.Pg}(1^\lambda)$ ; $(sk, pk) \leftarrow_\$ \mathsf{DS.Kg}(1^\lambda, pp)$
$(M, \sigma) \leftarrow_\$ A^{\text{SIGN}}(1^\lambda, pp, pk)$
Ret $(\mathsf{DS.Ver}(1^\lambda, pp, pk, M, \sigma)$ and $(M, \sigma) \notin Q)$

$\underline{\text{SIGN}(M)}$
$\sigma \leftarrow_\$ \mathsf{DS.Sig}(1^\lambda, pp, sk, M)$ ; $Q \leftarrow Q \cup \{(M, \sigma)\}$
Ret $\sigma$
```

Figure 6: **Game defining strong unforgeability of signature scheme DS.**

[55] K. G. Paterson, J. C. N. Schuldt, M. Stam, and S. Thomson. On the joint security of encryption and signature, revisited. In D. H. Lee and X. Wang, editors, *ASIACRYPT 2011*, volume 7073 of *LNCS*, pages 161–178. Springer, Dec. 2011. (Cited on page 3, 5, 9, 10.)

[56] P. Rogaway. Authenticated-encryption with associated-data. In V. Atluri, editor, *ACM CCS 02*, pages 98–107. ACM Press, Nov. 2002. (Cited on page 5.)

[57] J. Rompel. One-way functions are necessary and sufficient for secure signatures. In *22nd ACM STOC*, pages 387–394. ACM Press, May 1990. (Cited on page 6.)

[58] A. Sahai. Non-malleable non-interactive zero knowledge and adaptive chosen-ciphertext security. In *40th FOCS*, pages 543–553. IEEE Computer Society Press, Oct. 1999. (Cited on page 8.)

[59] C.-P. Schnorr. Efficient signature generation by smart cards. *Journal of Cryptology*, 4(3):161–174, 1991. (Cited on page 3.)

[60] B. R. Waters. Efficient identity-based encryption without random oracles. In R. Cramer, editor, *EUROCRYPT 2005*, volume 3494 of *LNCS*, pages 114–127. Springer, May 2005. (Cited on page 10.)

[61] Y. Zheng. Digital signcryption or how to achieve cost(signature & encryption) ¡¡ cost(signature) + cost(encryption). In B. S. Kaliski Jr., editor, *CRYPTO'97*, volume 1294 of *LNCS*, pages 165–179. Springer, Aug. 1997. (Cited on page 4, 5.)

A Sim+Ext implies SUF: Proof of Theorem 3.1

We say a signature scheme DS is *strongly unforgeable* if $\mathbf{Adv}^{\text{suf}}_{\mathsf{DS},A}(\cdot)$ is negligible for all PT adversaries A, where $\mathbf{Adv}^{\text{suf}}_{\mathsf{DS},A}(\lambda) = \Pr[\text{SUF}^A_{\mathsf{DS}}(\lambda)]$ and game SUF is in Figure 6. We proceed to prove Theorem 3.1.

Proof: Let A be a PT adversary playing game SUF. We build PT adversaries I, A_1, A_2 such that

$$\mathbf{Adv}^{\text{suf}}_{\mathsf{DS},A}(\lambda) \le \mathbf{Adv}^{\text{ow}}_{\mathsf{F},I}(\lambda) + \mathbf{Adv}^{\text{sim}}_{\mathsf{DS},A_1}(\lambda) + \mathbf{Adv}^{\text{ext}}_{\mathsf{DS},A_2}(\lambda)$$

for all $\lambda \in \mathbb{N}$, from which the theorem follows. The proof uses the games in Figure 7. These games switch to using simulated parameters and signatures. We will build I, A_1, A_2 so that for all $\lambda \in \mathbb{N}$ we have

$$\Pr[\text{SUF}^A_{\mathsf{DS}}(\lambda)] - \Pr[\text{G}^A_0(\lambda)] \le \mathbf{Adv}^{\text{sim}}_{\mathsf{DS},A_1}(\lambda) \tag{1}$$

$$\Pr[\text{G}^A_0(\lambda) \text{ sets } \mathsf{bad}] \le \mathbf{Adv}^{\text{ext}}_{\mathsf{DS},A_2}(\lambda) \tag{2}$$

$$\Pr[\text{G}^A_1(\lambda)] \le \mathbf{Adv}^{\text{ow}}_{\mathsf{F},I}(\lambda) \; . \tag{3}$$

Games G_0 and G_1 are identical until bad, so by the Fundamental Lemma of Game-Playing [17] and the

```
main $\mathrm{G}_0^A(\lambda)$ / $\boxed{\mathrm{G}_1^A(\lambda)}$
$Q \leftarrow \emptyset$ ; $d \leftarrow \mathsf{false}$
$fp \leftarrow\$ \mathsf{F.Pg}(1^\lambda)$ ; $(ap, std, xtd) \leftarrow\$ \mathsf{DS.SimPg}(1^\lambda)$ ; $pp \leftarrow (fp, ap)$ ; $(sk, pk) \leftarrow\$ \mathsf{DS.Kg}(1^\lambda, pp)$
$(M, \sigma) \leftarrow\$ A^{\textsc{Sign}}(1^\lambda, pp, pk)$ ; $sk' \leftarrow\$ \mathsf{DS.Ext}(1^\lambda, pp, xtd, pk, M, \sigma)$
If $(\mathsf{DS.Ver}(1^\lambda, pp, pk, M, \sigma)$ and $(M, \sigma) \notin Q)$ then
    $d \leftarrow \mathsf{true}$
    If $(\mathsf{F.Ev}(1^\lambda, fp, sk') \neq pk)$ then $\mathsf{bad} \leftarrow \mathsf{true}$ ; $\boxed{d \leftarrow \mathsf{false}}$
Ret $d$

$\textsc{Sign}(M)$
$\sigma \leftarrow\$ \mathsf{DS.SimSig}(1^\lambda, pp, std, pk, M)$
$Q \leftarrow Q \cup \{(M, \sigma)\}$
Ret $\sigma$
```

Figure 7: **Games for the proof of Theorem 3.1:** Game G_1 includes the boxed code while G_0 does not.

above, for all $\lambda \in \mathbb{N}$ we have:

$$\begin{aligned}
\mathbf{Adv}^{\mathrm{suf}}_{\mathsf{DS},A}(\lambda) &= \Pr[\mathrm{SUF}^A_{\mathsf{DS}}(\lambda)] \\
&= (\Pr[\mathrm{SUF}^A_{\mathsf{DS}}(\lambda)] - \Pr[\mathrm{G}^A_0(\lambda)]) + (\Pr[\mathrm{G}^A_0(\lambda)] - \Pr[\mathrm{G}^A_1(\lambda)]) + \Pr[\mathrm{G}^A_1(\lambda)] \\
&\leq (\Pr[\mathrm{SUF}^A_{\mathsf{DS}}(\lambda)] - \Pr[\mathrm{G}^A_0(\lambda)]) + \Pr[\mathrm{G}^A_0(\lambda) \text{ sets } \mathsf{bad}] + \Pr[\mathrm{G}^A_1(\lambda)] \\
&\leq \mathbf{Adv}^{\mathrm{sim}}_{\mathsf{DS},A_1}(\lambda) + \mathbf{Adv}^{\mathrm{ext}}_{\mathsf{DS},A_2}(\lambda) + \mathbf{Adv}^{\mathrm{ow}}_{\mathsf{F},I}(\lambda)
\end{aligned}$$

as desired. We proceed to the constructions of A_1, A_2, I. Adversary A_1 behaves as follows:

```
$A_1^{\textsc{Sign}}(1^\lambda, pp)$
$(sk, pk) \leftarrow\$ \mathsf{DS.Kg}(1^\lambda, pp)$ ; $Q \leftarrow \emptyset$
$(M, \sigma) \leftarrow\$ A^{\textsc{SignSim}}(1^\lambda, pp, pk)$
If $(\mathsf{DS.Ver}(1^\lambda, pp, pk, M, \sigma)$ and $(M, \sigma) \notin Q)$ then $b' \leftarrow 1$
Else $b' \leftarrow 0$
Return $b'$

$\textsc{SignSim}(M)$
$\sigma \leftarrow\$ \textsc{Sign}(sk, M)$ ; $Q \leftarrow Q \cup \{(M, \sigma)\}$
Ret $\sigma$
```

When the challenge bit b in game SIM is 0, adversary A_1 simulates for A game G_0, while if $b = 1$, adversary A_1 simulates game SUF. We thus have

$$\Pr[\mathrm{SUF}^A_{\mathsf{DS}}(\lambda)] - \Pr[\mathrm{G}^A_0(\lambda)] = \Pr[b' = 1 \mid b = 1] - \Pr[b' = 1 \mid b = 0] \leq \mathbf{Adv}^{\mathrm{sim}}_{\mathsf{DS},A_1}(\lambda) ,$$

which establishes Equation (1). Adversary A_2 behaves as follows:

```
$A_2^{\textsc{Sign}}(1^\lambda, pp)$
$(sk, pk) \leftarrow\$ \mathsf{DS.Kg}(1^\lambda, pp)$ ; $(M, \sigma) \leftarrow\$ A^{\textsc{SignSim}}(1^\lambda, pp, pk)$
Ret $(pk, M, \sigma)$

$\textsc{SignSim}(M)$
$\sigma \leftarrow\$ \textsc{Sign}(sk, M)$
Ret $\sigma$
```

We skip the simple analysis establishing Equation (2). Adversary I behaves as follows:

```
$I(1^\lambda, fp, pk)$
$(ap, std, xtd) \leftarrow\$ \mathsf{DS.SimPg}(1^\lambda)$
$(M, \sigma) \leftarrow\$ A^{\textsc{SignSim}}(1^\lambda, (fp, ap), pk)$
$sk' \leftarrow\$ \mathsf{DS.Ext}(1^\lambda, pp, xtd, pk, M, \sigma)$
Ret $sk'$

$\textsc{SignSim}(M)$
$\sigma \leftarrow\$ \mathsf{DS.SimSig}(1^\lambda, pp, std, pk, M)$
Ret $\sigma$
```

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

$\textsc{Prove}(x, w)$
If not $\mathsf{R}(x, w)$ then Ret false
If $b = 1$ then $\pi \leftarrow_{\$} \Pi.\mathsf{P}(1^\lambda, crs_1, x, w)$
Else $\pi \leftarrow_{\$} \Pi.\mathsf{SimP}(1^\lambda, crs_0, std, x)$
Ret $\pi$
```

```
MAIN $\mathrm{SE}^{A}_{\Pi,\mathsf{R}}(\lambda)$
$Q \leftarrow \emptyset$ ; $(crs, std, xtd) \leftarrow_{\$} \Pi.\mathsf{SimPg}(1^\lambda)$
$(x, \pi) \leftarrow_{\$} A^{\textsc{Prove}}(1^\lambda, crs)$
If $x \notin L(\mathsf{R})$ then Ret false
If not $\Pi.\mathsf{V}(1^\lambda, crs, x, \pi)$ then Ret false
If $(x, \pi) \in Q$ then Ret false
$w \leftarrow_{\$} \Pi.\mathsf{Ext}(1^\lambda, crs, xtd, x, \pi)$
Ret not $\mathsf{R}(x, w)$

$\textsc{Prove}(x, w)$
If not $\mathsf{R}(x, w)$ then Ret $\perp$
$\pi \leftarrow_{\$} \Pi.\mathsf{SimP}(1^\lambda, crs, std, x)$ ; $Q \leftarrow Q \cup \{(x, \pi)\}$
Ret $\pi$
```

Figure 8: **Games defining security of NIZK system Π.** Left: Game defining zero knowledge. Right: Game defining simulation extractability.

We skip the simple analysis establishing Equation (3). ▌

B Construction: Proof of Theorem 3.2

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 Π for R specifies the following PT algorithms: via $crs \leftarrow_{\$} \Pi.\mathsf{Pg}(1^\lambda)$ one generates a common reference string crs; via $\pi \leftarrow_{\$} \Pi.\mathsf{P}(1^\lambda, crs, x, w)$ the prover given x and $w \in \mathsf{R}(x)$ generates a proof π that $x \in L(\mathsf{R})$; via $d \leftarrow \Pi.\mathsf{V}(1^\lambda, crs, x, \pi)$ 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 $\Pi.\mathsf{V}(1^\lambda, crs, x, \Pi.\mathsf{P}(1^\lambda, crs, x, w)) = \mathsf{true}$ for all $\lambda \in \mathbb{N}$, all $crs \in [\Pi.\mathsf{Pg}(\lambda)]$, all $x \in \{0,1\}^*$ and all $w \in \mathsf{R}(x)$. We say that Π is zero-knowledge (ZK) if it specifies additional PT algorithms $\Pi.\mathsf{SimPg}$ and $\Pi.\mathsf{SimP}$ such that $\mathbf{Adv}^{\mathrm{zk}}_{\Pi,\mathsf{R},A}(\cdot)$ is negligible for every PT adversary A, where $\mathbf{Adv}^{\mathrm{zk}}_{\Pi,\mathsf{R},A}(\lambda) = 2\Pr[\mathrm{ZK}^{A}_{\Pi,\mathsf{R}}(\lambda)] - 1$ and game ZK is specified on the left-hand side of Figure 8. This definition is based on [22, 37]. We say that Π is simulation-extractable (SE) if it specifies an additional PT algorithm $\Pi.\mathsf{Ext}$ such that $\mathbf{Adv}^{\mathrm{se}}_{\Pi,\mathsf{R},A}(\cdot)$ is negligible for every PT adversary A, where $\mathbf{Adv}^{\mathrm{se}}_{\Pi,\mathsf{R},A}(\lambda) = \Pr[\mathrm{SE}^{A}_{\Pi,\mathsf{R}}(\lambda)]$ and game SE is specified on the right-hand side of Figure 8. This definition is based on [37, 45, 46, 38].

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 [45], but for a fairly restricted language related to sets of pairing product equations in bilinear groups. In [38] (and further formalized in [48]), 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**.

CONSTRUCTION. Let F be the function family given in the theorem statement. We associate to it the **NP**-relation R defined by $\mathsf{R}((1^\lambda, fp, pk, M), sk) = (\mathsf{F.Ev}(1^\lambda, fp, sk) = pk)$ for all $\lambda \in \mathbb{N}$ and all $fp, pk, M, sk \in \{0,1\}^*$. Let Π be a NI system for R that is zero knowledge and simulation extractable. The signature scheme $\mathsf{DS} = \mathbf{KvS}[\mathsf{F}]$ is specified as follows:

- $\underline{\mathsf{DS.Pg}(1^\lambda)}$: $crs \leftarrow_{\$} \Pi.\mathsf{Pg}(1^\lambda)$; $fp \leftarrow_{\$} \mathsf{F.Pg}(1^\lambda)$. Return (fp, crs).

- $\underline{\mathsf{DS.Kg}(1^\lambda, (fp, crs))}$: $sk \leftarrow\$ \mathsf{F.Dom}(1^\lambda, fp)$; $pk \leftarrow \mathsf{F.Ev}(1^\lambda, fp, sk)$; Ret (sk, pk).
- $\underline{\mathsf{DS.Sig}(1^\lambda, (fp, crs), sk, M)}$: $pk \leftarrow \mathsf{F.Ev}(1^\lambda, fp, sk)$; Ret $\mathsf{\Pi.P}(1^\lambda, crs, (1^\lambda, fp, pk, M), sk)$.
- $\underline{\mathsf{DS.Ver}(1^\lambda, (fp, crs), pk, M, \sigma)}$: Ret $\mathsf{\Pi.V}(1^\lambda, crs, (1^\lambda, fp, pk, M), \sigma)$.
- $\underline{\mathsf{DS.SimPg}(1^\lambda)}$: Ret $\mathsf{\Pi.SimPg}(1^\lambda)$.
- $\underline{\mathsf{DS.SimSig}(1^\lambda, (fp, crs), std, pk, M)}$: Ret $\mathsf{\Pi.SimP}(1^\lambda, crs, std, (1^\lambda, fp, pk, M))$.
- $\underline{\mathsf{DS.Ext}(1^\lambda, (fp, crs), xtd, pk, M, \sigma)}$: Ret $\mathsf{\Pi.Ext}(1^\lambda, crs, xtd, (1^\lambda, fp, pk, M), \sigma)$.

Security of the construction. Simulatability of the signature scheme follows directly from the zero knowledge property of the NIZK. Let A be a PT adversary playing game SIM. We construct a PT adversary B such that $\mathbf{Adv}^{\mathrm{sim}}_{\mathsf{DS},A}(\lambda) \le \mathbf{Adv}^{\mathrm{zk}}_{\mathsf{\Pi},B}(\lambda)$ for all $\lambda \in \mathbb{N}$. B behaves as follows:

```
$\underline{B^{\text{Prove}}(1^\lambda, crs)}$
$fp \leftarrow\$ \mathsf{F.Pg}(1^\lambda)$ ; $pp \leftarrow (fp, crs)$
$b' \leftarrow\$ A^{\text{SignSim}}(1^\lambda, pp)$
Return $b'$

$\underline{\text{SignSim}(sk, M)}$
If $sk \notin \mathsf{F.Dom}(1^\lambda, fp)$ then Ret $\perp$
$pk \leftarrow \mathsf{F.Ev}(1^\lambda, fp, sk)$ ; $\pi \leftarrow\$ \text{Prove}((1^\lambda, fp, pk, M), sk)$
Ret $\pi$
```

The key extractability of the signature scheme likewise follows from the SE security of the NIZK. Let A be a PT adversary playing game EXT. We construct a PT adversary B such that $\mathbf{Adv}^{\mathrm{ext}}_{\mathsf{DS},A}(\lambda) \le \mathbf{Adv}^{\mathrm{se}}_{\mathsf{\Pi},B}(\lambda)$ for all $\lambda \in \mathbb{N}$. B behaves as follows:

```
$\underline{B^{\text{Prove}}(1^\lambda, crs)}$
$fp \leftarrow\$ \mathsf{F.Pg}(1^\lambda)$ ; $pp \leftarrow (fp, crs)$
$(pk, M, \sigma) \leftarrow\$ A^{\text{SignSim}}(1^\lambda, pp)$
Return $((1^\lambda, fp, pk, M), \sigma)$

$\underline{\text{SignSim}(sk, M)}$
If $sk \notin \mathsf{F.Dom}(1^\lambda, fp)$ then Ret $\perp$
$pk \leftarrow \mathsf{F.Ev}(1^\lambda, fp, sk)$ ; $\pi \leftarrow\$ \text{Prove}((1^\lambda, fp, pk, M), sk)$
Ret $\pi$
```

If $(pk, M, \sigma) \notin Q$ in game EXT then $((1^\lambda, fp, pk, M), \sigma) \notin Q$ in game SE, the sets being those defined in the games. Furthermore, by the definition of $\mathsf{DS.Ext}$ and R, if $sk \leftarrow \mathsf{DS.Ext}(1^\lambda, crs, xtd, pk, M, \sigma)$ is such that $\mathsf{F.Ev}(1^\lambda, fp, sk) \ne pk$, then $\mathsf{R}((1^\lambda, fp, pk, M), sk) = \mathsf{false}$.

C Proof of Theorem 4.1

We say PKE scheme PKE is 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 9. The adversary is allowed only one query to LR. This definition is from [10, 35]. We proceed to prove Theorem 4.1.

Proof: Part (1): IND security

Let A be a PT adversary playing game IND. We build PT adversaries A_1, D such that

$$\mathbf{Adv}^{\mathrm{ind}}_{\mathsf{JES},A}(\lambda) \le \mathbf{Adv}^{\text{ind-cca}}_{\mathsf{PKE},D}(\lambda) + 2\mathbf{Adv}^{\mathrm{sim}}_{\mathsf{DS},A_1}(\lambda)$$

for all $\lambda \in \mathbb{N}$, from which part (1) of the theorem follows.

The proof uses the game in Figure 10. This game switches to using simulated parameters and signatures. We will build A_1, D so that for all $\lambda \in \mathbb{N}$ we have

$$\Pr[\text{IND}^A_{\mathsf{JES}}(\lambda)] - \Pr[\text{G}^A_0(\lambda)] \le \mathbf{Adv}^{\mathrm{sim}}_{\mathsf{DS},A_1}(\lambda) \quad (4)$$

$$2\Pr[\text{G}^A_0(\lambda)] - 1 \le \mathbf{Adv}^{\text{ind-cca}}_{\mathsf{PKE},D}(\lambda) \; . \quad (5)$$

```
main IND-CCA$^A_{\mathsf{PKE}}(\lambda)$
$b \leftarrow\$ \{0,1\}$ ; $C^* \leftarrow \perp$
$fp \leftarrow\$ \mathsf{PKE.Pg}(1^\lambda)$ ; $(pk, sk) \leftarrow\$ \mathsf{PKE.Kg}(1^\lambda, fp)$
$b' \leftarrow\$ A^{\textsc{Dec},\textsc{Sign},\mathrm{LR}}(1^\lambda, fp, pk)$
Ret $(b = b')$

proc $\textsc{Dec}(C)$
If $(C = C^*)$ then Ret $\perp$
Ret $M \leftarrow \mathsf{PKE.Dec}(1^\lambda, fp, sk, C)$

proc $\mathrm{LR}(M_0, M_1)$
If $(|M_0| \neq |M_1|)$ then Ret $\perp$
$C^* \leftarrow\$ \mathsf{PKE.Enc}(1^\lambda, fp, pk, M_b)$
Ret $C^*$
```

Figure 9: **Game defining IND-CCA security of PKE scheme PKE.**

```
main $\mathrm{G}_0^A(\lambda)$
$b \leftarrow\$ \{0,1\}$ ; $C^* \leftarrow \perp$
$fp \leftarrow\$ \mathsf{F.Pg}(1^\lambda)$ ; $(ap, std, xtd) \leftarrow\$ \mathsf{DS.SimPg}(1^\lambda)$ ; $jp \leftarrow (fp, ap)$ ; $(sk, pk) \leftarrow\$ \mathsf{DS.Kg}(1^\lambda, jp)$
$b' \leftarrow\$ A^{\textsc{Dec},\textsc{Sign},\mathrm{LR}}(1^\lambda, jp, pk)$
Ret $(b = b')$

proc $\textsc{Dec}(C)$
If $(C = C^*)$ then Ret $\perp$
Ret $M \leftarrow \mathsf{JES.Dec}(1^\lambda, jp, sk, C)$

proc $\textsc{Sign}(M)$
Ret $\mathsf{DS.SimSig}(1^\lambda, jp, std, pk, M)$

proc $\mathrm{LR}(M_0, M_1)$
If $(|M_0| \neq |M_1|)$ then Ret $\perp$
$C^* \leftarrow\$ \mathsf{JES.Enc}(1^\lambda, jp, pk, M_b)$
Ret $C^*$
```

Figure 10: **Game used in the proof of part (1) of Theorem 4.1**

Using this we have

$$\begin{aligned}
\mathbf{Adv}^{\text{ind-cca}}_{\mathsf{PKE},A}(\lambda) &= 2\Pr[\text{IND-CCA}^A_{\mathsf{PKE}}(\lambda)] - 1 \\
&= 2\left(\Pr[\text{IND-CCA}^A_{\mathsf{PKE}}(\lambda)] - \Pr[\mathrm{G}_0^A(\lambda)] + \Pr[\mathrm{G}_0^A(\lambda)]\right) - 1 \\
&= 2\left(\Pr[\text{IND-CCA}^A_{\mathsf{PKE}}(\lambda)] - \Pr[\mathrm{G}_0^A(\lambda)]\right) + 2\Pr[\mathrm{G}_0^A(\lambda)] - 1 \\
&\leq 2\mathbf{Adv}^{\text{sim}}_{\mathsf{DS},A_1}(\lambda) + \mathbf{Adv}^{\text{ind-cca}}_{\mathsf{PKE},D}(\lambda)
\end{aligned}$$

as desired. We proceed to the constructions of A_1, D. Adversary A_1 behaves as follows:

```
MAIN $\mathrm{G}_0^A(\lambda)$ / $\boxed{\mathrm{G}_1^A(\lambda)}$
$Q \leftarrow \emptyset$ ; $d \leftarrow \mathsf{false}$
$fp \leftarrow\$ \mathsf{F.Pg}(1^\lambda)$ ; $(ap, std, xtd) \leftarrow\$ \mathsf{DS.SimPg}(1^\lambda)$ ; $jp \leftarrow (fp, ap)$ ; $(sk, pk) \leftarrow\$ \mathsf{DS.Kg}(1^\lambda, jp)$
$(M, \sigma) \leftarrow\$ A^{\mathrm{SIGN},\mathrm{DEC}}(1^\lambda, jp, pk)$ ; $sk' \leftarrow\$ \mathsf{DS.Ext}(1^\lambda, jp, xtd, pk, M, \sigma)$
If $(\mathsf{DS.Ver}(1^\lambda, jp, pk, M, \sigma)$ and $(M, \sigma) \notin Q)$ then
    $d \leftarrow \mathsf{true}$
    If $(\mathsf{F.Ev}(1^\lambda, fp, sk') \neq pk)$ then $\mathsf{bad} \leftarrow \mathsf{true}$ ; $\boxed{d \leftarrow \mathsf{false}}$
Ret $d$

proc SIGN($M$)
$\sigma \leftarrow\$ \mathsf{DS.SimSig}(1^\lambda, jp, std, pk, M)$
$Q \leftarrow Q \cup \{(M, \sigma)\}$
Ret $\sigma$

proc DEC($C$)
Ret $M \leftarrow \mathsf{Dec}(1^\lambda, jp, sk, C)$
```

Figure 11: **Games used in the proof of part (2) of Theorem 4.1:** Game G_1 includes the boxed code and G_0 does not.

```
$A_1^{\mathrm{SIGN}}(1^\lambda, pp)$
$(sk, pk) \leftarrow\$ \mathsf{DS.Kg}(1^\lambda, pp)$
$C^* \leftarrow \bot$ ; $d \leftarrow\$ \{0,1\}$
$d' \leftarrow\$ A^{\mathrm{DECSIM},\mathrm{SIGNSIM},\mathrm{LRSIM}}(1^\lambda, pp, pk)$
If $(d' = d)$ then $b' \leftarrow 1$
Else $b' \leftarrow 0$
Ret $b'$
```

```
SIGNSIM($M$)
$\sigma \leftarrow\$ \mathrm{SIGN}(sk, M)$
Ret $\sigma$
```

```
DECSIM($C$)
If $C = C^*$ then $M \leftarrow \bot$
Else $M \leftarrow \mathsf{JES.Dec}(1^\lambda, pp, sk, C)$
Ret $M$

LRSIM($M_0, M_1$)
If $(|M_0| \neq |M_1|)$ then Ret $\bot$
$C^* \leftarrow \mathsf{JES.Enc}(1^\lambda, pp, pk, M_d)$
Ret $C^*$
```

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. We thus have

$$\Pr[\mathrm{IND}_{\mathsf{JES}}^A(\lambda)] - \Pr[\mathrm{G}_0^A(\lambda)] = \Pr[b' = 1 \,|\, b = 1] - \Pr[b' = 1 \,|\, b = 0] \leq \mathbf{Adv}_{\mathsf{DS},A_1}^{\mathrm{sim}}(\lambda) ,$$

establishing Equation (4). Adversary D behaves as follows:

```
$D^{\mathrm{DEC},\mathrm{LR}}(1^\lambda, fp, pk)$
$(ap, std, xtd) \leftarrow\$ \mathsf{DS.SimPg}(1^\lambda)$ ; $jp \leftarrow (fp, ap)$
$b' \leftarrow\$ A^{\mathrm{DEC},\mathrm{SIGNSIM},\mathrm{LR}}(1^\lambda, jp, pk)$
Ret $b'$
```

```
SIGNSIM($M$)
$\sigma \leftarrow\$ \mathsf{DS.SimSig}(1^\lambda, jp, std, pk, M)$
Ret $\sigma$
```

We omit the analysis establishing Equation (5).

Part (2): SUF security

Let A be a PT adversary playing game SUF. We build PT adversaries A_1, A_2, D such that

$$\mathbf{Adv}_{\mathsf{JES},A}^{\mathrm{suf}}(\lambda) \leq \mathbf{Adv}_{\mathsf{PKE},D}^{\text{ind-cca}}(\lambda) + \mathbf{Adv}_{\mathsf{DS},A_1}^{\mathrm{sim}}(\lambda) + \mathbf{Adv}_{\mathsf{DS},A_2}^{\mathrm{ext}}(\lambda)$$

for all $\lambda \in \mathbb{N}$, from which part (2) of the theorem follows.

The proof uses the games in Figure 11. These games 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

$$\Pr[\mathrm{SUF}^A_{\mathsf{JES}}(\lambda)] - \Pr[\mathrm{G}^A_0(\lambda)] \leq \mathbf{Adv}^{\mathrm{sim}}_{\mathsf{DS},A_1}(\lambda) \tag{6}$$

$$\Pr[\mathrm{G}^A_0(\lambda) \text{ sets } \mathsf{bad}] \leq \mathbf{Adv}^{\mathrm{ext}}_{\mathsf{DS},A_2}(\lambda) \tag{7}$$

$$\Pr[\mathrm{G}^A_1(\lambda)] \leq \mathbf{Adv}^{\text{ind-cca}}_{\mathsf{PKE},D}(\lambda)\ . \tag{8}$$

Games G_0 and G_1 are identical until bad, so by the Fundamental Lemma of Game-Playing [17] and the above, for all $\lambda \in \mathbb{N}$ we have:

$$\begin{aligned}
\mathbf{Adv}^{\mathrm{suf}}_{\mathsf{JES},A}(\lambda) &= \Pr[\mathrm{SUF}^A_{\mathsf{JES}}(\lambda)]\\
&= (\Pr[\mathrm{SUF}^A_{\mathsf{JES}}(\lambda)] - \Pr[\mathrm{G}^A_0(\lambda)]) + (\Pr[\mathrm{G}^A_0(\lambda)] - \Pr[\mathrm{G}^A_1(\lambda)]) + \Pr[\mathrm{G}^A_1(\lambda)]\\
&\leq (\Pr[\mathrm{SUF}^A_{\mathsf{JES}}(\lambda)] - \Pr[\mathrm{G}^A_0(\lambda)]) + \Pr[\mathrm{G}^A_0(\lambda) \text{ sets } \mathsf{bad}] + \Pr[\mathrm{G}^A_1(\lambda)]\\
&\leq \mathbf{Adv}^{\mathrm{sim}}_{\mathsf{DS},A_1}(\lambda) + \mathbf{Adv}^{\mathrm{ext}}_{\mathsf{DS},A_2}(\lambda) + \mathbf{Adv}^{\text{ind-cca}}_{\mathsf{PKE},D}(\lambda)
\end{aligned}$$

as desired. We proceed to the constructions of A_1, A_2, D. Adversary A_1 behaves as follows:

```
$A_1^{\mathrm{SIGN}}(1^\lambda, pp)$
$(sk, pk) \leftarrow\$ \mathsf{DS.Kg}(1^\lambda, pp)$ ; $Q \leftarrow \emptyset$
$(M, \sigma) \leftarrow\$ A^{\mathrm{DECSIM},\mathrm{SIGNSIM}}(1^\lambda, pp, pk)$
If ($\mathsf{DS.Ver}(1^\lambda, pp, pk, M, \sigma)$ and $(M, \sigma) \notin Q$) then $b' \leftarrow 1$
Else $b' \leftarrow 0$
Return $b'$

$\mathrm{DECSIM}(C)$
$M \leftarrow \mathsf{JES.Dec}(1^\lambda, pp, sk, C)$
Ret $M$

$\mathrm{SIGNSIM}(M)$
$Q \leftarrow Q \cup \{(M, \sigma)\}$
$\sigma \leftarrow\$ \mathrm{SIGN}(sk, M)$
Ret $\sigma$
```

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. We thus have

$$\Pr[\mathrm{SUF}^A_{\mathsf{JES}}(\lambda)] - \Pr[\mathrm{G}^A_0(\lambda)] = \Pr[b' = 1 \mid b = 1] - \Pr[b' = 1 \mid b = 0] \leq \mathbf{Adv}^{\mathrm{sim}}_{\mathsf{DS},A_1}(\lambda)\ ,$$

establishing Equation (6). Adversary A_2 behaves as follows:

```
$A_2^{\mathrm{SIGN}}(1^\lambda, pp)$
$(sk, pk) \leftarrow\$ \mathsf{DS.Kg}(1^\lambda, pp)$
$(M, \sigma) \leftarrow\$ A^{\mathrm{DECSIM},\mathrm{SIGNSIM}}(1^\lambda, pp, pk)$
Ret $(pk, M, \sigma)$

$\mathrm{DECSIM}(C)$
$M \leftarrow \mathsf{JES.Dec}(1^\lambda, pp, sk, C)$
Ret $M$

$\mathrm{SIGNSIM}(M)$
$\sigma \leftarrow\$ \mathrm{SIGN}(sk, M)$
Ret $\sigma$
```

We omit the analysis establishing Equation (7). Adversary D behaves as follows:

```
$D^{\mathrm{DEC},\mathrm{LR}}(1^\lambda, fp, pk)$
$(ap, std, xtd) \leftarrow\$ \mathsf{DS.SimPg}(1^\lambda)$ ; $jp \leftarrow (fp, ap)$
$(M, \sigma) \leftarrow\$ A^{\mathrm{DEC},\mathrm{SIGNSIM}}(1^\lambda, jp, pk)$
$sk' \leftarrow\$ \mathsf{DS.Ext}(1^\lambda, pp, xtd, pk, M, \sigma)$
If ($\mathsf{F.Ev}(1^\lambda, fp, sk') \neq pk$) then Ret 0
$M_0 \leftarrow 0^\lambda$ ; $M_1 \leftarrow 1^\lambda$
$C^* \leftarrow\$ \mathrm{LR}(M_0, M_1)$ ; $M \leftarrow \mathsf{PKE.Dec}(1^\lambda, fp, sk', C^*)$
If ($M = M_1$) then $b' \leftarrow 1$ else $b' \leftarrow 0$
Ret $b'$

$\mathrm{SIGNSIM}(M)$
$\sigma \leftarrow\$ \mathsf{DS.SimSig}(1^\lambda, jp, std, pk, M)$
Ret $\sigma$
```

```
MAIN $G_0^A(\lambda)$ / $\boxed{G_1^A(\lambda)}$
$Q \leftarrow \emptyset$ ; $d \leftarrow \mathsf{false}$
$fp \leftarrow_\$ \mathsf{F.Pg}(1^\lambda)$ ; $(ap, std, xtd) \leftarrow_\$ \mathsf{DS.SimPg}(1^\lambda)$ ; $pp \leftarrow (fp, ap)$ ; $(sk, pk) \leftarrow_\$ \mathsf{DS.Kg}(1^\lambda, pp)$
$(M, \sigma) \leftarrow_\$ A^{\mathrm{SIGN}}(1^\lambda, pp, pk)$ ; $sk' \leftarrow_\$ \mathsf{DS.Ext}(1^\lambda, pp, xtd, pk, M, \sigma)$
If ($\mathsf{DS.Ver}(1^\lambda, pp, pk, M, \sigma)$ and $(pk, M, \sigma) \notin Q$) then
    $d \leftarrow \mathsf{true}$
    If ($\mathsf{F.Ev}(1^\lambda, fp, sk') \neq pk$) then $\mathsf{bad} \leftarrow \mathsf{true}$ ; $\boxed{d \leftarrow \mathsf{false}}$
Ret $d$

SIGN($\phi, M$)
$sk' \leftarrow \Phi(1^\lambda, fp, \phi, sk)$ ; $pk' \leftarrow \mathsf{F.Ev}(1^\lambda, fp, sk')$
$\sigma \leftarrow_\$ \mathsf{DS.SimSig}(1^\lambda, pp, std, pk', M)$
$Q \leftarrow Q \cup \{(pk', M, \sigma)\}$
Ret $\sigma$
```

Figure 12: **Games used in proof of Theorem 5.1:** Game G_1 includes the boxed code and G_0 does not.

When A wins G_1 we have that $\mathsf{F.Ev}(1^\lambda, fp, sk') = pk$, so sk' is a valid secret key for pk. By correctness of PKE, we then have $\mathsf{PKE.Dec}(1^\lambda, pp, sk', \mathsf{PKE.Enc}(1^\lambda, pp, pk, M_b)) = M_b$, where b is the challenge bit in game IND-CCA, so

$$\mathbf{Adv}^{\text{ind-cca}}_{\mathsf{PKE},D}(\lambda) = \Pr[b' = 1 | b = 1] - \Pr[b' = 1 | b = 0] \geq \Pr[G_1^A(\lambda)] .$$

This is because the first term in the difference above is at least $\Pr[G_1^A(\lambda)]$ and the second term is zero. This establishes Equation (8). ▌

D Proof of Theorem 5.1

Proof: Let A be a PT adversary playing game RKASIG. We build PT adversaries A_1, A_2, I such that

$$\mathbf{Adv}^{\text{rka}}_{\mathsf{DS},A,\Phi}(\lambda) \leq \mathbf{Adv}^{\text{rka}}_{\mathsf{F},I,\Phi}(\lambda) + \mathbf{Adv}^{\text{sim}}_{\mathsf{DS},A_1}(\lambda) + \mathbf{Adv}^{\text{ext}}_{\mathsf{DS},A_2}(\lambda)$$

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

The proof uses the games in Figure 12. These games 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

$$\Pr[\text{RKASIG}^A_{\mathsf{DS},\Phi}(\lambda)] - \Pr[G_0^A(\lambda)] \leq \mathbf{Adv}^{\text{sim}}_{\mathsf{DS},A_1}(\lambda) \tag{9}$$

$$\Pr[G_0^A(\lambda) \text{ sets } \mathsf{bad}] \leq \mathbf{Adv}^{\text{ext}}_{\mathsf{DS},A_2}(\lambda) \tag{10}$$

$$\Pr[G_1^A(\lambda)] \leq \mathbf{Adv}^{\text{rka}}_{\mathsf{F},I,\Phi}(\lambda) . \tag{11}$$

Games G_0 and G_1 are identical until bad, so by the Fundamental Lemma of Game-Playing [17] and the above, for all $\lambda \in \mathbb{N}$ we have:

$$\begin{aligned}
\mathbf{Adv}^{\text{rka}}_{\mathsf{DS},A,\Phi}(\lambda) &= \Pr[\text{RKASIG}^A_{\mathsf{DS},\Phi}(\lambda)] \\
&= (\Pr[\text{RKASIG}^A_{\mathsf{DS},\Phi}(\lambda)] - \Pr[G_0^A(\lambda)]) + (\Pr[G_0^A(\lambda)] - \Pr[G_1^A(\lambda)]) + \Pr[G_1^A(\lambda)] \\
&\leq (\Pr[\text{RKASIG}^A_{\mathsf{DS},\Phi}(\lambda)] - \Pr[G_0^A(\lambda)]) + \Pr[G_0^A(\lambda) \text{ sets } \mathsf{bad}] + \Pr[G_1^A(\lambda)] \\
&\leq \mathbf{Adv}^{\text{sim}}_{\mathsf{DS},A_1}(\lambda) + \mathbf{Adv}^{\text{ext}}_{\mathsf{DS},A_2}(\lambda) + \mathbf{Adv}^{\text{rka}}_{\mathsf{F},I,\Phi}(\lambda)
\end{aligned}$$

as desired. We proceed to the constructions of A_1, A_2, I. Adversary A_1 behaves as follows:

```
$\underline{A_1^{\text{SIGN}}(1^\lambda, pp)}$
$(sk, pk) \leftarrow\$ \mathsf{DS.Kg}(1^\lambda, pp)$ ; $Q \leftarrow \emptyset$
$(M, \sigma) \leftarrow\$ A^{\text{SIGNSIM}}(1^\lambda, pp, pk)$
If ($\mathsf{DS.Ver}(1^\lambda, pp, pk, M, \sigma)$ and $(pk, M, \sigma) \notin Q$) then $b' \leftarrow 1$
Else $b' \leftarrow 0$
Return $b'$

$\underline{\text{SIGNSIM}(\phi, M)}$
$sk' \leftarrow \Phi(1^\lambda, fp, \phi, sk)$
$pk' \leftarrow \mathsf{F.Ev}(1^\lambda, fp, sk')$
$\sigma \leftarrow\$ \text{SIGN}(sk', M)$
$Q \leftarrow Q \cup \{(pk', M, \sigma)\}$
Ret $\sigma$
```

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 RKASIG. We thus have

$$\Pr[\text{RKASIG}^A_{\mathsf{DS},\Phi}(\lambda)] - \Pr[G_0^A(\lambda)] = \Pr[b' = 1 \,|\, b = 1] - \Pr[b' = 1 \,|\, b = 0] \leq \mathbf{Adv}^{\text{sim}}_{\mathsf{DS},A_1}(\lambda) \,,$$

establishing Equation (9). Adversary A_2 behaves as follows:

```
$\underline{A_2^{\text{SIGN}}(1^\lambda, pp)}$
$(sk, pk) \leftarrow\$ \mathsf{DS.Kg}(1^\lambda, pp)$
$(M, \sigma) \leftarrow\$ A^{\text{SIGNSIM}}(1^\lambda, pp, pk)$
Ret $(pk, M, \sigma)$

$\underline{\text{SIGNSIM}(\phi, M)}$
$sk' \leftarrow \Phi(1^\lambda, fp, \phi, sk)$ ; $pk' \leftarrow \mathsf{F.Ev}(1^\lambda, fp, sk')$
$\sigma \leftarrow\$ \text{SIGN}(sk', M)$ ; $Q \leftarrow Q \cup \{(pk', M, \sigma)\}$
Ret $\sigma$
```

If bad is set to true in game G_0 then we have: (1) $\mathsf{DS.Ver}(1^\lambda, pp, pk, M, \sigma)$ (2) $(pk, M, \sigma) \notin Q$, and (3) $\mathsf{F.Ev}(1^\lambda, fp, sk') \neq pk$. These are exactly the necessary conditions for A_2 to win game EXT, establishing Equation (10). I behaves as follows:

```
$\underline{I^{\text{EVAL}}(1^\lambda, fp, pk)}$
$(ap, std, xtd) \leftarrow\$ \mathsf{DS.SimPg}(1^\lambda)$
$(M, \sigma) \leftarrow\$ A^{\text{SIGNSIM}}(1^\lambda, (fp, ap), pk)$
$sk' \leftarrow\$ \mathsf{DS.Ext}(1^\lambda, pp, xtd, pk, M, \sigma)$
Ret $sk'$

$\underline{\text{SIGNSIM}(\phi, M)}$
$pk' \leftarrow\$ \text{EVAL}(\phi)$
$\sigma \leftarrow\$ \mathsf{DS.SimSig}(1^\lambda, pp, std, pk', M)$
Ret $\sigma$
```

We omit the analysis establishing Equation (11). ▌

E Proof of Theorem 6.1

Proof: Part (1): 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 $\text{IND}^A_{\mathsf{ST},\Phi}(\lambda)$ 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

$$\mathbf{Adv}^{\text{ind}}_{\mathsf{ST},A,\Phi}(\lambda) \leq 2\mathbf{Adv}^{\text{sim}}_{\mathsf{DS},A_1}(\lambda) + \mathbf{Adv}^{\text{kdm}}_{\mathsf{PKE},D,\Phi}(\lambda) + 2\nu(\lambda) + 2q(\lambda) \cdot \mathbf{Adv}^{\text{ext}}_{\mathsf{DS},A_2}(\lambda)$$

for all $\lambda \in \mathbb{N}$, from which part (1) of the theorem follows.

The proof uses the games in Figure 13. We build A_1, A_2, D and $\nu(\cdot)$ such that for all $\lambda \in \mathbb{N}$ we have

$$\Pr[\text{IND}^A_{\mathsf{ST},\Phi}(\lambda)] - \Pr[G_0^A(\lambda)] \leq \mathbf{Adv}^{\text{sim}}_{\mathsf{DS},A_1}(\lambda) \tag{12}$$

$$2\Pr[G_1^A(\lambda)] - 1 \leq \mathbf{Adv}^{\text{kdm}}_{\mathsf{PKE},D,\Phi}(\lambda) \tag{13}$$

$$\Pr[G_2^A(\lambda) \text{ sets } \mathsf{bad}] \leq \nu(\lambda) + q(\lambda) \cdot \mathbf{Adv}^{\text{ext}}_{\mathsf{DS},A_2}(\lambda) \,. \tag{14}$$

```
MAIN $\boxed{G_0^A(\lambda)}$ / $G_1^A(\lambda)$ / $G_2^A(\lambda)$
$Q \leftarrow \emptyset$ ; $Q' \leftarrow \emptyset$ ; $b \leftarrow_\$ \{0,1\}$
$fp \leftarrow_\$ \mathsf{F.Pg}(1^\lambda)$ ; $(ap, std, xtd) \leftarrow_\$ \mathsf{DS.SimPg}(1^\lambda)$ ; $pp \leftarrow (fp, ap)$
$b' \leftarrow_\$ A^{\text{MkKey,Store,Retrieve}}(1^\lambda, pp)$
Ret $(b = b')$

MkKey$(1^n)$   // $G_0^A(\lambda)$ / $G_1^A(\lambda)$ / $G_2^A(\lambda)$
For $i = 1, \ldots, n$ do $(\mathbf{sk}[i], \mathbf{pk}[i]) \leftarrow_\$ \mathsf{DS.Kg}(1^\lambda, pp)$
Ret $\mathbf{pk}$

proc Store$(\phi, i)$   // $G_0^A(\lambda)$ / $G_1^A(\lambda)$ / $G_2^A(\lambda)$
If not $(1 \le i \le n)$ then return $\perp$
If $(b = 1)$ then $M \leftarrow \Phi(1^\lambda, \phi, \mathbf{sk})$ else $M \leftarrow 0^{\phi.m}$
$C \leftarrow_\$ \mathsf{PKE.Enc}(1^\lambda, fp, \mathbf{pk}[i], M)$ ; $\sigma \leftarrow_\$ \mathsf{DS.SimSig}(1^\lambda, pp, std, \mathbf{pk}[i], C)$
$Q \leftarrow Q \cup \{((C, \sigma), i)\}$ ; $Q' \leftarrow Q' \cup \{(\mathbf{pk}[i], C, \sigma)\}$
Ret $(C, \sigma)$

proc Retrieve$((C, \sigma), i)$   // $\boxed{G_0^A(\lambda)}$ / $G_1^A(\lambda)$
If not $(1 \le i \le n)$ then return $\perp$
If (not $\mathsf{DS.Ver}(1^\lambda, pp, \mathbf{pk}[i], C, \sigma)$) then Ret $\perp$
If $(((C, \sigma), i) \in Q)$ then Ret $\perp$
$sk' \leftarrow_\$ \mathsf{DS.Ext}(1^\lambda, pp, xtd, \mathbf{pk}[i], C, \sigma)$ ; $pk' \leftarrow \mathsf{F.Ev}(1^\lambda, fp, sk')$
$M \leftarrow \mathsf{PKE.Dec}(1^\lambda, fp, sk', C)$
If $(pk' \neq \mathbf{pk}[i])$ then bad $\leftarrow$ true ; $\boxed{M \leftarrow \mathsf{PKE.Dec}(1^\lambda, fp, \mathbf{sk}[i], C)}$
Ret $M$

proc Retrieve$((C, \sigma), i)$   // $G_2^A(\lambda)$
If not $(1 \le i \le n)$ then return $\perp$
If (not $\mathsf{DS.Ver}(1^\lambda, pp, \mathbf{pk}[i], C, \sigma)$) then Ret $\perp$
If $(((C, \sigma), i) \in Q)$ then Ret $\perp$
$M \leftarrow \mathsf{PKE.Dec}(1^\lambda, fp, \mathbf{sk}[i], C)$
If $(\mathbf{pk}[i], C, \sigma) \in Q'$ then bad $\leftarrow$ true ; Ret $M$
$sk' \leftarrow_\$ \mathsf{DS.Ext}(1^\lambda, pp, xtd, \mathbf{pk}[i], C, \sigma)$ ; $pk' \leftarrow \mathsf{F.Ev}(1^\lambda, fp, sk')$
If $(pk' \neq \mathbf{pk}[i])$ then bad $\leftarrow$ true
Ret $M$
```

Figure 13: **Games used in the proof of part (1) of Theorem 6.1.** Game G_0 includes the boxed code and game G_1 does not.

Games G_0, G_1 are identical until bad. We also observe that $\Pr[G_0^A(\lambda) \text{ sets bad}] \le \Pr[G_2^A(\lambda) \text{ sets bad}]$. (In both games, decryption in Retrieve is always done correctly.) Combining this with the above and the Fundamental Lemma of Game-Playing [17], for all $\lambda \in \mathbb{N}$ we have:

$$\begin{aligned}
\mathbf{Adv}^{\mathrm{ind}}_{\mathsf{ST},A,\Phi}(\lambda) &= 2\Pr[\mathrm{IND}^A_{\mathsf{ST},\Phi}(\lambda)] - 1 \\
&= 2\left(\Pr[\mathrm{IND}^A_{\mathsf{ST},\Phi}(\lambda) - \Pr[G_0^A(\lambda)]\right) + 2\left(\Pr[G_0^A(\lambda)] - \Pr[G_1^A(\lambda)]\right) + 2\Pr[G_1^A(\lambda)] - 1 \\
&\le 2\left(\Pr[\mathrm{IND}^A_{\mathsf{ST},\Phi}(\lambda) - \Pr[G_0^A(\lambda)]\right) + 2\Pr[G_0^A(\lambda) \text{ sets bad}] + 2\Pr[G_1^A(\lambda)] - 1 \\
&\le 2\mathbf{Adv}^{\mathrm{sim}}_{\mathsf{DS},A_1}(\lambda) + \mathbf{Adv}^{\mathrm{kdm}}_{\mathsf{PKE},D,\Phi}(\lambda) + 2\Pr[G_0^A(\lambda) \text{ sets bad}] \\
&\le 2\mathbf{Adv}^{\mathrm{sim}}_{\mathsf{DS},A_1}(\lambda) + \mathbf{Adv}^{\mathrm{kdm}}_{\mathsf{PKE},D,\Phi}(\lambda) + 2\Pr[G_2^A(\lambda) \text{ sets bad}] \\
&\le 2\mathbf{Adv}^{\mathrm{sim}}_{\mathsf{DS},A_1}(\lambda) + \mathbf{Adv}^{\mathrm{kdm}}_{\mathsf{PKE},D,\Phi}(\lambda) + 2\nu(\lambda) + 2q(\lambda) \cdot \mathbf{Adv}^{\mathrm{ext}}_{\mathsf{DS},A_2}(\lambda)
\end{aligned}$$

as desired. We proceed to the constructions of A_1, A_2, D and ν.

Adversary A_1 behaves as follows:

```
$\underline{A_1^{\textsc{Sign}}(1^\lambda, pp)}$
$Q \leftarrow \emptyset$ ; $d \leftarrow\$ \{0,1\}$
$d' \leftarrow\$ A^{\textsc{MkKeySim},\textsc{StoreSim},\textsc{RetrieveSim}}(1^\lambda, pp)$
If $(d' = d)$ then $b' \leftarrow 1$ else $b' \leftarrow 0$
Ret $b'$

$\underline{\textsc{MkKeySim}(1^n)}$
For $i = 1, \ldots, n$ do
    $(\mathbf{sk}[i], \mathbf{pk}[i]) \leftarrow\$ \mathsf{DS.Kg}(1^\lambda, pp)$
Ret $\mathbf{pk}$

$\underline{\textsc{StoreSim}(\phi, i)}$
If not $(1 \le i \le n)$ then Ret $\perp$
If $d = 1$ then $M \leftarrow \Phi(1^\lambda, \phi, \mathbf{sk})$
Else $M \leftarrow 0^{\phi.m}$
$C \leftarrow\$ \mathsf{PKE.Enc}(1^\lambda, fp, \mathbf{pk}[i], M)$
$\sigma \leftarrow\$ \textsc{Sign}(\mathbf{sk}[i], C)$ ; $Q \leftarrow Q \cup \{((C,\sigma), i)\}$
Ret $(C, \sigma)$

$\underline{\textsc{RetrieveSim}((C,\sigma), i)}$
If not $(1 \le i \le n)$ then Ret $\perp$
If (not $\mathsf{DS.Ver}(1^\lambda, pp, \mathbf{pk}[i], C, \sigma)$) then Ret $\perp$
If $((C,\sigma), i) \in Q$ then Ret $\perp$
$M \leftarrow \mathsf{PKE.Dec}(1^\lambda, fp, \mathbf{sk}[i], C)$
Ret $M$
```

When the challenge bit b in game SIM is 1, adversary A_1 simulates IND. We claim that if $b = 0$ then A_1 simulates G_0. This is because in G_0, procedure $\textsc{Retrieve}$ always performs the correct decryption, regardless of whether or not bad is set, and so does A_1. (A_1 does not need to invoke the extractor, and indeed could not, since it does not have an extraction trapdoor.) We thus have

$$\Pr[\mathrm{IND}^{A}_{\mathsf{ST},\Phi}(\lambda)] - \Pr[\mathrm{G}_0^{A}(\lambda)] = \Pr[b' = 1 \mid b = 1] - \Pr[b' = 1 \mid b = 0] \le \mathbf{Adv}^{\mathrm{sim}}_{\mathsf{DS},A_1}(\lambda),$$

establishing Equation (12).

Adversary D behaves as follows:

```
$\underline{D^{\textsc{MkKey},\textsc{Enc}}(1^\lambda, fp)}$
$Q \leftarrow \emptyset$
$(ap, std, xtd) \leftarrow\$ \mathsf{DS.SimPg}(1^\lambda)$
$pp \leftarrow (fp, ap)$
$b' \leftarrow\$ A^{\textsc{MkKey},\textsc{StoreSim},\textsc{RetrieveSim}}(1^\lambda, pp)$
Ret $b'$

$\underline{\textsc{StoreSim}(\phi, i)}$
If not $(1 \le i \le n)$ then Ret $\perp$
$C \leftarrow\$ \textsc{Enc}(\phi, i)$
$\sigma \leftarrow\$ \mathsf{DS.SimSig}(1^\lambda, pp, std, \mathbf{pk}[i], C)$
$Q \leftarrow Q \cup \{((C,\sigma), i)\}$
Ret $(C, \sigma)$

$\underline{\textsc{RetrieveSim}((C,\sigma), i)}$
If not $(1 \le i \le n)$ then Ret $\perp$
If (not $\mathsf{DS.Ver}(1^\lambda, pp, \mathbf{pk}[i], C, \sigma)$) then Ret $\perp$
If $((C,\sigma), i) \in Q$ then Ret $\perp$
$sk' \leftarrow\$ \mathsf{DS.Ext}(1^\lambda, pp, xtd, \mathbf{pk}[i], C, \sigma)$
$M \leftarrow \mathsf{PKE.Dec}(1^\lambda, fp, sk', C)$
Ret $M$
```

We omit the analysis to establish Equation (13).

Now we would like to show that G_0 (equivalently, G_1) 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 formalized via G_2, which breaks the setting of bad from G_0 into two parts corresponding to the two events of interest. To establish Equation (17), let E_1 be the event that bad is set by the line 5 "If" statement, and E_2 the event that bad is set by the line 7 "If" statement. We first show the existence of a negligible $\nu(\cdot)$ such that $\Pr[E_1] \le \nu(\lambda)$. Then we build A_2 such that $\Pr[E_2 \wedge \overline{E}_1] \le q(\lambda) \cdot \mathbf{Adv}^{\mathrm{ext}}_{\mathsf{DS},A_2}(\lambda)$. This establishes Equation (14), as we have

$$\begin{aligned}\Pr[\mathrm{G}_2^A(\lambda) \text{ sets } \mathsf{bad}] &= \Pr[E_1 \vee E_2] \\ &= \Pr[E_1] + \Pr[E_2 \wedge \overline{E}_1] \\ &\le \nu(\lambda) + q(\lambda) \cdot \mathbf{Adv}^{\mathrm{ext}}_{\mathsf{DS},A_2}(\lambda).\end{aligned}$$

For the first claim, let E be the event that there is a collision in the public keys chosen in 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 $\Pr[E_1] \le \Pr[E]$. 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 key-pairs, 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 behaves as follows:

```
$A_2^{\mathrm{Sign}}(1^\lambda, pp)$
$Q \leftarrow \emptyset$ ; $d \leftarrow\$ \{0,1\}$ ; $j \leftarrow 0$
$d' \leftarrow\$ A^{\mathrm{MkKeySim},\mathrm{StoreSim},\mathrm{RetrieveSim}}(1^\lambda, pp)$
$\ell \leftarrow\$ \{1, \ldots, j\}$
Ret $(pk_\ell, C_\ell, \sigma_\ell)$

$\mathrm{MkKeySim}(1^n)$
For $i = 1, \ldots, n$ do
    $(\mathbf{sk}[i], \mathbf{pk}[i]) \leftarrow\$ \mathsf{DS.Kg}(1^\lambda, pp)$
Ret $\mathbf{pk}$

$\mathrm{StoreSim}(\phi, i)$
If not $(1 \le i \le n)$ then Ret $\perp$
If $d = 1$ then $M \leftarrow \Phi(1^\lambda, \phi, \mathbf{sk})$
Else $M \leftarrow 0^{\phi.m}$
$C \leftarrow\$ \mathsf{PKE.Enc}(1^\lambda, fp, \mathbf{pk}[i], M)$
$\sigma \leftarrow\$ \mathrm{Sign}(\mathbf{sk}[i], C)$ ; $Q \leftarrow Q \cup \{((C,\sigma), i)\}$
Ret $(C, \sigma)$

$\mathrm{RetrieveSim}((C,\sigma), i)$
If not $(1 \le i \le n)$ then Ret $\perp$
If (not $\mathsf{DS.Ver}(1^\lambda, pp, \mathbf{pk}[i], C, \sigma)$) then Ret $\perp$
If $((C,\sigma), i) \in Q$ then Ret $\perp$
$M \leftarrow \mathsf{PKE.Dec}(1^\lambda, fp, \mathbf{sk}[i], C)$
$j \leftarrow j + 1$ ; $pk_j \leftarrow \mathbf{pk}[i]$ ; $C_j \leftarrow C$ ; $\sigma_j \leftarrow \sigma$
Ret $M$
```

Adversary A_2 always performs correct decryptions in responding to Retrieve queries, following G_2. If bad is set at line 7 but not at line 5, then there is some tuple on which the extractor would succeed. Since a tuple is guessed at random we have $\Pr[E_2 \wedge \overline{E}_1] \le q(\lambda) \cdot \mathbf{Adv}^{\mathrm{ext}}_{\mathsf{DS},A_2}(\lambda)$ as desired. The importance of bad not being set at line 5 is that otherwise extraction is not required to succeed according to game EXT.

Part (2): SUF security

Let A' be a PT adversary playing game SUF. Our first step is to consider a simplified form of the SUF game shown in Figure 14. 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 $\mathbf{Adv}^{\mathrm{suf}}_{\mathsf{ST},A',\Phi}(\lambda) \le \Pr[\mathrm{H}^A(\lambda)]$ 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 $\mathrm{Retrieve}((C,\sigma), i)$ and halts with output $\perp$. The flag win is set to true with at least the probability that A' wins its game. Now we proceed to upper bound $\Pr[\mathrm{H}^A(\lambda)]$.

```
$\underline{\text{MAIN } \mathrm{H}^A(\lambda)}$
$Q \leftarrow \emptyset$ ; $\mathsf{win} \leftarrow \mathsf{false}$ ; $(fp, ap) \leftarrow_{\$} \mathsf{DS.Pg}(1^\lambda)$ ; $pp \leftarrow (fp, ap)$
$\perp \leftarrow_{\$} A^{\textsc{MkKey},\textsc{Store},\textsc{Retrieve}}(1^\lambda, pp)$
Ret $\mathsf{win}$

$\underline{\textsc{MkKey}(1^n)}$
For $i = 1, \ldots, n$ do $(\mathbf{sk}[i], \mathbf{pk}[i]) \leftarrow_{\$} \mathsf{DS.Kg}(1^\lambda, pp)$
Ret $\mathbf{pk}$

$\underline{\textsc{Store}(\phi, i)}$
If not $(1 \le i \le n)$ then Ret $\perp$
$M \leftarrow \Phi(1^\lambda, \phi, \mathbf{sk})$ ; $C \leftarrow_{\$} \mathsf{PKE.Enc}(1^\lambda, fp, \mathbf{pk}[i], M)$
$\sigma \leftarrow_{\$} \mathsf{DS.Sig}(1^\lambda, pp, \mathbf{sk}[i], C)$ ; $Q \leftarrow Q \cup \{((C, \sigma), i)\}$
Ret $(C, \sigma)$

$\underline{\textsc{Retrieve}((C, \sigma), i)}$
If not $(1 \le i \le n)$ then Ret $\perp$
If $((C, \sigma), i) \in Q)$ then Ret $\perp$
If $\mathsf{DS.Ver}(1^\lambda, pp, \mathbf{pk}[i], C, \sigma)$ then $\mathsf{win} \leftarrow \mathsf{true}$ else Ret $\perp$
$M \leftarrow \mathsf{PKE.Dec}(1^\lambda, fp, \mathbf{sk}[i], C)$
Ret $M$
```

Figure 14: **Game defining alternate form of SUF for the proof of part (2) of Theorem 6.1.**

Let $q(\cdot)$ be a polynomial such that the number of RETRIEVE queries of A in game $\mathrm{H}^A(\lambda)$ 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

$$\Pr[\mathrm{H}^A(\lambda)] \le \mathbf{Adv}^{\mathrm{sim}}_{\mathsf{DS},A_1}(\lambda) + \mathbf{Adv}^{\mathrm{kdm}}_{\mathsf{PKE},D,\Phi}(\lambda) + \nu(\lambda) + q(\lambda) \cdot \mathbf{Adv}^{\mathrm{ext}}_{\mathsf{DS},A_2}(\lambda)$$

for all $\lambda \in \mathbb{N}$, from which part (2) of the theorem follows.

The proof uses the games in Figure 15. We build A_1, A_2, D and ν such that for all $\lambda \in \mathbb{N}$ we have

$$\Pr[\mathrm{H}^A(\lambda)] - \Pr[\mathrm{G}_0^A(\lambda)] \le \mathbf{Adv}^{\mathrm{sim}}_{\mathsf{DS},A_1}(\lambda) \tag{15}$$

$$\Pr[\mathrm{G}_1^A(\lambda) \wedge \overline{\mathsf{bad}}] \le \mathbf{Adv}^{\mathrm{kdm}}_{\mathsf{PKE},D,\Phi}(\lambda) \tag{16}$$

$$\Pr[\mathrm{G}_2^A(\lambda) \text{ sets } \mathsf{bad}] \le \nu(\lambda) + q(\lambda) \cdot \mathbf{Adv}^{\mathrm{ext}}_{\mathsf{DS},A_2}(\lambda)\ . \tag{17}$$

The notation in Equation (16) means that we are considering the event that the game returns true and also bad is not set. Now games $\mathrm{G}_0, \mathrm{G}_1$ are identical until bad so a variant of the Fundamental Lemma of Game-Playing [17] says that $\Pr[\mathrm{G}_0^A(\lambda) \wedge \overline{\mathsf{bad}}] = \Pr[\mathrm{G}_1^A(\lambda) \wedge \overline{\mathsf{bad}}]$. We also observe that $\Pr[\mathrm{G}_0^A(\lambda) \text{ sets } \mathsf{bad}] \le \Pr[\mathrm{G}_2^A(\lambda) \text{ sets } \mathsf{bad}]$. (In both games, decryption in RETRIEVE is always done correctly.) Combining this with the above, for all $\lambda \in \mathbb{N}$ we have:

$$\begin{aligned}
\mathbf{Adv}^{\mathrm{suf}}_{\mathsf{ST},A',\Phi}(\lambda) &\le \Pr[\mathrm{H}^A(\lambda)] \\
&= \Pr[\mathrm{H}^A(\lambda)] - \Pr[\mathrm{G}_0^A(\lambda)] + \Pr[\mathrm{G}_0^A(\lambda)] \\
&\le \Pr[\mathrm{H}^A(\lambda)] - \Pr[\mathrm{G}_0^A(\lambda)] + \Pr[\mathrm{G}_0^A(\lambda) \wedge \overline{\mathsf{bad}}] + \Pr[\mathrm{G}_0^A(\lambda) \text{ sets } \mathsf{bad}] \\
&\le \Pr[\mathrm{H}^A(\lambda)] - \Pr[\mathrm{G}_0^A(\lambda)] + \Pr[\mathrm{G}_1^A(\lambda) \wedge \overline{\mathsf{bad}}] + \Pr[\mathrm{G}_0^A(\lambda) \text{ sets } \mathsf{bad}] \\
&\le \Pr[\mathrm{H}^A(\lambda)] - \Pr[\mathrm{G}_0^A(\lambda)] + \Pr[\mathrm{G}_1^A(\lambda) \wedge \overline{\mathsf{bad}}] + \Pr[\mathrm{G}_2^A(\lambda) \text{ sets } \mathsf{bad}] \\
&\le \mathbf{Adv}^{\mathrm{sim}}_{\mathsf{DS},A_1}(\lambda) + \mathbf{Adv}^{\mathrm{kdm}}_{\mathsf{PKE},D}(\lambda) + \nu(\lambda) + q(\lambda) \cdot \mathbf{Adv}^{\mathrm{ext}}_{\mathsf{DS},A_2}(\lambda)
\end{aligned}$$

```
main $\boxed{G_0^A(\lambda)}$ / $G_1^A(\lambda)$ / $G_2^A(\lambda)$
$Q \leftarrow \emptyset$ ; $Q' \leftarrow \emptyset$ ; $\mathsf{win} \leftarrow \mathsf{false}$
$fp \leftarrow\$ \mathsf{F.Pg}(1^\lambda)$ ; $(ap, std, xtd) \leftarrow\$ \mathsf{DS.SimPg}(1^\lambda)$ ; $pp \leftarrow (fp, ap)$
$\perp \leftarrow\$ A^{\text{MkKey,Store,Retrieve}}(1^\lambda, pp)$
Ret $\mathsf{win}$

MkKey$(1^n)$    // $G_0^A(\lambda)$ / $G_1^A(\lambda)$ / $G_2^A(\lambda)$
For $i = 1, \ldots, n$ do $(\mathbf{sk}[i], \mathbf{pk}[i]) \leftarrow\$ \mathsf{DS.Kg}(1^\lambda, pp)$
Ret $\mathbf{pk}$

proc Store$(\phi)$    // $G_0^A(\lambda)$ / $G_1^A(\lambda)$ / $G_2^A(\lambda)$
If not $(1 \le i \le n)$ then return $\perp$
$M \leftarrow \Phi(1^\lambda, \phi, \mathbf{sk})$ ; $C \leftarrow\$ \mathsf{PKE.Enc}(1^\lambda, fp, \mathbf{pk}[i], M)$
$\sigma \leftarrow\$ \mathsf{DS.SimSig}(1^\lambda, pp, std, \mathbf{pk}[i], C)$ ; $Q \leftarrow Q \cup \{((C, \sigma), i)\}$ ; $Q' \leftarrow Q' \cup \{(\mathbf{pk}[i], C, \sigma)\}$
Ret $(C, \sigma)$

proc Retrieve$((C, \sigma), i)$    // $\boxed{G_0^A(\lambda)}$ / $G_1^A(\lambda)$
If not $(1 \le i \le n)$ then return $\perp$
If $(((C, \sigma), i) \in Q)$ then Ret $\perp$
If $\mathsf{DS.Ver}(1^\lambda, pp, \mathbf{pk}[i], C, \sigma)$ then $\mathsf{win} \leftarrow \mathsf{true}$ else Ret $\perp$
$sk' \leftarrow\$ \mathsf{DS.Ext}(1^\lambda, pp, xtd, \mathbf{pk}[i], C, \sigma)$ ; $pk' \leftarrow \mathsf{F.Ev}(1^\lambda, fp, sk')$
$M \leftarrow \mathsf{PKE.Dec}(1^\lambda, fp, sk', C)$
If $(pk' \ne \mathbf{pk}[i])$ then $\mathsf{bad} \leftarrow \mathsf{true}$ ; $\boxed{M \leftarrow \mathsf{PKE.Dec}(1^\lambda, fp, \mathbf{sk}[i], C)}$
Ret $M$

proc Retrieve$((C, \sigma), i)$    // $G_2^A(\lambda)$
If not $(1 \le i \le n)$ then return $\perp$
If $(((C, \sigma), i) \in Q)$ then Ret $\perp$
If $\mathsf{DS.Ver}(1^\lambda, pp, \mathbf{pk}[i], C, \sigma)$ then $\mathsf{win} \leftarrow \mathsf{true}$ else Ret $\perp$
$M \leftarrow \mathsf{PKE.Dec}(1^\lambda, fp, \mathbf{sk}[i], C)$
If $(\mathbf{pk}[i], C, \sigma) \in Q'$ then $\mathsf{bad} \leftarrow \mathsf{true}$ ; Ret $M$
$sk' \leftarrow\$ \mathsf{DS.Ext}(1^\lambda, pp, xtd, \mathbf{pk}[i], C, \sigma)$ ; $pk' \leftarrow \mathsf{F.Ev}(1^\lambda, fp, sk')$
If $(pk' \ne \mathbf{pk}[i])$ then $\mathsf{bad} \leftarrow \mathsf{true}$
Ret $M$
```

Figure 15: **Games used in the proof of part (2) of Theorem 6.1.** Game G_0 includes the boxed code and game G_1 does not.

as desired. We proceed to the constructions of A_1, A_2, D and ν.

Adversary A_1 behaves as follows:

```
$\underline{A_1^{\text{Sign}}(1^\lambda, pp)}$
$Q \leftarrow \emptyset$
$\bot \leftarrow_\$ A^{\text{MkKeySim},\text{StoreSim},\text{RetrieveSim}}(1^\lambda, pp)$
If $\mathsf{win}$ then $b' \leftarrow 1$ else $b' \leftarrow 0$
Ret $b'$

$\underline{\text{MkKeySim}(1^n)}$
For $i = 1, \ldots, n$ do
    $(\mathbf{sk}[i], \mathbf{pk}[i]) \leftarrow_\$ \mathsf{DS.Kg}(1^\lambda, pp)$
Ret $\mathbf{pk}$

$\underline{\text{StoreSim}(\phi, i)}$
If not $(1 \le i \le n)$ then Ret $\bot$
$M \leftarrow \Phi(1^\lambda, \phi, \mathbf{sk})$
$C \leftarrow_\$ \mathsf{PKE.Enc}(1^\lambda, fp, \mathbf{pk}[i], M)$
$\sigma \leftarrow_\$ \text{Sign}(\mathbf{sk}[i], C)$ ; $Q \leftarrow Q \cup \{((C, \sigma), i)\}$
Ret $(C, \sigma)$

$\underline{\text{RetrieveSim}((C, \sigma), i)}$
If not $(1 \le i \le n)$ then Ret $\bot$
If $((C, \sigma), i) \in Q$ then Ret $\bot$
If $\mathsf{DS.Ver}(1^\lambda, pp, \mathbf{pk}[i], C, \sigma)$ then $\mathsf{win} \leftarrow \mathsf{true}$ ;
Else Ret $\bot$
$M \leftarrow \mathsf{PKE.Dec}(1^\lambda, fp, \mathbf{sk}[i], C)$
Ret $M$
```

When the challenge bit b in game SIM is 1, adversary A_1 simulates H, and when $b = 0$ it simulates G_0. Note that in the latter, decryptions done by Retrieve are always correct, so A_1 does not need to invoke the extractor. (Indeed, it could not, since it does not have an extraction trapdoor.) This establishes Equation (15).

Adversary D behaves as follows:

```
$\underline{D^{\text{MkKey},\text{Enc}}(1^\lambda, fp)}$
$Q \leftarrow \emptyset$ ; $sk^* \leftarrow \bot$ ; $j \leftarrow \bot$
$(ap, std, xtd) \leftarrow_\$ \mathsf{DS.SimPg}(1^\lambda)$
$pp \leftarrow (fp, ap)$
$\bot \leftarrow_\$ A^{\text{MkKey},\text{StoreSim},\text{RetrieveSim}}(1^\lambda, pp)$
If $(sk^*, j) = (\bot, \bot)$ then Ret 0
$M_1 \leftarrow 1^\lambda$ ; $C^* \leftarrow_\$ \text{Enc}(\langle M_1 \rangle, j)$
$M \leftarrow \mathsf{PKE.Dec}(1^\lambda, fp, sk^*, C^*)$
If $(M = M_1)$ then $b' \leftarrow 1$ else $b' \leftarrow 0$
Ret $b'$

$\underline{\text{StoreSim}(\phi, i)}$
If not $(1 \le i \le n)$ then Ret $\bot$
$C \leftarrow_\$ \text{Enc}(\phi, i)$
$\sigma \leftarrow_\$ \mathsf{DS.SimSig}(1^\lambda, pp, std, \mathbf{pk}[i], C)$
$Q \leftarrow Q \cup \{((C, \sigma), i)\}$
Ret $(C, \sigma)$

$\underline{\text{RetrieveSim}((C, \sigma), i)}$
If not $(1 \le i \le n)$ then Ret $\bot$
If $((C, \sigma), i) \in Q$ then Ret $\bot$
If (not $\mathsf{DS.Ver}(1^\lambda, pp, \mathbf{pk}[i], C, \sigma)$) then Ret $\bot$
$sk' \leftarrow_\$ \mathsf{DS.Ext}(1^\lambda, pp, xtd, \mathbf{pk}[i], C, \sigma)$
If $(\mathsf{F.Ev}(1^\lambda, fp, sk') = \mathbf{pk}[i])$ then $(sk^*, j) \leftarrow (sk', i)$
$M \leftarrow \mathsf{PKE.Dec}(1^\lambda, fp, sk', C)$
Ret $M$
```

Recall that $\langle M_1 \rangle$ denotes the constant function that always returns M_1. If win is set in G_1 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 (16).

Now we would like to show that G_0 (equivalently, G_1) 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 formalized via G_2, which breaks the setting of bad from G_0 into two parts corresponding to the two events of interest. To establish Equation (17), let E_1 be the event that bad is set by the line 5 "If" statement, and E_2 the event that bad is set by the line 7 "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 (17), we now build A_2 so that $\Pr[E_2 \wedge \overline{E}_1] \leq q(\lambda) \cdot \mathbf{Adv}^{\mathrm{ext}}_{\mathsf{DS},A_2}(\lambda)$:

```
$\underline{A_2^{\textsc{Sign}}(1^\lambda, pp)}$
$Q \leftarrow \emptyset\ ;\ j \leftarrow 0$
$\bot \leftarrow\$\ A^{\textsc{MkKeySim},\textsc{StoreSim},\textsc{RetrieveSim}}(1^\lambda, pp)$
$\ell \leftarrow\$\ \{1, \ldots, j\}$
Ret $(pk_\ell, C_\ell, \sigma_\ell)$

$\underline{\textsc{MkKeySim}(1^n)}$
For $i = 1, \ldots, n$ do
    $(\mathbf{sk}[i], \mathbf{pk}[i]) \leftarrow\$\ \mathsf{DS.Kg}(1^\lambda, pp)$
Ret $\mathbf{pk}$

$\underline{\textsc{StoreSim}(\phi, i)}$
If not $(1 \leq i \leq n)$ then Ret $\bot$
$M \leftarrow \Phi(1^\lambda, \phi, \mathbf{sk})$
$C \leftarrow\$\ \mathsf{PKE.Enc}(1^\lambda, fp, \mathbf{pk}[i], M)$
$\sigma \leftarrow\$\ \textsc{Sign}(\mathbf{sk}[i], C)\ ;\ Q \leftarrow Q \cup \{((C, \sigma), i)\}$
Ret $(C, \sigma)$

$\underline{\textsc{RetrieveSim}((C, \sigma), i)}$
If not $(1 \leq i \leq n)$ then Ret $\bot$
If $((C, \sigma), i) \in Q$ then Ret $\bot$
If (not $\mathsf{DS.Ver}(1^\lambda, pp, \mathbf{pk}[i], C, \sigma)$) then Ret $\bot$
$M \leftarrow \mathsf{PKE.Dec}(1^\lambda, fp, \mathbf{sk}[i], C)$
$j \leftarrow j + 1\ ;\ pk_j \leftarrow \mathbf{pk}[i]\ ;\ C_j \leftarrow C\ ;\ \sigma_j \leftarrow \sigma$
Ret $M$
```

Adversary A_2 always performs correct decryptions in responding to RETRIEVE queries, following G_2. If bad is set at line 7 but not at line 5, then there is some tuple on which the extractor would succeed. Since a tuple is guessed at random we have $\Pr[E_2 \wedge \overline{E}_1] \leq q(\lambda) \cdot \mathbf{Adv}^{\mathrm{ext}}_{\mathsf{DS},A_2}(\lambda)$ as desired. The importance of bad not being set at line 5 is that otherwise extraction is not required to succeed according to game EXT. ▮