# Parallel Signcryption with OAEP, PSS-R, and other Feistel Paddings

Yevgeniy Dodis[†] Michael J. Freedman[†] Shabsi Walfish[†]

March 13, 2003

**Abstract**

We propose to study *universal two-padding schemes* which transform a message $m$ into a pair $w\|s$ with the property that for any trapdoor permutation $f$, $f(w)\|s$ is a chosen-ciphertext-secure (IND-CCA2) encryption of $m$, while $w\|f^{-1}(s)$ is an existentially-unforgeable (sUF-CMA) signature of $m$. As the main application, we show that universal two-padding schemes lead to a new elegant composition method for *joint* signature and encryption, also referred to as *signcryption*. On a high level, to securely send a message $m$ to some recipient with key $f_{rcv}$, a sender with key $f_{snd}$ computes $f_{rcv}(w)\|f_{snd}^{-1}(s)$. The new method, which we call *Padding-based Parallel Signcryption* ($\mathcal{PbPS}$), is inspired by the "commit-then-encrypt-and-sign" ($\mathcal{CtE}\&\mathcal{S}$) composition method of An, Dodis and Rabin [2]. Like $\mathcal{PbPS}$, $\mathcal{CtE}\&\mathcal{S}$ first transforms a message $m$ into a pair $d\|c$, but then applies an IND-CCA2-secure encryption to $d$ and a sUF-CMA-secure signature to $c$. Two-padding schemes allow one to replace the above "strong" encryption and signature schemes by a mere trapdoor permutation (resp. its inverse) such as RSA. Additionally, $\mathcal{PbPS}$ supports flexible key management, provides tight security reductions, easily handles long messages and/or associated data, and is completely compatible with the PKCS#1 infrastructure.

We give an extremely general construction of universal two-padding schemes, which essentially shows that applying one round of the Feistel Transform to a pair $\langle d, c\rangle$ sufficient for $\mathcal{CtE}\&\mathcal{S}$ (roughly, a commitment/decommitment pair with few special properties), we get a pair $\langle w = c,\ s = H(c) \oplus d\rangle$ sufficient for $\mathcal{PbPS}$. More surprisingly, we notice that all popular padding schemes with message recovery used for plain signature or encryption, such as OAEP, OAEP+, PSS-R, and "scramble all, encrypt small" [21], actually consist of *two* natural components $w$ and $s$. Moreover, the last step of computing $w$ and $s$ always consists of a Feistel Transform applied to some pair $d$ and $c$. Remarkably, we show that *all* these pairs $\langle d, c\rangle$ satisfy the general conditions of our universal two-padding construction. As a result, not only do we find a natural generalization of all conventional padding schemes such as OAEP and PSS-R, but we show that any such padding scheme defines a secure two-padding scheme for joint signature and encryption.

Of independent interest, we also define a new "hybrid" between PSS-R and OAEP, which we call *Probabilistic Signature-Encryption Padding* (PSEP). This two-padding allows us to achieve optimal message bandwidth for signcryption using $\mathcal{PbPS}$.

**Keywords:** Universal padding schemes, signcryption, joint signature and encryption, authenticated encryption, Feistel Transform, OAEP, PSS-R, extractable commitment.

[†]{dodis,mfreed,walfish}@cs.nyu.edu. Department of Computer Science, New York University.

# Contents

# 1 Introduction

SIGNCRYPTION. Until recently, the two main building-blocks of modern public-key cryptography — encryption and signature schemes — have been considered as *distinct* entities that may be *composed* in various ways to ensure message privacy and authentication. From a design and analysis standpoint, this evolution makes sense, as encryption and signatures serve fundamentally different purposes. In practice, however, there are increasingly fewer applications that do not use both primitives, whether one considers secure e-mail or the key-establishment protocols for SSL or SSH.

In the past few years, research in the symmetric key setting has introduced *authenticated encryption* [5, 19, 23] to combine both functionalities in a single primitive. Soon thereafter, a number of authenticated-encryption schemes were proposed and other related investigations followed [26, 1, 22, 32, 31, 4, 11]. These results produced a variety of practical and efficient implementations. As importantly, they established authenticated encryption as a new *cryptographic primitive* which can be used to design simpler higher-level protocols.

More recent research has extended authentication encryption to the public-key setting, which is also the setting of this paper. We refer to this notion of a "joint signature and encryption" primitive as *signcryption*, following the terminology of [38]. While several papers [38, 39, 28, 20] offered security arguments about various signcryption schemes, the first formal investigations appeared only recently [3, 2]. Both works define signcryption as a multi-user primitive which simultaneously satisfies chosen ciphertext security for privacy and existential unforgeability for authenticity.[1] In terms of constructions, Baek *et al.* [3] showed that the original "discrete log-based" proposal of Zheng [38] indeed can be proven secure in the random oracle model under the so called Gap Diffie-Hellman assumption. Zheng's signcryption scheme is quite elegant and efficient, but has the disadvantage that all parties must agree on the same public parameters, such as the common discrete log group. Thus, for example, all users must uniformly agree on the security parameter and have some trusted party perform system initialization. Also, if one party wants to use a different security parameter or a different signcryption scheme, this party has to convince all other parties to change their public keys, or he will no longer be able to communicate with them. Finally, the security of [3] is based on a specific, non-standard assumption. In contrast, An, Dodis, and Rabin [2] formally examined generic composition methods of building signcryption from *any* secure signature and encryption scheme. In addition to the sequential compositions such as "encrypt-then-sign" ($\mathcal{E}t\mathcal{S}$) and "sign-then-encrypt" ($\mathcal{S}t\mathcal{E}$), this work also introduced a novel construction — "commit-then-encrypt-and-sign" ($\mathcal{C}t\mathcal{E}\&\mathcal{S}$) — that allows encryption and authentication to be performed in parallel. All these composition paradigms are very general and give rise to a large variety of signcryption schemes. Additionally, users can easily change their public keys or their favorite signature/encryption scheme, and still be able to seamlessly communicate with other users.[2] However, these generic schemes suffer from poor efficiency. Indeed, they all utilize relatively-expensive encryption and signature schemes which by *themselves* must already be IND-CCA2 and sUF-CMA secure.

OUR GOAL. The main motivation of this work is to design a class of signcryption schemes satisfying the following desirable properties. (1) *Key management is simple and flexible*. In particular, each user chooses its public/secret key on its own and has freedom in the type/length of the key chosen. Also, users can easily change/create their keys "on the fly" and still be able to communicate with others users, provided that they circulate the new key. (2) *Signcryption/de-signcryption are "close" to current standards for plain signature and encryption, i.e.*, PKCS #1 [33]. In particular, the signcryption/de-signcryption procedure should be somewhat similar to popular efficient signature/encryption schemes, such as PSS-R [7] or OAEP [6], as few code changes would therefore be needed to support signcryption with the existing infrastructure. (3) Related to the above, *users should easily be able to use their signcryption keys for plain signature/encryption functionality*.

[1] We will only use the *strongest* variants of these notions, which we will denote IND-CCA2 and sUF-CMA.

[2] That is, user $S$ uses $S$'s signature scheme and $R$'s encryption scheme when talking to user $R$.

(4) *Schemes must be simple and efficient.* For example, they must be faster than using a generic composition of strong signature and encryption. (5) Despite this, *schemes should be general enough to allow many instantiations*. (6) Last, but certainly not least, *schemes should be provably secure* under well-established cryptographic assumptions.

OUR METHOD. We propose the following high-level method to achieve all of the above properties based on any family $\mathcal{F}$ of trapdoor permutations. Each player $U$ independently picks a trapdoor permutation $f_U \in \mathcal{F}$ (together with its trapdoor, denoted $f_U^{-1}$) and publishes $f_U$ as its public key. To signcrypt a message $m$ from user $S$ to user $R$, $S$ first preprocesses $m$ into a pair of strings $(w, s)$, using what we call a *universal two-padding scheme*, whose constructions and security properties will be determined later. Then, $S$ transmits $f_R(w)\|f_S^{-1}(s)$ to $R$. Upon receiving ciphertext $\psi\|\sigma$, $R$ computes $w = f_R^{-1}(\psi)$, $s = f_S(\sigma)$, and $R$ recovers $m$ from $w$ and $s$ (possibly performing some "consistency check" before outputting $m$; see later). We call this method *Padding-based Parallel Signcryption* ($\mathcal{PbPS}$).[3]

We believe $\mathcal{PbPS}$ naturally satisfies all of the above properties (1)-(6). For example, user $U$ independently picks his key $f_U$ and uses the same $f_U$ for both sending and receiving data. Moreover, the specific two-padding schemes we construct are extremely fast and very flexible in accommodating arbitrary domains for $f_S$ and $f_R$, which allows users to use different families and change their keys easily. In fact, we show that popular padding schemes like OAEP, PSS-R and many others — ordinarily used for plain signature or encryption — can be used for signcryption purposes too, when viewed as *two-paddings*! Furthermore, we provide a simple, general way to construct and verify the security of universal two-padding schemes. The resulting signcryption is provably secure in the strongest sense (in the random oracle model), assuming the mere one-wayness of the underlying trapdoor permutation. Moreover, we show that the security reduction is tight for a large class of trapdoor permutations, including all the known ones such as RSA, Rabin, and Paillier. Additionally, the schemes easily achieve *non-repudiation*, since $R$ can extract a regular, publicly verifiable signature $w\|f_S^{-1}(s)$ of $S$ from the ciphertext (see below). Finally, the more expensive "encrypting" and "signing" operations (using $f_R$ and $f_S^{-1}$) are indeed performed *in parallel*.

To summarize, we believe that $\mathcal{PbPS}$ is a very efficient, yet general method, for building a robust, flexible, and provably-secure signcryption infrastructure.

TWO-PADDING SCHEMES: OUR RESULTS. The soundness of our suggested $\mathcal{PbPS}$ paradigm for signcryption crucially depends on the properties of *universal two-padding schemes*, a new notion we introduce. Syntactically, such schemes (probabilistically) transform a message $m$ into a *pair* $w\|s$, from which $m$ can be later recovered. In terms of security, we require that for any trapdoor permutation $f$, $f(w)\|s$ is a chosen-ciphertext-secure (IND-CCA2) encryption of $m$, while $w\|f^{-1}(s)$ is an existentially-unforgeable (sUF-CMA) signature of $m$. The *universality* property additionally requires that the above *induced* signature and encryption schemes remain secure even when used with the *same* key $f$.

First, we formally argue that universal two-paddings — defined *entirely* using plain signature and encryption properties — are indeed sufficient for $\mathcal{PbPS}$. This result actually requires some work, as signcryption has to be defined in the multi-user setting to prevent "identity fraud" [2]. In particular, the naive signcryption candidate $f_R(w)\|f_S^{-1}(s)$ (informally stated earlier for simplicity) will *not* be secure, unless we ensure that $w$ and $s$ also non-trivially depend on the public keys of $S$ and $R$. Luckily, we found several simple and efficient ways to achieve this "binding" *at minimal or no extra cost*, formally justifying our initial claim.

Second, we give a simple and very general construction of universal two-paddings in the random oracle model. Our starting point was the observation that all popular padding schemes with message recovery currently used for ordinary signature or encryption, such as OAEP [6], OAEP+ [36], PSS-R [7], and "scramble all, encrypt small" [21] (in the future denoted SAP) actually consist of *two* natural components $w$ and $s$, which

[3] As stated, it applies only to relatively short messages $m$. However, we later efficiently extend it to support long messages.

is consistent with our two-padding syntax. Moreover, the last step of computing $w$ and $s$ always consists of a Feistel Transform applied to some pair $d$ and $c$ (this step uses the random oracle $H$). This led us to examine which general properties on $d$ and $c$ suffice to ensure that $\langle w = c,\ s = H(c) \oplus d \rangle$ form a universal two-padding scheme. Quite interestingly, we found that all one needs is that $\langle d, c \rangle$ form a *commitment scheme* with a special property, called *extractability* (see [10]). We formally define it later, but observe that extractable commitments are extremely easy to construct in the random oracle model, which we use anyway in the subsequent Feistel Transform. Indeed, we give a number of such simple and efficient constructions, which in turn gives many examples of provably-secure two-padding schemes. Moreover, our security reductions from the corresponding two-padding schemes to the problem inverting $f$ are *tight* for a large class of trapdoor permutations $f$ (defined later) which includes all currently known examples, such as RSA, Rabin, and Paillier. This makes the $\mathcal{PbPS}$ paradigm very attractive in terms of exact security.

Even more remarkable, however, is that all the aforementioned padding schemes — OAEP, OAEP+, PSS-R, SAP— become *special cases of our general construction* when viewed as two-paddings! As a result, not only do we find a natural generalization of all conventional padding schemes, but we show that any such padding scheme defines a secure two-padding scheme which can then be used for signcryption. Of independent interest, we will also define a new "hybrid" between PSS-R and OAEP, which we call *Probabilistic Signature-Encryption Padding* (PSEP). This two-padding will allow us to achieve optimal message bandwidth for signcryption using $\mathcal{PbPS}$.

RELATION TO PREVIOUS WORK. While padding schemes are very popular in the design of ordinary encryption and signature schemes (e.g. [6, 7, 36, 15]), the most relevant previous works are those of [9, 2, 27].

COMPARING WITH [9]. Our universal two-padding schemes are similar in spirit to "universal padding" schemes defined by Coron *et al.* [9], which we refer to as universal *one-paddings*. To explain this name, such one-paddings transform $m$ into a *single* string $\pi$ such that $f(\pi)$ is a secure encryption and $f^{-1}(\pi)$ is a secure signature. Additionally, [9] requires that users can use the same trapdoor permutation $f$ for both signing and encrypting. We now compare one- and two-paddings. Application-wise, one-paddings are conceptually used for plain signature and encryption *separately*; *i.e.*, the user can either sign or encrypt with the same key. In this setting, reusing the key for signing/decrypting is much more important than reusing the padding scheme. In fact, the latter property is really of marginal importance given the simplicity of current padding schemes.[4] In contrast, our motivation for two-padding scheme comes from *joint* signature and encryption; *i.e.*, the user can either signcrypt or de-signcrypt, and even with a single key for both. In this setting, reusing the two-padding is much more crucial: The whole application to parallel signcryption would not even *make sense* unless the same $w$ and $s$ can be used for encrypting and signing *simultaneously*.

On the other hand, from a technical perspective, every non-trivial partition of a one-padding into two parts *is* a secure two-padding — by considering trapdoor permutations of the form $f'(w\|s) = f(w)\|s$ and $f''(w\|s) = w\|f(s)$ — while the converse is easily seen to be false. Of course, prior to this work, *no* universal one-padding schemes were known, since [9] only constructs one specific to RSA rather than *any* $f$. Subsequent to our work, several constructions of one-paddings were found [12, 25]. However, the construction of [25] is a special case of a more general construction of [12], while the latter critically builds upon this current work. Needless to say, these one-paddings are more complicated than the two-paddings we construct, and we actually *do not need* these extra complications for our signcryption application. Finally, we remark that a very special case of our result — PSS-R is a secure two-padding — can be indirectly derived from previous work of [7, 9]: [7] can be seen to imply the signing part, while the "partial one-wayness" result of [9] is general enough to imply the encryption part. Of course, our construction is much more general, gives many more two-paddings, and the

[4] We also remark that reusing the key without reusing the padding already followed from the prior work of Haber and Pinkas [18], who show that OAEP+ [36] and PSS-R [7] can reuse the same key.

whole signcryption application was not previously considered.

COMPARING WITH [2]. The parallel "commit-then-encrypt-and-sign" ($\mathcal{CtE}\&\mathcal{S}$) paradigm of [2] for building signcryption first applies any commitment scheme to transform $m$ into a pair $\langle d, c\rangle$, and then encrypts $d$ and signs $c$, using a IND-CCA2-secure encryption and sUF-CMA-secure signature, respectively. Two-paddings can be viewed as allowing us to replace the above "strong" encryption and signature schemes by a mere trapdoor permutation (resp. its inverse) such as RSA. In fact, our Feistel-based two-padding construction essentially says that applying one round of the Feistel Transform to a pair $\langle d, c\rangle$ sufficient for $\mathcal{CtE}\&\mathcal{S}$, we get a pair $\langle w = c,\ s = H(c) \oplus d\rangle$ sufficient for $\mathcal{PbPS}$!

Of course, another natural question is to compare the generic composition paradigm of [2] with the $\mathcal{PbPS}$ approach in the random oracle model. For example, we could use a padding-based IND-CCA2-secure encryption, such as OAEP+ [36], and a padding-based sUF-CMA-secure signature, such as PSS or PSS-R [7]. While this is indeed a possibility, the resulting signcryption scheme is considerably more awkward and less efficient (in all respects!) than the optimized $\mathcal{PbPS}$ approach used with any of the simple two-padding schemes we construct. For example, using $\mathcal{CtE}\&\mathcal{S}$ with OAEP+, PSS-R, and any random-oracle based commitment scheme, we essentially have to pad the message twice, which only allows us to signcrypt significantly shorter messages, and results in a poorer exact security. The sequential approaches have similar disadvantages, while additionally losing the parallelism.

COMPARING WITH [27]. This recent work suggests to use the PSS-R padding for sequential signcryption with RSA. Namely, to transmit $RSA_R(RSA_S^{-1}(\pi))$, where $\pi$ is the result of PSS-R applied to the message $m$, and $RSA_U$ is the RSA key of user $U$. This approach has several disadvantages as compared to the $\mathcal{PbPS}$ approach. (1) While the approach syntactically makes sense for general $f$, using PSS-R effectively restricts its use to RSA [9]. On the other hand, $\mathcal{PbPS}$ works for general $f$ with a wide variety of padding schemes. (2) The exact security of encryption is very poor, while $\mathcal{PbPS}$ gives tight security reductions. (3) The scheme can be proven secure only in the so-called two-user setting [2], and is provably insecure in a more realistic multi-user setting for signcryption. (4) While the above problem could potentially be fixed by somehow "binding" the message with the users' public keys, a more serious problem is that [27] use a relatively weak notion of so-called Outsider security [2] for privacy. In contrast, $\mathcal{PbPS}$ uses a much stronger notion of Insider security [2] for both privacy and authenticity. We see no obvious way how to overcome this problem in the construction of [27]. (5) The scheme of [27] is sequential, while $\mathcal{PbPS}$ is parallel. (6) Using a sequential composition with RSA creates syntactic problems of ensuring that the domain sizes for $RSA_S$ and $RSA_R$ "match up", which requires special ad-hoc care. In particular, the suggested scheme is not flexible to support RSA keys of different sizes, while $\mathcal{PbPS}$ has no such problem.

On a positive note, if all users in the system have RSA keys of size $k$, the scheme of [27] allows a user to signcrypt (very short) messages with a ciphertext of length $k$. On the other hand, the minimal length of the ciphertext with $\mathcal{PbPS}$ would be $2k$. We believe that this disadvantage is minor in light of (1)-(6), especially since it is relevant only for very short messages. Moreover, using the scheme of [27], one can only signcrypt messages of length significantly less than $k/2$, while $\mathcal{PbPS}$ with an appropriate two-padding scheme allows a user to signcrypt messages of length close to $2k$.

EXTENSIONS. We extend the basic $\mathcal{PbPS}$ approach in two important ways. First, it can effortlessly support *associated data* [31], allowing one to "bind" a public label to a message when signcrypting it. This capability has many nice applications, including allowing us to trivially bind the message to the public keys of $S$ and $R$, thus solving the aforementioned "multi-user" problem for signcryption. Second, using the recent work of Dodis and An [11], we efficiently extend our method to signcrypt arbitrarily long messages; namely, to build a full-fledged, practical signcryption scheme of arbitrary messages (that also supports associated data).

## 2 Definitions

In this section, we start by quickly reviewing some common cryptographic definitions; Appendix A provides more formal security definitions of such schemes. Second, we introduce the notion of a two-padding. We put off the discussion of signcryption until Section 5.

### 2.1 Encryption, Signatures, and Trapdoor and Claw-Free Permutations

ENCRYPTION. A public-key encryption scheme consists of the algorithms $(\mathsf{Enc\text{-}Gen}, \mathsf{Enc}, \mathsf{Dec})$. $\mathsf{Enc\text{-}Gen}(1^\lambda)$ generates the public/private key-pair $(\mathsf{EK}, \mathsf{DK})$, with a security parameter $\lambda$. Syntactically, we write the randomized encryption algorithm as $\psi \leftarrow \mathsf{Enc}_{\mathsf{EK}}(m)$, where $m$ is a message chosen from message space $\mathcal{M}$ and $\psi$ is the associated ciphertext. We express the behavior of the deterministic decryption algorithm as $\{m, \perp\} \leftarrow \mathsf{Dec}_{\mathsf{DK}}(\psi)$, where Dec outputs $m$ or $\perp$ if $\psi$ is invalid. In this paper, we only consider the strongest notion of security: IND-CCA2 security. This property means that the encryption scheme provides *indistinguishability* of ciphertexts (IND) [16] under *adaptive chosen-ciphertext attacks* (CCA2) [30, 14]: No probabilistic poly-time (PPT) adversary can distinguish between the ciphertexts of two chosen messages, $m_0$ and $m_1$, given the corresponding public key EK and oracle access to Dec, with probability greater than $\varepsilon_{\mathsf{CCA2}}$, where $\varepsilon_{\mathsf{CCA2}}$ is negligible in $\lambda$.

SIGNATURES. A public-key signature scheme consists of the algorithms $(\mathsf{Sig\text{-}Gen}, \mathsf{Sig}, \mathsf{Ver})$. $\mathsf{Sig\text{-}Gen}(1^\lambda)$ generates the key-pair $(\mathsf{SK}, \mathsf{VK})$, where SK is the signing key kept private, and VK is the verification key made public. We write the randomized signature algorithm as $\sigma \leftarrow \mathsf{Sig}_{\mathsf{SK}}(m)$. As we assume that the signature scheme has message recovery, the deterministic verification algorithm can be expressed as $a \leftarrow \mathsf{Ver}_{\mathsf{VK}}(\sigma)$, where the answer $a \in \{\mathsf{succeed}, \mathsf{invalid}\}$, where invalid is again denoted by $\perp$. Correctness requires that $\mathsf{Ver}(\mathsf{Sig}(m)) \neq \perp$ for any $m \in \mathcal{M}$. We consider the strongest notion of signature security, *strong unforgeability* against a *chosen-message attack* (sUF-CMA) [17, 5]: Given VK and oracle access to Sig, no PPT adversary can forge a *new* signature $\sigma^*$ with probability greater than $\varepsilon_{\mathsf{CMA}}$, where $\varepsilon_{\mathsf{CMA}}$ is negligible in $\lambda$.

TRAPDOOR PERMUTATIONS. Informally, a family of trapdoor permutations (TDPs) is a family of permutations such that it is easy to randomly select a permutation $f$ and some "trapdoor" associated with $f$. Furthermore, $f$ is easy to compute and, given the trapdoor information, so is its inverse $f^{-1}$. However, without the trapdoor, $f$ is "hard" to invert: No PPT adversary $\mathcal{A}$, given some $y \leftarrow f(x)$, can find $x$ with probability greater than $\varepsilon_{\mathsf{TDP}}$, which is negligible in the security parameter $\lambda$ of the generation algorithm.

CLAW-FREE PERMUTATIONS. To improve the exact security of our constructions, we will also talk about a general class of TDPs — those induced by a family of *claw-free permutation* pairs [17], following the observation made by [13]. In this context, the generation algorithm outputs $(f, f^{-1}, g)$, where $g$ is another efficient permutation over the same domain as $f$. The task of the PPT adversary $\mathcal{B}$ now is to find a "claw" $(x, z)$, *i.e.*, $f(x) = g(z)$, which it succeeds at with probability $\varepsilon_{\mathsf{claw}}$, negligible in $\lambda$. It is trivial to see that omitting $g$ from the generation algorithm induces a TDP family with $\varepsilon_{\mathsf{TDP}} \leq \varepsilon_{\mathsf{claw}}$ ($\mathcal{B}$ calls $\mathcal{A}$ on random $g(z)$). On the other hand, all known TDP families, such as RSA, Rabin, and Paillier, are easily seen to be induced by some claw-free permutation families with $\varepsilon_{\mathsf{claw}} = \varepsilon_{\mathsf{TDP}}$. Thus, a tight reduction to "claw-freeness" of such families implies a tight reduction to inverting them. See [13] for more details.

### 2.2 Two-Paddings

SYNTAX. A two-padding scheme consists of the poly-time algorithms PAD and DePAD. The probabilistic algorithm PAD accepts input messages $m \in \mathcal{M}$ and produces a pair of outputs, denoted as $(w, s) \leftarrow \mathsf{PAD}(m)$.

The deterministic algorithm $\mathsf{DePAD}$ accepts input pairs of the same form $(w, s)$ and returns either message $m \in \mathcal{M}$ or $\perp$. Correctness requires that $\mathsf{DePAD}(\mathsf{PAD}(m)) = m$ for any $m \in \mathcal{M}$.

For syntactical convenience, we further define a pair of operations, with respect to any $\mathsf{TDP}$s $f$ and $f'$, as the following: $\psi \| s \leftarrow \mathsf{PadEnc}_f(m)$ and $w \| \sigma \leftarrow \mathsf{PadSig}_{f'}(m)$. $\mathsf{PadEnc}_f(m)$ first computes $(w, s) \leftarrow \mathsf{PAD}(m)$ and then outputs $\psi \| s = f(w) \| s$. Similarly, $\mathsf{PadSig}_{f'}(m)$ computes $(w, s) \leftarrow \mathsf{PAD}(m)$ and outputs $w \| \sigma = w \| f'^{-1}(s)$. The corresponding pair of operations $\mathsf{PadDec}_f(\psi \| s)$ and $\mathsf{PadVer}_{f'}(w \| \sigma)$ are defined in the natural way, both recovering the pair $(w, s)$ and outputting $\mathsf{DePAD}(w, s)$.

SECURITY. We call $\mathcal{PS} = (\mathsf{PAD}, \mathsf{DePAD})$ a $(t, \varepsilon_{\mathsf{CCA2}}, \varepsilon_{\mathsf{CMA}}, q_D, q_S)$-secure *two-padding* scheme if, for any $(T_f, \varepsilon_{\mathsf{TDP}})$-secure $\mathsf{TDP}$s $f$ and $f'$, the corresponding $\mathsf{PadEnc}_f$ is a $(t, \varepsilon_{\mathsf{CCA2}}, q_D)$-secure $\mathsf{IND\text{-}CCA2}$ encryption and $\mathsf{PadSig}_{f'}$ is a $(t, \varepsilon_{\mathsf{CMA}}, q_S)$-secure $\mathsf{sUF\text{-}CMA}$ signature.

We call $\mathcal{PS} = (\mathsf{PAD}, \mathsf{DePAD})$ a $(t, \varepsilon_{\mathsf{CCA2}}, \varepsilon_{\mathsf{CMA}}, q_D, q_S)$-secure *universal two-padding* scheme if, for any $\mathsf{TDP}$ $f$, the corresponding $\mathsf{PadEnc}_f$ and $\mathsf{PadSig}_f$ are simultaneously $(t, \varepsilon_{\mathsf{CCA2}}, q_D)$- and $(t, \varepsilon_{\mathsf{CCA2}}, q_D)$-secure, respectively, when a user reuses the same $f$ for both encryption and signature. Formally, the adversary has access to a $\mathsf{PadSig}_f$ oracle during the $\mathsf{IND\text{-}CCA2}$ attack game played against $\mathsf{PadEnc}_f$, and, similarly, the adversary has access to a $\mathsf{PadDec}_f$ oracle during the $\mathsf{sUF\text{-}CMA}$ attack game played against $\mathsf{PadSig}_f$.

### 2.3 Extractable Commitments

Our constructions for two-paddings will involve a specialized commitment scheme we call an "extractable" commitment. Extractable commitments have a syntax similar to standard commitment schemes, but with the additional property that there exists an extraction algorithm which a simulator can use to extract a unique decommitment from any valid commitment with high probability. This extraction algorithm immediately follows for most commitment schemes based on the random oracle model, but requires the existence of a trapdoor for commitment schemes which do not make use of the random oracle model. Note that this differs from what is commonly referred to as a "trapdoor commitment" [8] where the goal is to construct alternative decommitments (with different openings) for a given commitment.

SYNTAX. An extractable commitment scheme $\mathcal{C}$ consists of four algorithms $(\mathsf{Setup}, \mathsf{Commit}, \mathsf{Open}, \mathsf{Extract})$. The optional setup algorithm $\mathsf{Setup}(1^\lambda)$ outputs a public commitment key $\mathsf{CK}$ (possibly empty) and possibly a secret trapdoor $\mathsf{TK}$ used by the extraction algorithm $\mathsf{Extract}$. Given a message $m \in \mathcal{M}$ and some random coins $r$, $\mathsf{Commit}_{\mathsf{CK}}(m; r)$ outputs a pair $(c, d)$ where $c$ is $k_1$-bit string representing the commitment to $m$ and $d$ is the corresponding $k_2$-bit long decommitment. As a shorthand, we will write $(c, d) \leftarrow \mathsf{Commit}(m)$ and $c(m)$ to denote a commitment to message $m$. $\mathsf{Open}_{\mathsf{CK}}(c, d)$ outputs $m$ if $(c, d)$ is a valid commitment/decommitment pair for $m$, or $\perp$ otherwise. Correctness requires $\mathsf{Open}(\mathsf{Commit}(m)) = m$ for all $m \in \mathcal{M}$.

SECURITY. We require this commitment scheme to satisfy two security properties:

HIDING. No $\mathsf{PPT}$ adversary can distinguish the commitment of any messages of its choice from a $k_1$-bit random string: $c(m) \approx R$. Formally, for any $\mathsf{PPT}$ $\mathcal{A}$ running in two stages, find and guess, in time at most $t$,

$$\Pr\left[\mathcal{A}(c;\, \alpha, \mathsf{guess}) = 1 \,\middle|\, \begin{array}{c}(m, \alpha) \leftarrow \mathcal{A}(1^\lambda, \mathsf{find}),\\ (c, d) \leftarrow \mathsf{Commit}(m)\end{array}\right] - \Pr\left[\mathcal{A}(R;\, \alpha, \mathsf{guess}) = 1 \,\middle|\, \begin{array}{c}(m, \alpha) \leftarrow \mathcal{A}(1^\lambda, \mathsf{find}),\\ R \xleftarrow{R} \{0,1\}^{k_1}\end{array}\right] \leq \varepsilon_{\mathsf{hide}}$$

where $\varepsilon_{\mathsf{hide}}$ is negligible in the security parameter $\lambda$. Note that $\mathsf{Setup}(1^\lambda)$ is implicitly run in both experiments if necessary, and $\mathsf{CK}$ is given to $\mathcal{A}$. This property is a slightly stronger requirement than that of an ordinary commitment scheme which only requires $c(m_0) \approx c(m_1)$.

EXTRACTABILITY. There exists a deterministic poly-time algorithm $\mathsf{Extract}$ which can extract the "correct" decommitment from any valid commitment, given access to all random oracle transcripts (and the trapdoor $\mathsf{TK}$ output by $\mathsf{Setup}$ if necessary). Formally, for any $\mathsf{PPT}$ $\mathcal{A}$ running in time at most $t$,

$$\Pr[\mathsf{Extract}(c, \mathcal{T}) \neq d \wedge \mathsf{Open}(c, d) \neq \perp \mid (c, d) \leftarrow \mathcal{A}(1^\lambda)] \leq \varepsilon_{\mathsf{extract}}$$

where $\mathcal{T}$ is either a transcript of all random oracle queries or the trapdoor information $\mathsf{TK}$, and $\varepsilon_{\mathsf{extract}}$ is negligible in $\lambda$. The $\mathsf{Setup}$ operation is implicit when necessary, and $\mathsf{CK}$ is given to $\mathcal{A}$.

We say that a commitment scheme $\mathcal{C}$ is a $(t, \varepsilon_{\mathsf{hide}}, \varepsilon_{\mathsf{extract}})$-secure extractable commitment if it satisfies the above properties. The "standard" notion of a commitment requires a binding property, instead of our extractability property. We now show that a strong form of binding follows from the extractability property.

**Lemma 1 (Binding property of extractable commitments)** *Given* $\mathsf{CK}$*, it is computationally hard to produce* $(c, d, d')$ *such that* $(c, d)$ *and* $(c, d')$ *are valid commitment pairs and* $d \neq d'$*. Formally, for any* $\mathsf{PPT}$ $\mathcal{A}$ *running in time at most* $t$ *(where* $\mathsf{Setup}$ *is implicit and* $\mathsf{CK}$ *is given to* $\mathcal{A}$*),*

$$\Pr[\mathsf{Open}(c, d) \neq \perp \wedge \mathsf{Open}(c, d') \neq \perp \wedge d \neq d' \mid (c, d, d') \leftarrow \mathcal{A}(1^\lambda)] \stackrel{\text{def}}{\leq} \varepsilon_{\mathsf{bind}} \leq 2\varepsilon_{\mathsf{extract}}$$

When appropriate, we directly use $\varepsilon_{\mathsf{bind}}$ for conceptual clarity and because $\varepsilon_{\mathsf{bind}}$ may in fact be tighter than $2\varepsilon_{\mathsf{extract}}$. Note this is slightly stronger than the normal binding property of commitment schemes, where $(c, d')$ must not represent a valid commitment for a *different* message: We also disallow alternative decommitments of the same message.

**Proof:** Consider a reduction $\mathcal{B}$ against the extractability property of the commitment scheme as follows. $\mathcal{B}$ runs $\mathcal{A}$ and obtains $(c, d, d')$ if $\mathcal{A}$ succeeds. $\mathcal{B}$ then randomly outputs $(c, d)$ or $(c, d')$ with equal probability. Since $\mathsf{Extract}(c, \mathcal{T})$ is a deterministic value, it matches the output of $\mathcal{B}$ with probability at most $1/2$. In the event that it does not match, $\mathcal{B}$ has broken the extractability property. Since this must happens with probability at most $\varepsilon_{\mathsf{extract}}$, we find that $\mathcal{A}$ succeeds with probability at most $2\varepsilon_{\mathsf{extract}}$. □

We now state an additional useful property of $(t, \varepsilon_{\mathsf{hide}}, \varepsilon_{\mathsf{extract}})$-secure extractable commitments:

**Lemma 2** $\forall$ $\mathsf{PPT}$ $\mathcal{A}$ *running in time* $t$*,* $\Pr[\mathsf{Open}(c, d) \neq \perp \mid c \leftarrow A(1^k); d \stackrel{R}{\leftarrow} \{0,1\}^{k_2}] \leq \varepsilon_{\mathsf{extract}} + 2^{-k_2}$

**Proof:** Consider a reduction $\mathcal{B}$ against the extractability property of the commitment scheme as follows. $\mathcal{B}$ runs $\mathcal{A}$ and obtains $c \leftarrow A(1^k)$, chooses a $d$ uniformly at random, and returns $(c, d)$. The probability that $\mathcal{B}$ succeeds is at least the probability that $\mathcal{A}$ succeeds minus the probability that $d = \mathsf{Extract}(c, \mathcal{T})$. Since $d$ is chosen randomly, the probability that $d = \mathsf{Extract}(c, \mathcal{T})$ is $2^{-k_2}$. The lemma follows. □

# 3 Feistel Two-Padding

We now provide a generic construction for a class of provably secure two-padding schemes in the random oracle model based on a single round of the Feistel Transform.

**Definition 1 (Feistel Two-Padding)** *Let* $\mathcal{C} = (\mathsf{Setup}, \mathsf{Commit}, \mathsf{Open}, \mathsf{Extract})$ *be any* $(t, \varepsilon_{\mathsf{hide}}, \varepsilon_{\mathsf{extract}})$*-secure extractable commitment scheme and* $H : \{0,1\}^{k_1} \rightarrow \{0,1\}^{k_2}$ *be a random oracle. A* Feistel Two-Padding $\mathsf{PAD}_{\mathcal{C}}(m) \rightarrow (w, s)$ induced by $\mathcal{C}$ *is given by:*

$$\begin{aligned} (c, d) &\leftarrow \mathsf{Commit}(m) \\ w &\leftarrow c \\ s &\leftarrow H(w) \oplus d \end{aligned}$$

*Note that $(w, s)$ represents a Feistel Transform on input $(d, c)$ using $H$. The corresponding $\mathsf{DePAD}_{\mathcal{C}}$ algorithm computes $d = H(w) \oplus s$ and $c = w$, then returns $\mathsf{Open}(c, d)$.*

**Theorem 1** *Feistel two-padding is a universal two-padding. In terms of exact security: For any $(t, \varepsilon_{\mathsf{hide}}, \varepsilon_{\mathsf{extract}})$-secure extractable commitment $\mathcal{C}$ and any $(t, \varepsilon_{\mathsf{TDP}})$-secure TDP $f$, the Feistel two-padding scheme induced by $\mathcal{C}$ and $f$ is a $(t', \varepsilon_{\mathsf{CCA2}}, \varepsilon_{\mathsf{CMA}}, q_D, q_S, q_H)$-secure universal two-padding scheme, with*

$$\begin{aligned} \varepsilon_{\mathsf{CCA2}} &\leq \varepsilon_{\mathsf{TDP}} + 2\varepsilon_{\mathsf{hide}} + (q_D + 1)(2^{-k_2} + \varepsilon_{\mathsf{extract}} + \varepsilon_{\mathsf{bind}}) \\ \varepsilon_{\mathsf{CMA}} &\leq q_H \cdot \varepsilon_{\mathsf{TDP}} + q_S(2^{-k_1} + \varepsilon_{\mathsf{hide}}) + (q_D + 2)(2^{-k_2} + \varepsilon_{\mathsf{extract}} + \varepsilon_{\mathsf{bind}}) \\ t' &= t - \mathrm{O}((q_H + q_S) \cdot (T_f + T_{extract})) \end{aligned}$$

*where $q_H$ is the number of queries to $H$, $T_f$ is the running time of $f$, and $T_{extract}$ is the running time of* Extract.

*Furthermore, if $f$ is induced by a $(t, \varepsilon_{\mathsf{claw}})$-secure family claw-free permutations, then the signature reduction becomes tight:*

$$\varepsilon_{\mathsf{CMA}} \leq \varepsilon_{\mathsf{claw}} + q_S(2^{-k_1} + \varepsilon_{\mathsf{hide}}) + (q_D + 2)(2^{-k_2} + \varepsilon_{\mathsf{extract}} + \varepsilon_{\mathsf{bind}})$$

The proof of this theorem is given in Appendix B. We note that by the result of [13], a security loss of $\Omega(q_H)$ for our signature reduction with general TDPs is inevitable. Thus, the restriction to claw-free permutations is necessary to obtain a tight security reduction.

# 4 PSS-R, OAEP, OAEP+, SAP and other Feistel 2-Paddings

We now demonstrate that our Feistel two-padding construction generalizes nearly all common padding schemes currently used for plain encryption or signature. First, we observe that conventional padding schemes such as OAEP [6], OAEP+ [36], PSS-R [7], and SAP [21] naturally consist of *two* parts $w\|s$ and indeed utilize a round of the Feistel Transform on various pairs of $\langle d, c\rangle$ at their last step: $\langle w = c, s = H(w) \oplus d\rangle$. Thus, using Theorem 1 we only need to prove that the corresponding $\langle d, c\rangle$ satisfy the simple requirements of extractable commitments.

The basic arguments we present are quite standard in the random oracle model, so we will often only sketch the proofs in this section, omitting the obvious (but tedious) details for brevity. We start with a short discussion of OAEP and PSS-R, before generalizing both of these padding schemes to PSEP. Exact bounds on $\varepsilon_{\mathsf{hide}}, \varepsilon_{\mathsf{extract}}$ will be given later.

- **OAEP.** This padding is defined as $\left\langle w = (m\|0^{\hat{k}}) \oplus G''(r),\ s = H(w) \oplus r \right\rangle$, which is syntactically equivalent to $\langle w = (m \oplus G(r))\|G'(r),\ s = H(w) \oplus r\rangle$. This $\langle w, s\rangle$ pair is constructed by a one-round Feistel Transform on $\langle d = r,\ c = (m \oplus G(r))\|G'(r)\rangle$. We now argue that this $\langle d, c\rangle$ forms an extractable commitment. Recall $|c| = k_1$ and let $|m| = n < k_1$. (1) Hiding is true as $G''(r)$ is a perfect one-time-pad (OTP) unless $r$ is reused, which occurs with negligible probability. (2) Extractability is easily seen to be true by noticing that one can "extract" $r$ from $G'(r)$ (which is part of $c$) by simply going through the random oracle transcripts and seeing which one matches the last $\hat{k} = (k_1 - n)$ bits of $c$.

- **PSS-R.** This padding scheme has the form $\langle w = G'(m\|r),\ s = H(w) \oplus (m\|r)\rangle$, which yields the pair $\langle d = m\|r,\ c = G'(m\|r)\rangle$. Here $|m| = n < k_2$. We again argue that this is an extractable commitment. (1) Hiding is true since $r$ is random and thus $G'(m\|r)$ hides all information about $m$, unless the adversary queries $G'$ on $m\|r$, which happens with negligible probability. (2) Extractability is again easily seen by simply going through the transcripts of $G'$ and extracting the corresponding $m\|r$ which matches $c$.

- PSEP. In fact, we observe the surprising fact that the above two padding schemes can be commonly generalized into the following form. For any parameter $a \in [0, n]$ (where $|m| = n$), write $m = m_1 \| m_2$, where $|m_1| = a$ and $|m_2| = n - a$, and define

$$\begin{array}{rcl} w & \leftarrow & \Big(m_1 \oplus G(r)\Big) \parallel G'(m_2 \| r) \\ s & \leftarrow & H(w) \oplus (m_2 \| r) \end{array}$$

  Notice if we set $a = 0$, this scheme yields PSS-R; while if $a = n$, we exactly have OAEP. Accordingly, we call this "hybrid" scheme *Probabilistic Signature Encryption Padding*. In Appendix D, we argue the following bounds on PSEP.

**Lemma 3** *The commitment scheme* $\langle d = m_2 \| r,\ c = (m_1 \oplus G(r)) \| G'(m_2 \| r) \rangle$ *defining* PSEP *satisfies:*

$$\varepsilon_{\mathsf{hide}} \leq (q_G + q_{G'}) \cdot 2^{-(k_2 + a - n)}; \qquad \varepsilon_{\mathsf{extract}} \ \leq \ (q_{G'}^2 + 1) \cdot 2^{-(k_1 - a)}$$

*where* $q_G$ *and* $q_{G'}$ *are the number of oracle queries to* $G$ *and* $G'$ *made by the adversary.*

By playing with the parameter $a$, we now have greater flexibility in choosing the lengths of $w$ and $s$, or maximizing the length $n$ of our message $m$ when $|w| = k_1$ and $|s| = k_2$ are fixed. For example, consider the natural "balanced case" $k_1 = k_2 = k$. With both OAEP and PSS-R, we had to set $n < k$, while the total length $2k$ of our two-padding potentially allowed to make $n \approx 2k$. With the more general scheme, we can easily achieve this goal! Indeed, Lemma 3 implies that it is safe to set $|m_1| = k - 2\hat{k}$, $|m_2| = k - \hat{k}$, $|r| = |G'(\cdot)| = \hat{k}$, where $\hat{k}$ is just large enough to be a security parameter (*i.e.*, 150 bits is more than enough in practice). For example, when $k = 1024$, we can (conservatively) fit 1600-bit message inside our two-padding (of total length 2048), instead of about 700-800 bits allowed by plain PSS-R and OAEP. (Of course, for many applications, signcrypting short messages is sufficient. For example, Dodis and An [11] recently showed that one can easily build an arbitrary-length signcryption from one supporting only about 300-bit messages.)

- OAEP+. This padding is a similar but slightly more "conservative" form of OAEP, with $d = r$ and $c = (m \oplus G(r) \| G'(m \| r)$. The proof that above $\langle c, d \rangle$ form an extractable commitment a simple variation of the argument for OAEP, and is included in Appendix D. We remark, however, that the "extra" input $m$ to $G'$ used in OAEP+ is not actually necessary for our application (although it does provide a slightly tighter bound for $\varepsilon_{\mathsf{extract}}$). The original reason for which Shoup [36] proposed OAEP+ in place of OAEP was to provide security for encryption when a generic TDP $f$ is applied to *both* $w \| s$, instead of only to $w$. In fact, Fujisaki *et al*. [15] already showed that it is safe to use plain OAEP when $f$ is only applied to $w$. Our much more general framework gives yet another verification of this fact.

- SAP. This padding scheme can be viewed as the following. Write $m = m_1 \| m_2$, and set

$$\begin{array}{rcll} w & \leftarrow & G\Big(m_1 \| r \| G'(m_2)\Big) \oplus m_2 & \text{// pad } m_2 \text{ with 0's if } |m_2| < |G(\cdot)| \\ s & \leftarrow & H(w) \oplus \Big(m_1 \| r \| G'(m_2)\Big) & \end{array}$$

  Actually, this is a slight simplification of the "scramble all" padding scheme used in [21]. In the original version, $G'(\cdot)$ was more conservatively applied to $m_1 \| m_2 \| r$. We show that for our purposes, even the simpler version suffices, which we also show in Appendix D. (Of course, the original version can be easily shown secure as well.) Interestingly, if we set $|m_2| = |G'(\cdot)| = 0$ for SAP, we again get PSS-R! On the other hand, if we set $|m_1| = 0$, we get yet another, new extractable commitment scheme: $\langle d = r \| G'(m),\ c = G(d) \oplus m \rangle$.

In short, the approach of Theorem 1 allows us to derive both old and new provably-secure two-padding constructions, by only showing a few straightforward properties in $\langle d, c \rangle$!

# 5 Two-Padding as a Secure Signcryption

As observed by [3, 2], full-fledged signcryption must be defined in the *multi-party* setting, where issues with users' identities are addressed. In contrast, authenticated encryption in the symmetric setting only needs to consider a much simpler *two-party* setting. A similar two-party model could be used for signcryption too [2]. However, as transforming two-user signcryption to the multi-user setting is quite subtle in general (see [2]), we will right away consider the more challenging and generally required multi-user setting.

## 5.1 Definition of Signcryption

SYNTAX. A signcryption scheme consists of the algorithms $(\mathsf{Gen}, \mathsf{SigEnc}, \mathsf{VerDec})$. In the multi-party setting, the $\mathsf{Gen}(1^\lambda)$ algorithm for user $U$ generates the key-pair $(\mathsf{SDK}_U, \mathsf{VEK}_U)$, where $\lambda$ is the security parameter, $\mathsf{SDK}_U$ is the signing/decryption key that is kept private, and $\mathsf{VEK}_U$ is the verification/encryption key made public. Without loss of generality, we assume that $\mathsf{VEK}_U$ is determined from $\mathsf{SDK}_U$.

The randomized signcryption algorithm $\mathsf{SigEnc}$ for user $U$ implicitly takes as input the user's secret key $\mathsf{SDK}_U$, and explicitly takes as input the message $m \in \mathcal{M}$ and the identity of the recipient, in order to compute and output the signcryption $\Pi$. For simplicity, we consider this identity $\mathsf{ID}$ to be a public key $\mathsf{VEK}$, although $\mathsf{ID}$ could be of more complex form, provided that other users can easily obtain $\mathsf{VEK}$ from $\mathsf{ID}$. Thus, we write $\mathsf{SigEnc}_{\mathsf{SDK}_U}(m, \mathsf{ID}_R)$ as $\mathsf{SigEnc}_{\mathsf{SDK}_U}(m, \mathsf{VEK}_R)$, or simply $\mathsf{SigEnc}_U(m, \mathsf{VEK}_R)$.

Similarly, user $U$'s deterministic de-signcryption algorithm $\mathsf{VerDec}$ implicitly takes the user's private $\mathsf{SDK}_U$, and explicitly takes as input the signcryption $\tilde{\Pi}$ and the senders' identity. Again, we assume $\mathsf{ID}_S = \mathsf{VEK}_S$, and write $\mathsf{VerDec}_{\mathsf{SDK}_U}(\Pi, \mathsf{VEK}_S)$, or simply $\mathsf{VerDec}_U(\Pi, \mathsf{VEK}_S)$. The algorithm outputs some message $\tilde{m}$, or $\perp$ if the signcryption does not verify or decrypt successfully. Correctness ensures that for any users $S$ and $R$, $\mathsf{VerDec}_R(\mathsf{SigEnc}_S(m, \mathsf{VEK}_R), \mathsf{VEK}_S) = m$, for any $m \in \mathcal{M}$.

SECURITY. Below we will use the strongest notion of *Insider* security for multi-user signcryption [2]. Clearly, a weaker notion of the so called *Outsider* security easily follows as well.

As expected, the security for signcryption consists on $\mathsf{IND\text{-}CCA2}$ and $\mathsf{sUF\text{-}CMA}$ components when attacking some user $U$. Both games with the adversary, however, share the following common component. After $(\mathsf{SDK}_U, \mathsf{VEK}_U) \leftarrow \mathsf{Gen}(1^\lambda)$ is run and $\mathcal{A}$ gets $\mathsf{VEK}_U$, $\mathcal{A}$ can make up to $q_{\mathsf{SE}}$ adaptive signcryption queries $\mathsf{SigEnc}_U(m, \mathsf{VEK}_R)$ for *arbitrary* $\mathsf{VEK}_R$, as well as up to $q_{\mathsf{VD}}$ de-signcryption queries $\mathsf{VerDec}_U(\Pi, \mathsf{VEK}_S)$, again for arbitrary $\mathsf{VEK}_S$.

The $\mathsf{IND\text{-}CCA2}$ security of signcryption requires that no $\mathsf{PPT}$ adversary $\mathcal{A}$ can find some pair $m_0, m_1$ for which he can distinguish $\mathsf{SigEnc}_S(m_0, \mathsf{VEK}_U)$ from $\mathsf{SigEnc}_S(m_1, \mathsf{VEK}_U)$. Notice, to make *sense* of the statement, $\mathcal{A}$ has to output the *secret key* $\mathsf{SDK}_S$ of the sender whose messages to $U$ he can "understand". While seemingly restrictive, this is a *much stronger* guarantee than if $\mathcal{A}$ tried to do it with some sender $S$ whose key he *did not know*. A good way to interpret this requirement is to say that even when *compromising* $S$, $\mathcal{A}$ still cannot "understand" messages $S$ sent to $U$. In fact, we allow $\mathcal{A}$ much more, as he can *come up* with the secret key $\mathsf{SDK}_S$ without necessarily generating it via $\mathsf{Gen}$! Formally, for any $\mathsf{PPT}$ $\mathcal{A}$ running in time $t$,

$$\Pr\left[b = \tilde{b} \;\middle|\; \begin{array}{c} (m_0, m_1, \mathsf{SDK}_S, \alpha) \leftarrow \mathcal{A}^{\mathsf{SigEnc}_U(\cdot,\cdot), \mathsf{VerDec}_U(\cdot,\cdot)}(\mathsf{VEK}_U, \mathsf{find}),\ b \xleftarrow{R} \{0,1\}, \\ \Pi \leftarrow \mathsf{SigEnc}_S(m_b, \mathsf{VEK}_U),\ \tilde{b} \leftarrow \mathcal{A}^{\mathsf{SigEnc}_U(\cdot,\cdot), \mathsf{VerDec}_U(\cdot,\cdot)}(\Pi;\ \alpha, \mathsf{guess}) \end{array}\right] \leq \frac{1}{2} + \varepsilon_{\mathsf{SC-CCA2}}$$

where $\varepsilon_{\mathsf{SC-CCA2}}$ is negligible in the security parameter $\lambda$, and $(\mathsf{SDK}_U, \mathsf{VEK}_U) \leftarrow \mathsf{Gen}(1^\lambda)$ is implicitly called at the beginning. In the guess stage, $\mathcal{A}$ only has the natural restriction of not querying $\mathsf{VerDec}_U$ with $(\Pi, \mathsf{VEK}_S)$, but can still use $(\Pi, \mathsf{VEK}_{S'})$ for $\mathsf{VEK}_{S'} \neq \mathsf{VEK}_S$.

For $\mathsf{sUF\text{-}CMA}$ security, no $\mathsf{PPT}$ $\mathcal{A}$ can forge a "valid" signcryption $\Pi$ (of some message $m$) from $U$ to *any* user $R$, provided that $\Pi$ was not previously returned from a query to $\mathsf{SigEnc}_U$. Again, to make sense of the word

"valid", $\mathcal{A}$ has to come up with the presumed secret key $\mathsf{SDK}_R$ as part of his forgery. Again, this seemingly restrictive condition makes the definition actually *stronger*, similar to the IND-CCA2 case. Formally, for any PPT $\mathcal{A}$ running in time $t$,

$$\Pr\left[\mathsf{VerDec}_R(\Pi, \mathsf{VEK}_U) = m \wedge m \neq \perp \;\middle|\; (\Pi, \mathsf{SDK}_R) \leftarrow \mathcal{A}^{\mathsf{SigEnc}_U(\cdot,\cdot),\mathsf{VerDec}_U(\cdot,\cdot)}(\mathsf{VEK}_U)\right] \leq \varepsilon_{\mathsf{SC-CMA}}$$

where $\varepsilon_{\mathsf{SC-CMA}}$ is negligible in the security parameter $\lambda$, $\mathsf{Gen}(1^\lambda)$ is implicit, and $\mathcal{A}$ did not obtain $\Pi$ in response to any $\mathsf{SigEnc}_U(m, \mathsf{VEK}_R)$ query. We call any scheme satisfying these properties a $(t, \varepsilon_{\mathsf{SC-CCA2}}, \varepsilon_{\mathsf{SC-CMA}}, q_{\mathsf{VD}}, q_{\mathsf{SE}})$-secure signcryption scheme.

### 5.2 $\mathcal{PbPS}$ Gives Secure Signcryption

Now, we can formally argue that universal two-padding schemes, when used in the $\mathcal{PbPS}$ paradigm, are sufficient for secure signcryption. Recall, in our setting each user $U$ generates a trapdoor permutation $f_U = \mathsf{VEK}_U$, of which only $U$ knows the trapdoor $f_U^{-1} = \mathsf{SDK}_U$.

To signcrypt a message from $S$ to $R$, one is first tempted to generate the two-padding $(w, s) \leftarrow \mathsf{PAD}(m)$, and then compute the signcryption $\psi \| \sigma \leftarrow f_R(w) \| f_S^{-1}(s)$. De-signcryption is done in reverse, by first recovering $w = f_R^{-1}(\psi)$, $s = f_S(\sigma)$, and finally $m = \mathsf{DePAD}(w, s)$. We let $Basic\text{-}\mathcal{PbPS}$ (Basic Padding-based Signcryption Scheme) denote this natural signcryption scheme. In fact, $Basic\text{-}\mathcal{PbPS}$ is secure in the simplistic "two-party" model [2]. Unfortunately, it is trivially insecure according to our definition of multi-party signcryption. For example, an adversary $\mathcal{A}$ can ask some honest $S$ to send a message $m$ to $\mathcal{A}$. Upon receiving $\psi \| \sigma$ from $S$, $\mathcal{A}$ can recover $w = f_{\mathcal{A}}^{-1}(\psi)$, and then forge a valid signcryption $f_R(w) \| \sigma$ of $m$ from $S$ to any other user $R$. Similar "identity fraud" allows $\mathcal{A}$ to break the IND-CCA2 security as well.

As this demonstrates, in the multi-user setting, the signcryption must non-trivially depend on the identities of the message's sender and its intended recipient, in order to protect both the authenticity of $S$'s messages and the privacy of $R$'s messages. In this section, we provide one simple way to accomplish this; an optimized version is presented in Section 6 when we introduce the notion of associated data. We let $\mathcal{PbPS}$ be the following scheme, where $h$ is a collision-resistant hash function (CRHF) with running time $T_h$. The sender $S$ simply applies $Basic\text{-}\mathcal{PbPS}$ to the message $m' = m \| h(\mathsf{VEK}_S, \mathsf{VEK}_R)$. On the receiver's side, $R$ first recovers $m' = m \| \hbar$ just like in $Basic\text{-}\mathcal{PbPS}$, but outputs $m$ only if $\hbar = h(\mathsf{VEK}_S, \mathsf{VEK}_R)$; otherwise, $R$ outputs $\perp$.

**Theorem 2** *$\mathcal{PbPS}$ is a $\Big(t - \mathrm{O}((q_D + q_S) \cdot (T_f + T_h)), (\varepsilon_{\mathsf{CCA2}} + q_h^2 \cdot \varepsilon_{\mathsf{CRHF}}), (\varepsilon_{\mathsf{CMA}} + q_h^2 \cdot \varepsilon_{\mathsf{CRHF}}), q_D, q_S, q_H\Big)$-secure signcryption, provided $h$ is a $(t, \varepsilon_{\mathsf{CRHF}})$-secure CRHF and $(\mathsf{PAD}, \mathsf{DePAD})$ is any $(t, \varepsilon_{\mathsf{CCA2}}, \varepsilon_{\mathsf{CMA}}, q_D, q_S)$-secure universal two-padding scheme.*

We include the proof of this theorem in Appendix E. As an immediate corrolary, however, we get that any Feistel two-padding schemes, such as PSEP, PSS-R, OAEP, OAEP+, SAP, *etc.*, can be used in $\mathcal{PbPS}$.

## 6 Signcryption for Long Messages (with Associated Data)

The signcryption scheme described in Section 5 is adequate for short messages or keys; however, we can construct a signcryption scheme for long messages virtually "for free", using an approach similar to the one described in [11]. We first extend our Feistel Two-Padding to support labels, or "associated data" as in [31]. Then, to handle long messages, we apply the previous signcryption scheme to a short one-time key used to encrypt the long message. The ciphertext of the long message is simply attached to the signcryption in the form of a label. This provides the required authenticity guarantee.

### 6.1 Two-Paddings with Associated Data

SYNTAX. The syntax is similar to the previously-described two-padding scheme, with the addition of a new "label" parameter $\mathcal{L}$ provided to both PAD and DePAD. That is, we now write $(w, s) \leftarrow \mathsf{PAD}^{\mathcal{L}}(m)$ and $m \leftarrow \mathsf{DePAD}^{\mathcal{L}}(w, s)$. PadEnc and PadSig are similarly enhanced as follows: $\mathcal{L}\|\psi\|s \leftarrow \mathsf{PadEnc}^{\mathcal{L}}_f(m)$ and $\mathcal{L}\|w\|\sigma \leftarrow \mathsf{PadSig}^{\mathcal{L}}_f(m)$. Note that $\psi$ and $\sigma$ are defined exactly as before, *i.e.*, $f(w)$ and $f^{-1}(s)$, respectively. Naturally, PadDec and PadVer now expect $\mathcal{L}$ at the front of their input string as well.

SECURITY. We call $\mathcal{LPS} = (\mathsf{PAD}, \mathsf{DePAD})$ a $(t, \varepsilon_{\mathsf{CCA2}}, \varepsilon_{\mathsf{CMA}}, q_D, q_S)$-secure *labelled two-padding* scheme if, for any TDPs $f$ and $f'$, the corresponding $\mathsf{PadEnc}^{\mathcal{L}}_f$ is a $(t, \varepsilon_{\mathsf{CCA2}}, q_D)$-secure IND-CCA2 encryption on the message input $m$ (considering the entire output $\mathcal{L}\|\psi\|s$ as a ciphertext) and $\mathsf{PadSig}^{\mathcal{L}}_{f'}$ is a $(t, \varepsilon_{\mathsf{CMA}}, q_S)$-secure sUF-CMA signature on $(\mathcal{L}, m)$. Note that $\mathcal{A}$ must choose a fixed label $\mathcal{L}^*$, in addition to $m_0$ and $m_1$, during the find stage of the IND-CCA2 game. Although the IND-CCA2 security does not require hiding for $\mathcal{L}$—in fact, it is given in the clear—$\mathcal{L}$ is considered part of the ciphertext. For example, given a challenge ciphertext $\mathcal{L}^*\|\psi^*\|s^*$ during the IND-CCA2 game, the adversary may ask the decryption oracle to decrypt $\mathcal{L}'\|\psi^*\|s^*$ for any $\mathcal{L}' \neq \mathcal{L}^*$.

We call $\mathcal{LPS} = (\mathsf{PAD}, \mathsf{DePAD})$ a $(t, \varepsilon_{\mathsf{CCA2}}, \varepsilon_{\mathsf{CMA}}, q_D, q_S)$-secure *labelled universal two-padding* scheme if it is a $(t, \varepsilon_{\mathsf{CCA2}}, \varepsilon_{\mathsf{CMA}}, q_D, q_S)$-secure labelled two-padding even when a user is reusing the same TDP $f$ for both encryption and signature. This definition is analogous to that of Section 2.2.

**Definition 2 (Labelled Feistel Two-Padding)** *Let $\mathcal{C} = (\mathsf{Setup}, \mathsf{Commit}, \mathsf{Open}, \mathsf{Extract})$ be any extractable commitment scheme and $H : \{0,1\}^* \times \{0,1\}^{k_1} \rightarrow \{0,1\}^{k_2}$ be a random oracle. A Labelled Feistel Two-Padding $\mathsf{PAD}^{\mathcal{L}}(m) \rightarrow (w, s)$ is given by:*

$$\begin{aligned} (c, d) &\leftarrow \mathsf{Commit}(m) \\ w &\leftarrow c \\ s &\leftarrow H(\mathcal{L}, w) \oplus d \end{aligned}$$

*Note that $(w, s)$ represents a Feistel Transform on input $\langle d, c\rangle$ using $H(\mathcal{L}, \cdot)$. The corresponding* DePAD *computes $d = H(\mathcal{L}, w) \oplus s$ and $c = w$, then returns $\mathsf{Open}(c, d)$.*

**Theorem 3** *The Labelled Feistel Two-Padding described above is a secure labelled universal two-padding scheme, with the same exact security as the Feistel Two-Padding in Theorem 1, including the tighter exact security when a claw-free permutation family is used instead of a* TDP.

The proof of this theorem is in Appendix C. Intuitively, the label $\mathcal{L}$ selects a random oracle $H(\mathcal{L}, \cdot)$ from an infinite family of oracles to be applied in the Feistel transform. Using an incorrect oracle will cause $d$ to become randomly defined; by Lemma 2, this will cause Open to return invalid. Thus, the label is effectively bound to the rest of the padding. Notice that security for the label is "free" as a consequence of our use of a random oracle in the padding: (1) there is *no* loss in security due to the inclusion of the label, and (2) the computational cost of adding the label is negligible in practice, as it entails merely increasing the size of input to $H$.

### 6.2 Signcryption with Associated Data

SYNTAX. The syntax of the labelled signcryption and de-signcryption algorithms differs from normal signcryption only by the inclusion of a "label" parameter $\ell$. These algorithms are denoted $\mathsf{SigEnc}^{\ell}$ and $\mathsf{VerDec}^{\ell}$.

SECURITY. The security notions for these labelled algorithms are similar to those of standard signcryption, with the added requirement that $\ell$ is considered part of the ciphertext (for the purposes of CCA2 decryption oracle queries), and must be authenticated. However, there is no hiding requirement for $\ell$.

If we replace the universal two-padding in the $\mathcal{PbPS}$ construction described in Section 5 with a labelled universal two-padding, the resulting signcryption $\mathsf{SigEnc}^{\ell}$ supports associated data by simply setting $\mathcal{L} = \ell$. This is a natural extension, and the proof is similar to that of Theorem 2. However, we can do better than this by taking advantage of the associated data to bind the identities of the participants, rather than wasting part of the message space to append a hash of their public keys. We define *Labelled Padding-based Parallel Signcryption* ($\ell$-$\mathcal{PbPS}$) to be $Basic$-$\mathcal{PbPS}$ using a labelled universal two-padding with label $\mathcal{L} = \ell \| \mathsf{VEK}_S \| \mathsf{VEK}_R$, where $\ell$ is the associated data to be used for signcryption and $\mathsf{VEK}_S$, $\mathsf{VEK}_R$ are the public keys of the sender and recipient, respectively.

**Theorem 4** *$\ell$-$\mathcal{PbPS}$ as described above is a $(t - \mathrm{O}((q_D + q_S) \cdot T_f), \varepsilon_{\mathsf{CCA2}}, \varepsilon_{\mathsf{CMA}}, q_D, q_S, q_H)$-secure signcryption, provided that $(\mathsf{PAD}, \mathsf{DePAD})$ is a $(t, \varepsilon_{\mathsf{CCA2}}, \varepsilon_{\mathsf{CMA}}, q_D, q_S)$-secure labelled universal two-padding scheme.*

The proof is similar to that of Theorem 2 but it provides improved exact security since the CRHF is not involved. Note that including the public keys in the associated data does not increase the length of a ciphertext since the keys are already available.

### 6.3 Signcryption of Long Messages using Associated Data

Using the "concealment" approach described in [11], we can extend any short-message signcryption scheme with support for associated data to include support for long messages. Although arbitrary concealment schemes will suffice, for efficiency purposes we will consider concealments utilizing any one-time $(t, \varepsilon_{\mathsf{OTE}})$-secure encryption scheme $(E, D)$,[5] as discussed below. As will be obvious from our implementation below, the construction described in the following theorem is analogous to that of symmetric key authenticated encryption with support for associated data given in [11].

Let $\mathcal{SC} = (\mathsf{Gen}, \mathsf{SigEnc}, \mathsf{VerDec})$ be any signcryption scheme on $\hat{n}$-bit messages with support for associated data, and $(E, D)$ be any one-time encryption scheme with keysize $\hat{n}$. We define a signcryption scheme $\mathcal{SC}' = (\mathsf{Gen}, \mathsf{SigEnc}', \mathsf{VerDec}')$ on long messages with support for associated data as follows. Let $\mathsf{SigEnc}'^{\ell}(M) = \mathsf{SigEnc}^{L}(\tau)$, where $L = \ell \| E_\tau(M)$ and $\tau$ is a random $\hat{n}$-bit string. Similarly, $\mathsf{VerDec}'^{\ell}(\pi, \Pi) = D_\tau(\pi)$, where $\tau = \mathsf{VerDec}^{L}(\Pi)$ and $L = \ell \| \pi$.

**Theorem 5** *If $\mathcal{SC}$ is $(t, \varepsilon_{\mathsf{SC-CCA2}}, \varepsilon_{\mathsf{SC-CMA}}, q_{\mathsf{VD}}, q_{\mathsf{SE}})$-secure and $(E, D)$ is $(t, \varepsilon_{\mathsf{OTE}})$-secure (with encryption/decryption time $T_{\mathsf{OTE}}$), then $\mathcal{SC}'$ is $(t - \mathrm{O}((q_{\mathsf{VD}} + q_{\mathsf{SE}}) \cdot T_{\mathsf{OTE}}), \varepsilon_{\mathsf{SC-CCA2}} + \varepsilon_{\mathsf{OTE}}, \varepsilon_{\mathsf{SC-CMA}}, q_{\mathsf{VD}}, q_{\mathsf{SE}})$-secure.*

The proof of this theorem is given in Appendix F. The result of this theorem allows us extend $\ell$-$\mathcal{PbPS}$ to support long messages by using the padding label $\mathcal{L} = L \| \mathsf{VEK}_S \| \mathsf{VEK}_R = \ell \| E_\tau(M) \| \mathsf{VEK}_S \| \mathsf{VEK}_R$.

### 6.4 Putting the Pieces Together

We now construct a complete signcryption scheme with support for long messages and associated data by collecting the pieces described in Sections 6.1 through 6.3.

**Definition 3 (Feistel-$\ell$-$\mathcal{PbPS}$)** *Let $\mathcal{C} = (\mathsf{Setup}, \mathsf{Commit}, \mathsf{Open}, \mathsf{Extract})$ be an extractable commitment scheme and $(E, D)$ be a one-time secure encryption scheme with $\hat{n}$-bit keys. Let $H$ be a random oracle, and assume that $\langle f_S, f_S^{-1} \rangle$ and $\langle f_R, f_R^{-1} \rangle$ are $\mathsf{TDP}$s known to the sender $S$ and receiver $R$, respectively. Also, let $\ell$ denote an arbitrary length label and $M \in \{0,1\}^{poly(\hat{n})}$ be a (long) message.*

[5] *I.e.*, no distinguisher in time $t$ can tell $E_\tau(M_0)$ from $E_\tau(M_1)$ for any two messages $(M_0, M_1)$ with probability greater than $\varepsilon_{\mathsf{OTE}}$.

*Define* $\mathsf{SigEnc}_S^{\ell}(M, \mathsf{VEK}_R)$ *as the following:*

$$\begin{array}{rcl} \tau & \stackrel{R}{\leftarrow} & \{0,1\}^{\hat{n}} \\ \pi & = & E_\tau(M) \\ \mathcal{L} & = & \ell \parallel \pi \parallel \mathsf{VEK}_S \parallel \mathsf{VEK}_R \\ (c,d) & \leftarrow & \mathsf{Commit}(\tau) \\ w = c & ; & s = H(\mathcal{L}, c) \oplus d \\ \psi = f_R(w) & ; & \sigma = f_S^{-1}(s) \\ \text{Output } \Pi & = & \ell \| \pi \| \psi \| \sigma \end{array}$$

*Define* $\mathsf{VerDec}_R^{\ell}(\Pi, \mathsf{VEK}_S)$ *as the following, parsing* $\Pi$ *as* $\ell \| \pi \| \psi \| \sigma$ *and setting* $\mathcal{L} = \ell \| \pi \| \mathsf{VEK}_S \| \mathsf{VEK}_R$*:*

$$\begin{array}{rcl} w = f_R^{-1}(\psi) & ; & s = f_S(\sigma) \\ c = w & ; & d = s \oplus H(\mathcal{L}, c) \\ \tau & = & \mathsf{Open}(c,d) \\ \text{If } \tau = \perp & \Rightarrow & \text{Output } \perp \\ \text{Output } M & = & D_\tau(\pi) \end{array}$$

We also note that a user $U$ can utilize the same $f_U^{-1}, f_U$ for both sending and receiving the data.

**Theorem 6** *$\ell$-$\mathcal{PbPS}$ is a $(t - \mathrm{O}((q_D + q_S) \cdot (T_f + T_{\mathsf{OTE}})), \varepsilon_{\mathsf{CCA2}} + \varepsilon_{\mathsf{OTE}}, \varepsilon_{\mathsf{CMA}}, q_D, q_S)$-secure signcryption with associated data when instantiated with a $(t, \varepsilon_{\mathsf{hide}}, \varepsilon_{\mathsf{extract}})$-secure extractable commitment scheme $\mathcal{C}$ and a one-time $(t, \varepsilon_{\mathsf{OTE}})$-secure encryption scheme $(E, D)$, where $\varepsilon_{\mathsf{CCA2}}$, $\varepsilon_{\mathsf{CMA}}$, $q_D$, $q_S$, and $T_f$ are defined as in Theorem 1 and $T_{\mathsf{OTE}}$ as in Theorem 5.*

The proof of this theorem follows trivially from Theorem 3, Theorem 4, and Theorem 5.

PRACTICAL CONSIDERATIONS. We believe this general Feistel $\ell$-$\mathcal{PbPS}$ scheme meets nearly all of the desirable goals for a signcryption scheme we describe in Section 1 using any typical one-time secure encryption and extractable commitment. We now offer some final recommendations for a specific instantiation of Feistel $\ell$-$\mathcal{PbPS}$ that we feel best achieves our goals.

It is our recommendation to use $E_\tau(M) = K(\tau) \oplus M$ and $D_\tau(\pi) = K(\tau) \oplus \pi$ for the one-time encryption scheme, where $K : \{0,1\}^{\hat{n}} \rightarrow \{0,1\}^{poly(\hat{n})}$ is either a random oracle or a pseudo-random generator. We are already in the random oracle model, so we can achieve the tightest security by using a random oracle, giving $\varepsilon_{\mathsf{OTE}} \leq q_K \cdot 2^{-\hat{n}}$. This allows us to select $\hat{n}$ as small as 128-bits in practical schemes. We also recommend using PSEP for the extractable commitment scheme, with a suitable choice of parameters to match the input sizes of the TDPs and the desired exact security bound. Appropriate selection of the parameter $a$ even allows the use of existing OAEP or PSS-R padding implementations. For applications where bandwidth is at a premium, careful selection of the parameters allows over 1600 data bits to fit into the padding when using typical 1024-bit RSA moduli. For such applications it is possible to transfer almost 1500 message bits from the one-time encryption into the padding (along with $\tau$), resulting in less than 600 bits of overhead for long messages.

We believe that this instantiation of Feistel $\ell$-$\mathcal{PbPS}$ is extremely practical and flexible, and provides optimal exact security. Our scheme is consistent with current PKCS#1 infrastructure, and using low-exponent RSA, the cost of a signcryption or de-signcryption operation is approximately the cost of a single modular exponentiation. Thus, our recommended scheme truly satisfies *all* the goals for a signcryption scheme.

# References


[1] Jee Hea An and Mihir Bellare. Does encryption with redundancy provide authenticity? In Pfitzmann [29].

[2] Jee Hea An, Yevgeniy Dodis, and Tal Rabin. On the security of joint signature and encrytion. In Lars Knudsen, editor, *Advances in Cryptology—EUROCRYPT 2002*, Lecture Notes in Computer Science. Springer-Verlag, 28 April–2 May 2002. Available from `http://eprint.iacr.org/2002/046/`.

[3] Joonsang Baek, Ron Steinfeld, and Yuliang Zheng. Formal proofs for the security of signcryption. In David Naccache and Pascal Pailler, editors, *5th International Workshop on Practice and Theory in Public Key Cryptosystems — PKC 2002*, volume 2274 of *Lecture Notes in Computer Science*. Springer-Verlag, February 2002.

[4] Mihir Bellare, Tadayoshi Kohno, and Chanathip Namprempre. Authenticated encryption in ssh: Provably fixing the ssh binary packet protocol. In Sandhu [35].

[5] Mihir Bellare and Chanathip Namprempre. Authenticated encryption: Relations among notions and analysis of the generic composition paradigm. In Tatsuaki Okamoto, editor, *Advances in Cryptology—ASIACRYPT 2000*, volume 1976 of *Lecture Notes in Computer Science*, pages 531–545, Kyoto, Japan, 3–7 December 2000. Springer-Verlag.

[6] Mihir Bellare and Phillip Rogaway. Random oracles are practical: A paradigm for designing efficient protocols. In *Proceedings of the 1st ACM Conference on Computer and Communication Security*, pages 62–73, November 1993. Revised version appears in `http://www-cse.ucsd.edu/users/mihir/papers/crypto-papers.html`.

[7] Mihir Bellare and Phillip Rogaway. The exact security of digital signatures: How to sign with RSA and Rabin. In Ueli Maurer, editor, *Advances in Cryptology—EUROCRYPT 96*, volume 1070 of *Lecture Notes in Computer Science*, pages 399–416. Springer-Verlag, 12–16 May 1996. Revised version appears in `http://www-cse.ucsd.edu/users/mihir/papers/crypto-papers.html`.

[8] Gilles Brassard, David Chaum, and Claude Crépeau. Minimum disclosure proofs of knowledge. *Journal of Computer and System Sciences*, 37(2):156–189, October 1988.

[9] Jean-Sébastian Coron, Marc Joye, David Naccache, and Pascal Pailler. Universal padding schemes for RSA. In Yung [37]. Available from `http://eprint.iacr.org/2002/115/`.

[10] Ivan Damgård and Jesper Buus Nielsen. Perfect hiding and perfect binding universally composable commitment schemes with constant expansion factor. In Yung [37].

[11] Yevgeniy Dodis and Jee Hea An. Concealment and its applications to authenticated encryption. In Eli Biham, editor, *Advances in Cryptology—EUROCRYPT 2003*, Lecture Notes in Computer Science. Springer-Verlag, 4 May–8 May 2003.

[12] Yevgeniy Dodis, Michael J. Freedman, Stanislaw Jarecki, and Shabsi Walfish. Universal padding schemes and their applications to signcryption. Manuscript, 2003.

[13] Yevgeniy Dodis and Leonid Reyzin. On the power of claw-free permutations. In *Conference on Security in Communication Networks*, 2002.

[14] D. Dolev, C. Dwork, and M. Naor. Nonmalleable cryptography. *SIAM*, 30:391–437, 2000.

[15] Eiichiro Fujisaki, Tatsuaki Okamotoi, David Pointcheval, and Jacques Stern. RSA-OAEP is secure under the RSA assumption. In Kilian [24], pages 260–274.

[16] S. Goldwasser and S. Micali. Probabilistic encryption. *Journal of Computer and System Sciences*, 28(2):270–299, April 1984.

[17] Shafi Goldwasser, Silvio Micali, and Ronald L. Rivest. A digital signature scheme secure against adaptive chosen-message attacks. *SIAM Journal on Computing*, 17(2):281–308, April 1988.

[18] Stuart Haber and Benny Pinkas. Combining public key cryptosystems. In Samarati [34].

[19] J. Håstad, A. W. Schrift, and A. Shamir. The discrete logarithm modulo a composite hides $O(n)$ bits. *Journal of Computer and System Sciences*, 47:376–404, 1993.

[20] W. He and T. Wu. Cryptanalysis and improvement of petersen-michels signcryption schemes. *IEEE Computers and Digital Communications*, 146(2):123–124, 1999.

[21] Markus Jakobsson, Julien P. Stern, and Moti Yung. Scramble all, encrypt small. In Lars Knudsen, editor, *Fast Software Encryption: 6th Internation Workshop, FSE1999*, volume 1636 of *Lecture Notes in Computer Science*, pages 95–111. Springer-Verlag, 1999.

[22] Charanjit S. Jutla. Encryption modes with almost free message integrity. In Pfitzmann [29].

[23] Jonathan Katz and Moti Yung. Unforgeable encryption and adaptively secure modes of operation. In Bruce Schneier, editor, *Fast Software Encryption: 8th Internation Workshop, FSE2000*, volume 1978 of *Lecture Notes in Computer Science*. Springer-Verlag, 10–12 April 2000.

[24] Joe Kilian, editor. *Advances in Cryptology—CRYPTO 2001*, volume 2139 of *Lecture Notes in Computer Science*. Springer-Verlag, 19–23 August 2001.

[25] Yuichi Komano and Kazuo Ohta. Efficient universal padding techniques for multiplicative trapdoor one-way permutation. Manuscript, February 2003. MIT/LCS Seminar.

[26] Hugo Krawczyk. The order of encryption and authentication for protecting communications (or: How secure is ssl?). In Kilian [24], pages 310–331.

[27] Wenbo Mao and John Malone-Lee. Two birds one stone: Signcryption using RSA. In Marc Joye, editor, *Progress in Cryptology — CT-RSA 2003*, Lecture Notes in Computer Science. Springer-Verlag, 13–17 April 2003.

[28] H. Petersen and M. Michels. Cryptanalysis and improvement of signcryption schemes. *IEEE Computers and Digital Communications*, 145(2):140–151, 1998.

[29] Birgit Pfitzmann, editor. *Advances in Cryptology—EUROCRYPT 2001*, volume 2045 of *Lecture Notes in Computer Science*. Springer-Verlag, 6–10 May 2001.

[30] Charles Rackoff and Daniel Simon. Non-interactive zero-knowledge proof of knowledge and chosen ciphertext attack. In J. Feigenbaum, editor, *Advances in Cryptology—CRYPTO '91*, volume 576 of *Lecture Notes in Computer Science*, pages 433–444. Springer-Verlag, 1992, 11–15 August 1991.

[31] Phillip Rogaway. Authenticated-encryption with associated-data. In Sandhu [35].

[32] Phillip Rogaway, Mihir Bellare, John Black, and Ted Krovetz. OCB: A block-cipher mode of operation for efficient authenticated encryption. In Samarati [34], pages 196–205. Full version available from `http://www.cs.ucsdavis.edu/~rogaway`.

[33] *PKCS #1: RSA Encryption Standard. Version 1.5*. RSA Laboratories, November 1993. Available from `http://www.rsa.com/rsalabs/pubs/PKCS/`.

[34] Pierangela Samarati, editor. *Eighth ACM Conference on Computer and Communication Security*. ACM, November 5–8 2001.

[35] Ravi Sandhu, editor. *Ninth ACM Conference on Computer and Communication Security*. ACM, November 17–21 2002.

[36] Victor Shoup. OAEP reconsidered. In Kilian [24], pages 240–259.

[37] Moti Yung, editor. *Advances in Cryptology—CRYPTO 2002*, Lecture Notes in Computer Science. Springer-Verlag, 18–22 August 2002.

[38] Y. Zheng. Digital signcryption or how to achieve cost(signature & encryption) $\ll$ cost(signature) + cost(encryption). In Burton S. Kaliski Jr., editor, *Advances in Cryptology—CRYPTO '97*, volume 1294 of *Lecture Notes in Computer Science*, pages 165–179. Springer-Verlag, 17–21 August 1997.

[39] Y. Zheng and H. Imai. Efficient signcryption schemes on elliptic curves. *Information Processing Letters*, 68(6):227–233, December 1998.

# A Formal security definitions

ENCRYPTION. This paper only considers IND-CCA2 security for encryption: No probabilistic poly-time (PPT) adversary $\mathcal{A}$ can distinguish between the ciphertexts of two chosen messages, $m_0$ and $m_1$, given the corresponding public key EK and oracle access to Dec. Formally, for any PPT $\mathcal{A}$ which runs in two stages, find and guess, in total time $t$ making at most $q_D$ decryption oracle queries, we say that Enc is $(t, \varepsilon_{\mathsf{CCA2}}, q_D)$-secure if

$$\Pr\left[b = \tilde{b} \;\middle|\; \begin{array}{c} (\mathsf{EK}, \mathsf{DK}) \leftarrow \mathsf{Enc\text{-}Gen}(1^\lambda),\ (m_0, m_1, \alpha) \leftarrow \mathcal{A}^{\mathsf{Dec}(\cdot)}(\mathsf{EK}, \mathsf{find}), \\ b \stackrel{R}{\leftarrow} \{0,1\},\ \psi \leftarrow \mathsf{Enc}_{\mathsf{EK}}(m_b),\ \tilde{b} \leftarrow \mathcal{A}^{\mathsf{Dec}(\cdot)}(\psi;\ \alpha, \mathsf{guess}) \end{array}\right] \leq \frac{1}{2} + \varepsilon_{\mathsf{CCA2}}$$

where $\varepsilon_{\mathsf{CCA2}}$ is negligible in the security parameter $\lambda$. Naturally, $\mathcal{A}$ cannot query Dec on input $\psi$ in the guess stage.

SIGNATURES. The strongest notion of signature security, sUF-CMA, is defined as the following: For any PPT adversary $\mathcal{A}$, running in time $t$ and making at most $q_S$ signature oracle queries, we say that Sig is $(t, \varepsilon_{\mathsf{CMA}}, q_S)$-secure if

$$\Pr\left[\mathsf{Ver}_{\mathsf{VK}}(\sigma^*) \neq \perp \;\middle|\; (\mathsf{SK}, \mathsf{VK}) \leftarrow \mathsf{Sig\text{-}Gen}(1^\lambda),\ \sigma^* \leftarrow \mathcal{A}^{\mathsf{Sig}(\cdot)}(\mathsf{VK})\right] \leq \varepsilon_{\mathsf{CMA}}$$

where $\varepsilon_{\mathsf{CMA}}$ is negligible in $\lambda$, and $\sigma^*$ was not returned to $\mathcal{A}$ by $\mathsf{Sig}(\cdot)$.

TRAPDOOR PERMUTATIONS. A trapdoor permutation generator consists of the algorithms $(\mathsf{TDP\text{-}Gen}, \mathsf{Eval}, \mathsf{Inv})$. $\mathsf{TDP\text{-}Gen}(1^\lambda)$ generates the pair $\langle f, f^{-1} \rangle$, such that the algorithms $\mathsf{Eval}_f(\cdot)$ and $\mathsf{Inv}_{f^{-1}}(\cdot)$ define permutations of $\{0,1\}^k$ which are inverses of one another. We abuse notation and write $f(x)$ ($f^{-1}(y)$) for $\mathsf{Eval}_f(x)$ ($\mathsf{Inv}_{f^{-1}}(y)$).

For any PPT adversary $\mathcal{A}$ running in time $t$, we say that $f$ is a $(t, \varepsilon_{\mathsf{TDP}})$-secure TDP if

$$\Pr\left[x = \tilde{x} \;\middle|\; (f, f^{-1}) \leftarrow \mathsf{TDP\text{-}Gen}(1^\lambda),\ x \leftarrow \{0,1\}^k,\ y \leftarrow f(x),\ \tilde{x} \leftarrow \mathcal{A}(f, y)\right] \leq \varepsilon_{\mathsf{TDP}}$$

where $\varepsilon_{\mathsf{TDP}}$ is negligible in the security parameter $\lambda$.

CLAW-FREE PERMUTATIONS. Formally, for any PPT adversary $\mathcal{B}$ running in time $t$, we say that $f$ is a $(t, \varepsilon_{\mathsf{claw}})$-secure claw-free permutation if

$$\Pr\left[f(x) = g(z) \;\middle|\; (f, f^{-1}, g) \leftarrow \mathsf{CFP\text{-}Gen}(1^\lambda),\ (x, z) \leftarrow \mathcal{B}(f, g)\right] \leq \varepsilon_{\mathsf{claw}}$$

where $\varepsilon_{\mathsf{claw}}$ is negligible in the security parameter $\lambda$.

# B Proof of Theorem 1 (Feistel Two-Padding)

The following two sub-theorems (of independent interest), which establish regular ("non-universal") security of Feistel two-paddings, form the main building blocks of the proof.

**Theorem 7** *The Feistel Two-Padding described above produces an* IND-CCA2 *secure* PadEnc. *Specifically,*

$$\varepsilon_{\mathsf{CCA2}} \leq \varepsilon_{\mathsf{TDP}} + 2\varepsilon_{\mathsf{hide}} + q_D(2^{-k_2} + \varepsilon_{\mathsf{extract}} + \varepsilon_{\mathsf{bind}})$$

**Proof:** Assume there exists an adversary $\mathcal{A}$ who succeeds in the IND-CCA2 attack game with probability $1/2 + \varepsilon_{\mathsf{CCA2}}$. We describe a reduction $\mathcal{B}$ which inverts a TDP with probability nearly $\varepsilon_{\mathsf{CCA2}}$.

DESCRIPTION OF THE REDUCTION. The reduction $\mathcal{B}$ accepts a random challenge $y^*$ and attempts to produce a pre-image $x^* = f^{-1}(y^*)$. $\mathcal{B}$ finds the pre-image by running $\mathcal{A}$ and simulating responses to oracle queries made by $\mathcal{A}$. To initialize the game, $\mathcal{B}$ may run $\mathsf{Setup}(1^\lambda)$ for the extractable commitment scheme, obtaining commitment key $\mathsf{CK}$ and possibly a trapdoor $\mathcal{T}$ (which $\mathcal{B}$ never uses). $\mathcal{B}$ then runs $\mathcal{A}$ and simulates responses to $H$ oracle queries "honestly" by providing truly random responses. However, for each query $w$ to the $H$ oracle, $\mathcal{B}$ tests to see if $f(w) = y^*$, and if so, the simulation halts and $\mathcal{B}$ returns $x^* = w$. $\mathcal{B}$ also simulates responses to decryption oracle queries of the form $\psi \| s$ by examining a transcript of all $H$ oracle queries made by $\mathcal{A}$ and comparing $f(w)$ to $\psi$ for every query $w$. If $w$ such that $f(w) = \psi$ is found, $\mathcal{B}$ responds with $\mathsf{DePAD}(w, s)$; otherwise, it rejects. Note, $\mathcal{B}$ will always reject decryption oracle queries of the form $y^* \| s$, as there is never a transcript containing a query $w$ such that $f(w) = y^*$ (since in this event the simulation halts). When $\mathcal{A}$ requests a challenge ciphertext, $\mathcal{B}$ randomly selects a bit $b$ and then returns $\psi^* \| s^*$ as the challenge ciphertext corresponding to $m_b$, where $s^*$ is random and $\psi^* = y^*$ is the challenge.

ANALYSIS OF THE REDUCTION. To analyze the success of $\mathcal{B}$, we define a sequence of games $(\mathbf{G}_0, \ldots, \mathbf{G}_4)$ which we play against $\mathcal{A}$. Game $\mathbf{G}_0$ is the original "honest" $\mathsf{IND\text{-}CCA2}$ game, and game $\mathbf{G}_4$ is the game $\mathcal{B}$ runs against $\mathcal{A}$ (as described above). In game $\mathbf{G}_i$ let $S_i$ denote the event that $\mathcal{A}$ guessed $b$ correctly. By our assumption,

$$\Pr[S_0] = 1/2 + \varepsilon_{\mathsf{CCA2}} \tag{1}$$

Let $\mathbf{G}_1$ be the same as $\mathbf{G}_0$, except we replace the original decryption oracle with the decryption oracle simulation performed by $\mathcal{B}$ as described above. We will also explicitly require the decryption oracle simulation to reject any ciphertext query of the form $\psi^* \| s$, where $\psi^* \| s^*$ is the challenge ciphertext presented to $\mathcal{A}$ and $s$ is arbitrary. (The original simulation as run by $\mathcal{B}$ would automatically reject these ciphertexts, but since we are simulating the rest of the game honestly, we explicitly reject queries of this form here.) Let $\mathsf{DecBad}$ denote the event that our decryption oracle simulator differs from the decryption oracle in $\mathbf{G}_0$. We note that the simulation may fail in only two ways:

*Case 1.* The decryption oracle rejects a valid ciphertext $\psi \| s$ because $w = c = f^{-1}(\psi)$ was not queried to $H$. We note that if $H(w)$ was never queried, it is randomly defined, and thus $d = H(w) \oplus s$ will be random as well. However, by Lemma 2, the probability that $(c, d)$ is a valid pair when $d$ is random is bounded by $2^{-k_2} + \varepsilon_{\mathsf{extract}}$.

*Case 2.* The decryption oracle rejects a valid ciphertext of the form $\psi^* \| s$. This differs from the first case because $H(w^*)$ will be defined by the challenger when it computes the challenge ciphertext $\psi^* \| s^*$. However, we note that if $s$ is valid, we have found $d \neq d^*$ such that both $(c^*, d^*)$ and $(c^*, d)$ represent valid pairs. One can imagine some other reduction which plays $\mathbf{G}_1$ against $\mathcal{A}$ in order to find these pairs, attacking the binding property described in Lemma 1. Thus, this event must occur with probability at most $\varepsilon_{\mathsf{bind}}$ ($\leq 2\varepsilon_{\mathsf{extract}}$), or the binding property would be broken by such a reduction.

The decryption oracle is queried at most $q_D$ times, and thus, combining the results for the previous two cases, we have that:

$$\Pr[\mathsf{DecBad}] \leq q_D(2^{-k_2} + \varepsilon_{\mathsf{extract}} + \varepsilon_{\mathsf{bind}}) \tag{2}$$

Since $\mathbf{G}_1$ plays identically to $\mathbf{G}_0$ unless $\mathsf{DecBad}$ occurs, we find that:

$$\Pr[S_1] + \Pr[\mathsf{DecBad}] \geq \Pr[S_0] \tag{3}$$

Let $\mathbf{G}_2$ be $\mathbf{G}_1$ modified so that it halts in the event that $\mathcal{A}$ queries $H(w^*)$. We denote this event as $\mathsf{Halt}_2$. In the event that $\mathbf{G}_2$ does not halt, it has played out identically to $\mathbf{G}_1$. We have that:

$$\Pr[S_2 \mid \neg\mathsf{Halt}_2] + \Pr[\mathsf{Halt}_2] \geq \Pr[S_1] \tag{4}$$

Let $\mathbf{G}_3$ be $\mathbf{G}_2$ modified so that the $s^*$ component of the original challenge ciphertext is replaced by a random string. Denote the event that $\mathbf{G}_3$ halts as $\mathsf{Halt}_3$. We note that $\mathcal{A}$ can never obtain a response to a query $H(w^*)$ in either $\mathbf{G}_2$ or $\mathbf{G}_3$. Thus, $H(w^*)$ should appear to be perfectly random to $\mathcal{A}$, and therefore $s^* = H(w^*) \oplus d^*$ should also appear to be perfectly random to $\mathcal{A}$. That is, by replacing $s^*$ with a random string, we have only made a conceptual change to the game - the probability space remains the same as in $\mathbf{G}_2$. Clearly,

$$\Pr[S_3 \mid \neg\mathsf{Halt}_3] + \Pr[\mathsf{Halt}_3] = \Pr[S_2 \mid \neg\mathsf{Halt}_2] + \Pr[\mathsf{Halt}_2] \tag{5}$$

Let $\mathbf{G}_4$ be $\mathbf{G}_3$ modified so that the $\psi^*$ component of the original challenge ciphertext is replaced by a random string. Clearly, this implies that the value of $w^* = c^*$ which corresponds to $f^{-1}(\psi^*)$ has been replaced by a random string. Denote the event that $\mathbf{G}_4$ halts as $\mathsf{Halt}_4$. We note that neither $\mathbf{G}_3$ or $\mathbf{G}_4$ ever use any information regarding the value of $d^*$ (which may be discarded, since $s^*$ is now chosen at random), and that $\mathbf{G}_4$ simply replaces the actual commitment of $m_b$ (that is, the original value of $c^*$) with a random string. One may imagine some reduction against the hiding property of the commitment scheme which runs $\mathcal{A}$ in either $\mathbf{G}_3$ or $\mathbf{G}_4$. If the probability of any observable event in $\mathbf{G}_4$ is different from the probability of the same event in $\mathbf{G}_3$, it may be used to distinguish the actual commitment of a message (used in $\mathbf{G}_3$) from a random string (used in $\mathbf{G}_4$). Thus, the probability of any observable event in $\mathbf{G}_4$ must be within $\varepsilon_{\mathsf{hide}}$ of the probability of the same event in $\mathbf{G}_3$, and we find that:

$$(\Pr[S_4 \mid \neg\mathsf{Halt}_4] + \varepsilon_{\mathsf{hide}}) + (\Pr[\mathsf{Halt}_4] + \varepsilon_{\mathsf{hide}}) \geq \Pr[S_3 \mid \neg\mathsf{Halt}_3] + \Pr[\mathsf{Halt}_3] \tag{6}$$

If $\mathbf{G}_4$ does not halt, the entire simulation operates independently of the challenge bit $b$. In this case, $\mathcal{A}$'s probability of success is exactly $1/2$.

$$\Pr[S_4 \mid \neg\mathsf{Halt}_4] = 1/2 \tag{7}$$

Combining (1)-(7), and solving for $\Pr[\mathsf{Halt}_4]$, we find the following:

$$\Pr[\mathsf{Halt}_4] \geq \varepsilon_{\mathsf{CCA2}} - 2\varepsilon_{\mathsf{hide}} - q_D(2^{-k_2} + \varepsilon_{\mathsf{extract}} + \varepsilon_{\mathsf{bind}}) \tag{8}$$

Since $\mathbf{G}_4$ is exactly the game played by our reduction $\mathcal{B}$ (where $\mathcal{B}$ substitutes $y^*$ as the random string for $\psi^*$), and $\mathsf{Halt}_4$ is the event that $\mathcal{B}$ inverts the TDP $f$, $\mathcal{B}$ succeeds with the probability claimed above. Since this probability it at most $\varepsilon_{\mathsf{TDP}}$, the claimed upper bound of $\varepsilon_{\mathsf{CCA2}}$ follows. □

**Theorem 8** *The Feistel Two-Padding described above produces an* sUF-CMA *secure* PadSig. *Specifically,*

$$\varepsilon_{\mathsf{CMA}} \leq q_H \cdot \varepsilon_{\mathsf{TDP}} + q_S(2^{-k_1} + \varepsilon_{\mathsf{hide}}) + 2\varepsilon_{\mathsf{bind}} + 2^{-k_2} + 2\varepsilon_{\mathsf{extract}}$$

**Proof:** We describe a reduction $\mathcal{B}$ which inverts a TDP $f$ given an algorithm $\mathcal{A}$ which outputs a forgery in the sUF-CMA game.

DESCRIPTION OF THE REDUCTION. The reduction $\mathcal{B}$ accepts a random challenge $y^*$ and returns a pre-image $x^* = f^{-1}(y^*)$. $\mathcal{B}$ begins by selecting a random integer $i \in \{1, \ldots, q_H\}$, and running $\mathsf{Setup}(1^k)$ for the commitment scheme if necessary, obtaining the public CK and possibly a trapdoor $\mathcal{T}$ which $\mathcal{B}$ keeps private. If there is no trapdoor, $\mathcal{T}$ will represent the current transcript of all oracle queries made thus far during the simulation. After this initialization is complete, $\mathcal{B}$ runs $\mathcal{A}$ and simulates responses to $H$ oracle queries and signing oracle queries. In response to the $j$-th $H$ oracle query $w_j$, for $j \neq i$, $\mathcal{B}$ selects a random value $x_j$ (which is stored for reference) and defines $H(w_j) = f(x_j) \oplus \mathsf{Extract}(w_j, \mathcal{T})$. Note that $H(w_j)$ is indeed defined to be a random string, and for all $j \neq i$, $\mathcal{B}$ has a reference containing a pre-image for $s_j = H(w_j) \oplus d_j$, where the decommitment $d_j$ has a matching $c_j = w_j$ (by the extractability property). In response to the $i$-th oracle query $H(w_i)$, $\mathcal{B}$ defines $H(w_i) = s^* \oplus \mathsf{Extract}(w_i, \mathcal{T})$, where $s^* = y^*$ is the challenge of $\mathcal{B}$.

In order to respond to a signing oracle query on $m$, $\mathcal{B}$ computes $(w, s) \leftarrow \mathsf{PAD}(m)$. Either $H(w)$ is previously defined, or $\mathcal{B}$ selects some random $x$ and defines $H(w)$ as before (although here $d$ is directly known without calling $\mathsf{Extract}$). Since $H(w)$ was defined in this fashion, $\mathcal{B}$ is able to compute the signature $w \| \sigma$, where $\sigma = x = f^{-1}(s)$. (Note that $H$ oracle queries generated by the signing oracle simulation are not counted toward $q_H$.) $\mathcal{B}$ successfully inverts the $\mathsf{TDP}$ by returning $x^* = \sigma^*$ if $\mathcal{A}$ returns a forgery of the form $w^* \| \sigma^*$ where $\sigma^* = f^{-1}(y^*)$, which corresponds to forgery derived by $\mathcal{A}$ using the $i$-th oracle query.

ANALYSIS OF THE REDUCTION. The simulation in $\mathcal{B}$ is a faithful recreation of the standard $\mathsf{sUF\text{-}CMA}$ game unless the signing oracle simulation fails. Let $\mathsf{SigFail}$ denote the event that this happens. The only possible failures occur when the signing oracle on message $m$ produces a $(w, s) \leftarrow \mathsf{PAD}(m)$ pair for which either (1) $w = w_i$ (since $\mathcal{B}$ does not know a pre-image corresponding to $H(w_i)$) or (2) $H(w)$ was previously defined for a valid pair $(w, s')$ such that $s' \neq s$. Denote the event that the first case occurs as $\mathsf{SigFail}_1$ and the event that the second case occurs as $\mathsf{SigFail}_2$.

We compute the probability of $\mathsf{SigFail}_1$ by imagining a reduction against the hiding property of the commitment. The reduction runs the $\mathsf{sUF\text{-}CMA}$ game with our simulated signing oracle against $\mathcal{A}$ to obtain $w_i$, and then queries its own "commitment" oracle on the same message $m$ that $\mathcal{A}$ wants to sign. The reduction then outputs 1 if this challenge is equal to $w_i$. If the challenge was a truly random string $R$, the probability that $R = w_i$ is exactly $2^{-k_1}$. By the hiding property of our commitment, we must therefore have that the "honest" commitment $w = c(m)$ is equal to $w_i$ with probability at most $2^{-k_1} + \varepsilon_{\mathsf{hide}}$. However, since $\mathsf{SigFail}_1$ can happen on any of $q_S$ signing queries made by $\mathcal{A}$, we get $\Pr[\mathsf{SigFail}_1] \leq q_S(2^{-k_1} + \varepsilon_{\mathsf{hide}})$.

We compute the probability of $\mathsf{SigFail}_2$ by noting that $s' \neq s$ implies that $(c = w,\ d = H(w) \oplus s)$ and $(c = w,\ d' = H(w) \oplus s')$ are both valid pairs and $d \neq d'$. But then we could easily create a reduction against the binding property described in Lemma 1, which runs the $\mathsf{sUF\text{-}CMA}$ game with our simulated signing oracle against $\mathcal{A}$ to obtain $w$, $s$, and $s'$. The reduction would then recover $d$ and $d'$ from $s$ and $s'$, and output $(c, d, d')$ breaking the binding property. Therefore, this case must occur with probability at most $\varepsilon_{\mathsf{bind}}$, and we have $\Pr[\mathsf{SigFail}_2] \leq \varepsilon_{\mathsf{bind}}$. Combining these bounds, we get:

$$\Pr[\mathsf{SigFail}] = \Pr[\mathsf{SigFail}_1] + \Pr[\mathsf{SigFail}_2] \leq q_S(2^{-k_1} + \varepsilon_{\mathsf{hide}}) + \varepsilon_{\mathsf{bind}} \tag{9}$$

Let us also denote $\mathsf{ForgeBad}$ as the event that $\mathcal{A}$ outputs a valid forgery $w \| \sigma$, such that either (1) $\mathcal{A}$ "reused" one of the $w$'s returned by the signing oracle, but with a different $\sigma$; or (2) the above did not happen and $H(w)$ was not first queried by $\mathcal{A}$; or (3) $f(\sigma) = s \neq H(w) \oplus \mathsf{Extract}(w, \tau)$. In the first case, we can easily build a reduction breaking the binding property described in Lemma 1, so it can happen with probability at most $\varepsilon_{\mathsf{bind}}$. In the second case, $H(w)$ remains randomly defined, which implies a random $d$, and Lemma 2 implies that $\mathcal{A}$ would be successful with probability at most $2^{-k_2} + \varepsilon_{\mathsf{extract}}$. The last case corresponds exactly to breaking the extractability property. Again, in this case we can easily build a reduction against the extractability property, so this case must occur with probability at most $\varepsilon_{\mathsf{extract}}$. Thus, totaling these probabilities, we have:

$$\Pr[\mathsf{ForgeBad}] \leq (2^{-k_2} + \varepsilon_{\mathsf{extract}}) + \varepsilon_{\mathsf{bind}} + \varepsilon_{\mathsf{extract}} \tag{10}$$

Now, conditioned on $(\neg\mathsf{SigFail} \wedge \neg\mathsf{ForgeBad})$, if $\mathcal{A}$ successfully forges, the forgery must correspond to one of the $q_H$ queries of the $H$ oracle. In this case $\mathcal{B}$ can invert the $\mathsf{TDP}$ provided the forgery corresponds to the $i$-th oracle query, since in this case we will have $s^* = H(w_i) \oplus d_i = y^*$. Since $i$ is chosen randomly and independently of $\mathcal{A}$'s operation, using Equations (9) and (10) we get the required bound:

$$\begin{aligned} \varepsilon_{\mathsf{TDP}} \geq \Pr[\mathcal{B} \text{ succeeds}] &\geq \Big(\varepsilon_{\mathsf{CMA}} - \Pr[\mathsf{SigFail}] - \Pr[\mathsf{ForgeBad}]\Big) \Big/ q_H \\ &\geq \Big(\varepsilon_{\mathsf{CMA}} - q_S(2^{-k_1} + \varepsilon_{\mathsf{hide}}) - 2\varepsilon_{\mathsf{bind}} - 2^{-k_2} - 2\varepsilon_{\mathsf{extract}}\Big) \Big/ q_H \end{aligned}$$

To conclude our proof of Theorem 1, we must extend the proofs of the sub-theorems by adding a PadSig oracle to the IND-CCA2 proof and a PadEnc oracle to the sUF-CMA proof. (Additionally, we also must consider the extension to claw-free permutations.) For conciseness, we omit the details of these extensions until Appendix C. In fact, Theorem 1 follows as a special case of Theorem 3 with the label $\mathcal{L}$ replaced by the empty string. □

# C Proof of Theorem 3 (Labelled Feistel Two-Padding)

The proof is established by definition, given the following two theorems (which are of independent interest).

**Theorem 9** *Labelled Feistel Two-Padding produces an* $(t', \varepsilon_{\mathsf{CCA2}}, q_D, q_S, q_H)$*-secure* IND-CCA2 *encryption* $\mathsf{PadEnc}_f^{\mathcal{L}}$*, in the presence of a* $\mathsf{PadSig}_f^{\mathcal{L}}$ *signing oracle (using the same* TDP $f$*), where*

$$\varepsilon_{\mathsf{CCA2}} \leq \varepsilon_{\mathsf{TDP}} + 2\varepsilon_{\mathsf{hide}} + \varepsilon_{\mathsf{bind}} + q_D(2^{-k_2} + \varepsilon_{\mathsf{extract}} + \varepsilon_{\mathsf{bind}})$$

*and* $t' = t - \mathrm{O}((q_H + q_S) \cdot (T_f + T_{extract}))$.

**Proof:** The proof is the same as the proof of Theorem 7, with a few minor alterations to the reduction. In particular, $H$ oracle queries are now of the form $(\mathcal{L}, w)$. This does not in fact alter *any* of the probabilities in the analysis, since a change to any part of the pair $(\mathcal{L}, w)$ will now have the same effect as a change to $w$ in the original proof. This effectively "binds" $\mathcal{L}$ to $w$ through the random oracle, preventing $\mathcal{A}$ from simply changing $\mathcal{L}$ to obtain a related ciphertext. In particular, recall that $\mathcal{A}$ will select some $\mathcal{L}^*$ to become part of the challenge ciphertext corresponding to $m_b$. One way to view this is that $\mathcal{L}^*$ determines the random oracle $H(\mathcal{L}^*, \cdot)$ which is of interest to $\mathcal{A}$, effectively removing consideration of any other label from our analysis. The only remaining alteration is the simulation of a signing oracle which must now be provided to $\mathcal{A}$.

The signing oracle simulation is performed as in the proof of Theorem 8, and we note that here it is not necessary to single out the $i$-th $H$ oracle query and modify the response. Thus, the signing oracle simulation can only fail if $\mathsf{SigFail}_2$ (as previously defined) occurs. This failure event is the only impact of the signing oracle simulation on the new proof, since the modified $H$ oracle simulation it employs still satisfies all the properties we require. Thus, for the purpose of analysis, we may simply introduce the signing oracle simulator along with the decryption oracle simulator in $\mathbf{G}_1$, replacing Equation 3 by:

$$\Pr[S_1] + \Pr[\mathsf{DecBad}] + \Pr[\mathsf{SigFail}_2] \geq \Pr[S_0] \tag{11}$$

Substituting this new equation into the proof, and using $\Pr[\mathsf{SigFail}_2] \leq \varepsilon_{\mathsf{bind}}$ (as per our earlier conclusion) gives the new final result:

$$\Pr[\mathsf{Halt}_4] \geq \varepsilon_{\mathsf{CCA2}} - 2\varepsilon_{\mathsf{hide}} - \varepsilon_{\mathsf{bind}} - q_D(2^{-k_2} + \varepsilon_{\mathsf{extract}} + \varepsilon_{\mathsf{bind}}) \tag{12}$$

Here, $\mathbf{G}_4$ is the game played by our new reduction, and $\mathsf{Halt}_4$ will correspond to the event that our reduction successfully inverts a TDP, giving us the claimed upper-bound. □

**Theorem 10** *The Labelled Feistel Two-Padding described above produces a* $(t', \varepsilon_{\mathsf{CMA}}, q_S, q_D, q_H)$*-secure* sUF-CMA *signature* $\mathsf{PadSig}_f^{\mathcal{L}}$*, in the presence of a* $\mathsf{PadDec}_f^{\mathcal{L}}$ *decryption oracle (using the same* TDP $f$*), where*

$$\varepsilon_{\mathsf{CMA}} \leq q_H \cdot \varepsilon_{\mathsf{TDP}} + q_S(2^{-k_1} + \varepsilon_{\mathsf{hide}}) + 2\varepsilon_{\mathsf{bind}} + q_D(2^{-k_2} + \varepsilon_{\mathsf{extract}} + \varepsilon_{\mathsf{bind}}) + 2^{-k_2} + 2\varepsilon_{\mathsf{extract}}$$

*and* $t' = t - \mathrm{O}((q_H + q_S) \cdot (T_f + T_{extract}))$.

**Proof:** The proof is the same as the proof of Theorem 8, with a few minor alterations to the reduction. In particular, we replace all appropriate instances of $w$ with the pair $(\mathcal{L}, w)$. By a similar argument to the one we used in the proof of Theorem 9, this alteration will not affect the analysis of the reduction. Most importantly, we note that the analysis now precludes the forging of a new $(\mathcal{L}, w)$ pair corresponding to some valid $\sigma$, by the extension of Equation 10. In fact, there is no additional loss of security to obtain this extension, as both events ForgeBad and SigFail occur with the same probability as before. Since this is ultimately responsible for providing the unforgeability property of the label, we can see that security of labels follows for "free". Finally, we note that the reduction must now provide a decryption oracle simulation as in the proof of Theorem 9. The decryption oracle simulation has no impact on the signing oracle simulation, but may cause our reduction to fail if the event DecBad occurs (as previously defined). Thus, the new bound we obtain, as claimed, is:

$$\begin{aligned}
\varepsilon_{\mathsf{TDP}} &\geq \Pr[\mathcal{B} \text{ succeeds}] \\
&\geq \Big(\varepsilon_{\mathsf{CMA}} - \Pr[\mathsf{SigFail}] - \Pr[\mathsf{DecBad}] - \Pr[\mathsf{ForgeBad}]\Big) \Big/ q_H \\
&\geq \Big(\varepsilon_{\mathsf{CMA}} - q_S(2^{-k_1} + \varepsilon_{\mathsf{hide}}) - 2\varepsilon_{\mathsf{bind}} - q_D(2^{-k_2} + \varepsilon_{\mathsf{extract}} + \varepsilon_{\mathsf{bind}}) - 2^{-k_2} - 2\varepsilon_{\mathsf{extract}}\Big) \Big/ q_H
\end{aligned}$$

□

TIGHTER SECURITY REDUCTION FOR CLAW-FREE PERMUTATIONS. We begin by noting that, since TDPs are special cases of claw-free permutations, the security bound previously established for IND-CCA2 security of $\mathsf{PadEnc}_f^{\mathcal{L}}$ holds. Since this bound is already tight, we will only tighten the reduction for the sUF-CMA proof, following the observation made by [13].

**Theorem 11** *If $f$ is induced by a family of $(t, \varepsilon_{\mathsf{claw}})$-secure claw-free permutations, The Labelled Feistel Two-Padding described above produces a $(t', \varepsilon_{\mathsf{CMA}}, q_S, q_D, q_H)$-secure* sUF-CMA *signature $\mathsf{PadSig}_f^{\mathcal{L}}$, in the presence of a $\mathsf{PadDec}_f^{\mathcal{L}}$ decryption oracle, where*

$$\varepsilon_{\mathsf{CMA}} \leq \varepsilon_{\mathsf{claw}} + q_S(2^{-k_1} + \varepsilon_{\mathsf{hide}}) + 2\varepsilon_{\mathsf{bind}} + q_D(2^{-k_2} + \varepsilon_{\mathsf{extract}} + \varepsilon_{\mathsf{bind}}) + 2^{-k_2} + 2\varepsilon_{\mathsf{extract}}$$

*and $t' = t - \mathrm{O}((q_H + q_S) \cdot (T_f + T_{extract}))$.*

**Proof:** We describe a reduction $\mathcal{B}$ which, given a claw-free pair $(f, g)$, finds a claw given an algorithm $\mathcal{A}$ which outputs a forgery in the sUF-CMA game.

DESCRIPTION OF THE REDUCTION. The reduction $\mathcal{B}$ against the claw-free properties of $(f, g)$ operates in a similar fashion to the earlier reduction in Theorem 10, but with modified signing and $H$ oracle simulations. To respond to the $j$-th $H$ oracle query $H(\mathcal{L}_j, w_j)$, $\mathcal{B}$ chooses a random value $z_j$, and defines $H(\mathcal{L}_j, w_j) = g(z_j) \oplus \mathsf{Extract}(w_j, \mathcal{T})$. To respond to a signing oracle query for message $m$ with label $\mathcal{L}$, $\mathcal{B}$ chooses a random $x$, computes $(w, s) \leftarrow \mathsf{PAD}^{\mathcal{L}}(m)$, and defines $H(\mathcal{L}, w) = f(x) \oplus d$ where $d$ is the decommitment corresponding to $c = w$ (which $\mathcal{B}$ may simply record during the computation of $w$ and $s$). $\mathcal{B}$ then returns a signature of the form $\mathcal{L} \| w \| x$, since clearly $x = f^{-1}(H(\mathcal{L}, w) \oplus d) = f^{-1}(\sigma)$. If $\mathcal{A}$ outputs a forgery of the form $\mathcal{L}_j \| w_j \| \sigma^*$ where $(L_j, w_j)$ correspond to the $j$-th oracle query, *for any* $j \in \{1, \ldots, q_H\}$, $\mathcal{B}$ returns $(x^* = \sigma^*, z^* = z_j)$. We note that, by construction, $f(x^*) = s_j = g(z^*)$, and $\mathcal{B}$ has successfully output a claw.

ANALYSIS OF THE REDUCTION. The analysis proceeds in an analogous fashion to our earlier analysis in Theorem 10. Let SigFail denote the event that our signing oracle simulation fails. The only possible failures occur when the signing oracle on message $m$ with label $\mathcal{L}$ produces $(w, s) \leftarrow \mathsf{PAD}^{\mathcal{L}}(m)$ pair for which either (1) $(\mathcal{L}, w)$ has already been queried to the $H$ oracle by $\mathcal{A}$ (since $\mathcal{B}$ has arranged those using $g$ rather than $f$) or

(2) $H(\mathcal{L}, w)$ was previously defined by the signing oracle for a triple $(\mathcal{L}, w, s')$ such that $s' \neq s$. Denote the event that the first case occurs as $\mathsf{SigFail}_1$ and the event that the second case occurs as $\mathsf{SigFail}_2$.

Applying the same reasoning as in Theorem 10, we notice that $\mathsf{SigFail}_1$ should occur with probability at most $(q_H 2^{-k_1} + \varepsilon_{\mathsf{hide}})$ for each of the $q_S$ signing oracle queries (since there are now at most $q_H$ different values for $(\mathcal{L}, w)$ to collide with, rather than a single value $(\mathcal{L}_i, w_i)$ as was previously the case). Thus we, we get $\Pr[\mathsf{SigFail}_1] \leq q_S(q_H 2^{-k_1} + \varepsilon_{\mathsf{hide}})$. The probability that $\mathsf{SigFail}_2$ occurs is once again bounded by the probability of breaking the binding property of the commitment scheme, and we get $\Pr[\mathsf{SigFail}_2] \leq \varepsilon_{\mathsf{bind}}$. Thus we have:

$$\Pr[\mathsf{SigFail}] = \Pr[\mathsf{SigFail}_1] + \Pr[\mathsf{SigFail}_2] \leq q_S(q_H 2^{-k_1} + \varepsilon_{\mathsf{hide}}) + \varepsilon_{\mathsf{bind}} \tag{13}$$

Defining events DecBad and ForgeBad as in the proof of Theorem 10, we find that their analysis proceeds identically, and we will not repeat it here. We also note that, conditioned on $(\neg\mathsf{SigFail} \wedge \neg\mathsf{DecBad} \wedge \neg\mathsf{ForgeBad})$, the reduction now *always* outputs a valid claw if $\mathcal{A}$ produces a valid forgery. Thus, we no longer have a multiplicative loss of $1/q_H$ in our reduction. Using Equations 13, 2, and 10, we obtain the bound:

$$\begin{aligned}
\varepsilon_{\mathsf{claw}} &\geq \Pr[\mathcal{B} \text{ succeeds}] \\
&\geq \varepsilon_{\mathsf{CMA}} - \Pr[\mathsf{SigFail}] - \Pr[\mathsf{DecBad}] - \Pr[\mathsf{ForgeBad}] \\
&\geq \varepsilon_{\mathsf{CMA}} - q_S(q_H 2^{-k_1} + \varepsilon_{\mathsf{hide}}) - q_D(2^{-k_2} + \varepsilon_{\mathsf{extract}} + \varepsilon_{\mathsf{bind}}) - 2^{-k_2} - 2\varepsilon_{\mathsf{extract}} - 2\varepsilon_{\mathsf{bind}}
\end{aligned}$$

□

# D Padding Schemes that use Extractable Commitments

In Section 4, we described how PSS-R, OAEP, OAEP+, PSEP, and SAP are all comprised of a Feistel Transform on an extractable commitment, and thus are universal secure two-paddings. Here, we provide more in-depth proof sketches supported these lemmas.

**PSEP (PROOF OF LEMMA 3).** The pair $\langle d = m_2 \| r,\ c = (m_1 \oplus G(r)) \| G'(m_2 \| r) \rangle$ resulting from PSEP is a secure extractable commitment for *any* value of $a$. Here, we briefly argue the scheme's exact security bounds for $\varepsilon_{\mathsf{hide}}$ and $\varepsilon_{\mathsf{extract}}$.padding scheme. To break hiding, an adversary $\mathcal{A}$ must differentiate $c$ from some random value $R \leftarrow \{0,1\}^{k_1}$, given the fixed $m$. It is easy to see that this can happen only if $\mathcal{A}$ queries $G(r)$ or $G'(m_2 \| r)$. Since $r$ was random,

$$\varepsilon_{\mathsf{hide}} \leq (q_G + q_{G'}) \cdot 2^{-(k_2 + a - n)}$$

To break extractability, the adversary finds some $\langle d', c \rangle$, where $d' = m_2' \| r'$, and one of two cases occur. In the first case, $m_2' \| r'$ was not queried to $G'$. In the second, the adversary finds some $d' \neq d$ that represents a birthday attack on $G'$, *i.e.*, finds some $G'(m_2' \| r') = G'(m_2 \| r)$. Upper-bounding the probability of both events in the obvious way, we get the following:

$$\varepsilon_{\mathsf{extract}} \ \leq \ 2^{-(k_1 - a)} \ + \ q_{G'}(q_{G'} - 1) \cdot 2^{-(k_1 - a + 1)} \ < \ (q_{G'}^2 + 1) \cdot 2^{-(k_1 - a)}$$

**OAEP+.** OAEP+ results in the pair $\langle d = r, c = (m \oplus G(r) \| G'(m \| r) \rangle$. We can easily see the following two results. (1) Hiding is achieved as in OAEP: $G(r)$ is a perfect OTP unless $r$ is reused and $G'(m \| r)$ also hides $m$ for random $r$. (2) Extractability is achieved as, given $c$, we examine all queries to $G'$ and look for the output value matching $c$ in its final $(k_1 - n)$ bits. For any corresponding input $m \| r$, extract $d = r$. To break extractability, an adversary must either "guess" some $d'$ without querying the random oracles, or, at the very least, perform a birthday attack on $G'$. In fact, the bound for $\varepsilon_{\mathsf{extract}}$ is tighter, as the values $\langle m, r \rangle \neq \langle m', r' \rangle$ returned by the birthday attack must simultaneously satisfy the equations $G'(m \| r) = G'(m' \| r')$ and $m \oplus G(r) = m' \oplus G(r')$.

SAP. Now, we briefly argue that the corresponding pair $\langle d = m_1 \| r \| G'(m_2),\ c = G(d) \oplus m_2 \rangle$ forms an extractable commitment. (1) Hiding is true as before: on inputs that include a random $r$ (such as $d$), $G(\cdot)$ is a perfect OTP unless $r$ is reused. (2) Extractability is achieved as we examine all input queries $(m_1' \| r' \| \gamma)$ to $G$ and look for the output value which, xor'd with $c$, yields an $m_2$ for which $G'(m_2) = \gamma$. Finding an alternative decommitment by Extract requires one to find some $\langle m_1, m_2, r \rangle$ and $\langle m_1', m_2', r' \rangle$ such that $G(m_1 \| r \| G'(m_2)) \oplus G(m_1' \| r' \| G'(m_2')) = m_2 \oplus m_2'$. This, however, can be easily seen to imply that either (1) $G'(m_2) = G'(m_2')$ for $m_2 \neq m_2'$, or (2) one has to find values $\alpha \neq \beta$ such that $G(\alpha) \oplus G(\beta)$ is equal to a fixed constant. By birthday bound, both events happen with negligible probability.

# E Proof of Theorem 2 ($\mathcal{P}b\mathcal{P}\mathcal{S}$)

This section proves that $\mathcal{P}b\mathcal{P}\mathcal{S}$ is a secure signcryption parameterized as follows:

$$\Big(t - \mathrm{O}((q_D + q_S) \cdot (T_f + T_h)), (\varepsilon_{\mathsf{CCA2}} + q_h^2 \cdot \varepsilon_{\mathsf{CRHF}}), (\varepsilon_{\mathsf{CMA}} + q_h^2 \cdot \varepsilon_{\mathsf{CRHF}}), q_D, q_S, q_H, q_h\Big)$$

Instead of reducing $\mathcal{P}b\mathcal{P}\mathcal{S}$ directly again to the underlying TDP, we make use of Theorem 1, which has already established that there exists a $(t, \varepsilon_{\mathsf{CCA2}}, \varepsilon_{\mathsf{CMA}}, q_D, q_S, q_H)$-secure universal two-padding scheme $\mathcal{P}\mathcal{S}$, such that PadEnc is a IND-CCA2 secure encryption and PadSig is a sUF-CMA secure signature, even when the same key pair is used for both encryption and signature.

Note that in the Insider model, the full universal two-padding $f(w) \| f^{-1}(s)$ reduces to just $f(w) \| s$ for the CCA2 game and $w \| f^{-1}(s)$ for the CMA game, or exactly PadEnc and PadSig, respectively. The inclusion of identities in signcryption adds a few definitional subtleties; otherwise, the reduction would be trivial.

**Proof:** Our proof of $\mathcal{P}b\mathcal{P}\mathcal{S}$ uses the following reductions:

- PadEnc is IND-CCA2 secure encryption $\Rightarrow$ $\mathcal{P}b\mathcal{P}\mathcal{S}$ is IND-CCA2 secure signcryption
- PadSig is sUF-CMA secure signature $\Rightarrow$ $\mathcal{P}b\mathcal{P}\mathcal{S}$ is sUF-CMA secure signcryption

If these reductions hold, any such $\mathcal{P}\mathcal{S}$ yields a secure signcryption $\mathcal{P}b\mathcal{P}\mathcal{S}$.

PROOF OF IND-CCA2. We give a reduction $\mathcal{B}$ which uses any adversary $\mathcal{A}$ against the IND-CCA2 security of the $\mathcal{P}b\mathcal{P}\mathcal{S}$ signcryption to break the IND-CCA2 security of PadEnc. $\mathcal{B}$ answers both signcryption and de-signcryption queries from $\mathcal{A}$. The reduction is straight-forward.

$\mathcal{B}$ handles $\mathcal{A}$'s signcryption queries of the form $(m, \mathsf{VEK}_R)$ as follows. $\mathcal{B}$ generates $m' = m \| h(\mathsf{VEK}_{\mathcal{A}}, \mathsf{VEK}_R)$ and sends $m'$ to the PadSig signing oracle. The oracle returns something of the form $(w \| \sigma) \leftarrow \mathsf{PadSig}_{\mathcal{A}}(m')$. $\mathcal{B}$ computes $\psi \leftarrow f_R(w)$ and returns $\psi \| \sigma$ as the response to $\mathcal{A}$.

At some point, $\mathcal{A}$ issues its IND-CCA2 challenge $(m_0, m_1, \mathsf{SDK}_S, \mathsf{VEK}_S)$. $\mathcal{B}$ similarly generates $(m_0', m_1')$ using $(\mathsf{VEK}_S, \mathsf{VEK}_{\mathcal{A}})$, and queries a PadEnc challenge oracle, which returns $\psi^* \| s^*$ for some $m_b'$. $\mathcal{B}$ then computes $\sigma^* \leftarrow f^{-1}_{\mathsf{SDK}_S}(s^*)$ and returns $\psi^* \| \sigma^*$ to $\mathcal{A}$ as the challenge ciphertext.

$\mathcal{B}$ handles $\mathcal{A}$'s de-signcryption queries of the form $(\psi \| \sigma, \mathsf{VEK}_S)$ as follows. $\mathcal{B}$ computes $s \leftarrow f_S(\sigma)$ and passes $\psi \| s$ to the PadDec decryption oracle. If the oracle returns some $m' = m \| \hbar$, such that $\hbar = h(\mathsf{VEK}_S, \mathsf{VEK}_{\mathcal{A}})$, $\mathcal{B}$ returns $m$. Otherwise, $\mathcal{B}$ returns $\perp$.

The re-use of keys must be considered in the case that $\mathcal{A}$ submits a valid de-signcryption query of the form $(\psi^* \| \sigma^*, \mathsf{VEK}_{S'})$. We know that $\mathsf{VEK}_{S'} \neq \mathsf{VEK}_S$, as $\mathcal{A}$ is not allowed to query the oracle with the IND-CCA2 challenge. Therefore, $\mathcal{A}$ must have found some $\mathsf{VEK}_{S'}$ that breaks the hash function's collision-resistance, *i.e.*, $h(\mathsf{VEK}_S, \mathsf{VEK}_{\mathcal{A}}) = h(\mathsf{VEK}_{S'}, \mathsf{VEK}_{\mathcal{A}})$, which is a legal oracle query in the multi-party Insider model.

Therefore, $\mathcal{B}$ has clearly completely simulated the actions of the signcryption oracle. If $\mathcal{B}$ returns the same guess $\tilde{b}$ that $\mathcal{A}$ returns, it has the same advantage in breaking the IND-CCA2 security of $\mathcal{P}b\mathcal{P}\mathcal{S}$ as $\mathcal{A}$ does for

breaking the IND-CCA2 security of PadEnc, modulo the birthday-bounded probability of finding a collision in $h$. That is, $\mathcal{B}$'s advantage is $\varepsilon_{\mathsf{CCA2}} + q_h^2 \cdot \varepsilon_{\mathsf{CRHF}}$.

PROOF OF sUF-CMA. We give a reduction $\mathcal{B}$ which uses any adversary $\mathcal{A}$ against the sUF-CMA security of the $\mathcal{P}b\mathcal{P}\mathcal{S}$ signcryption to break the sUF-CMA security of PadSig. $\mathcal{B}$ simulates responses for $\mathcal{A}$'s signcryption and de-signcryption queries in the same manner as described above.

At some point, $\mathcal{A}$ returns a forgery on $m$ of the form $(\psi^* \| \sigma^*, \mathsf{SDK}_R, \mathsf{VEK}_R)$. Given this forgery, $\mathcal{B}$ simply computes $w^* \leftarrow f^{-1}_{\mathsf{SDK}_R}(\psi^*)$ and returns $w^* \| \sigma^*$. Remember that this forgery can be on some message $m$ previously queried to the signcryption oracle, provided that the forgery differs in $\mathsf{VEK}_R$.

$\mathcal{B}$'s output $w^* || \sigma^*$ is a valid forgery on $m'$, provided that this value has not been returned previously by the PadSig oracle. This event again occurs only with the probability that $\mathcal{A}$ previously made the signcryption query $(m, \mathsf{VEK}_{R'})$, and $h(\mathsf{VEK}_{\mathcal{A}}, \mathsf{VEK}_R) = h(\mathsf{VEK}_{\mathcal{A}}, \mathsf{VEK}_{R'})$, for $\mathsf{VEK}_R \neq \mathsf{VEK}_{R'}$, Therefore, once again, $\mathcal{B}$ forges with the same probability as $\mathcal{A}$, modulo the probability of finding a collision in $h$. That is, $\mathcal{B}$'s advantage is $\varepsilon_{\mathsf{CMA}} + q_h^2 \cdot \varepsilon_{\mathsf{CRHF}}$. □

# F Proof of Theorem 5 (Signcryption of Long Messages)

**Proof:** The sUF-CMA security bound is automatic, since the notion of a forgery for signcryption with associated data encompasses the entire signcryptext, including the label. In other words, consider a reduction $\mathcal{B}$ against the sUF-CMA security of $\mathcal{SC}$ that uses any $\mathcal{A}$ that breaks the sUF-CMA security of $\mathcal{SC}'$. $\mathcal{B}$ simply answers $\mathcal{A}$'s signcryption queries $\mathsf{SigEnc}'^{\ell}(M, \mathsf{VEK})$ by selecting at random a $\tau$, and then returning $\mathsf{SigEnc}^{\ell \| E_\tau(M)}(\tau, \mathsf{VEK})$. $\mathcal{B}$ uses the obvious corresponding approach for $\mathsf{VerDec}'$ queries. Clearly, any signcryptext $A$ forges against $\mathcal{SC}'$ is also a valid forgery against $\mathcal{SC}$, and thus the reduction succeeds with the same probability as $A$ by simply returning $A$'s forgery.

The IND-CCA2 security reduction is also as described above, and the security bound follows from a simple two-step hybrid argument.

(1) We modify the original IND-CCA2 game by replacing the $E_\tau$ operation during the construction of the challenge ciphertext with $E_{\tilde{\tau}}$, where $\tilde{\tau}$ is a random key independent of the signcrypted key $\tau$. Any adversary capable of telling this game apart from the original game can be used to win the IND-CCA2 game against the underlying signcryption scheme $\mathcal{SC}$ with at least the same advantage. It does this by simply using the label $E_\tau(M_b)$ (where $b \leftarrow \{0, 1\}$) and providing $m_0 = \tau$ and $m_1 = \tilde{\tau}$ as the messages it claims to distinguish against $\mathcal{SC}$ in the IND-CCA2 attack (it also uses the same oracle simulations as $\mathcal{B}$). Thus, in this step, the advantage of $\mathcal{B}$ is reduced by at most $\varepsilon_{\mathsf{SC-CCA2}}$.

(2) We replace $E_{\tilde{\tau}}(M)$ in the challenge ciphertext by $E_{\tilde{\tau}}(\tilde{M})$, where $\tilde{M}$ is a random message. Any adversary capable of differentiating this game from the game of Step 1 can be used to break the security of the one-time encryption with at least the same advantage. (In a fashion similar to Step 1, we can use $\mathcal{SC}$ to signcrypt a random string with either $E_{\tilde{\tau}}(M)$ or $E_{\tilde{\tau}}(\tilde{M})$ as the label and use $\mathcal{A}$ to distinguish the resulting ciphertexts.) Thus, in going to this final step, the advantage of $\mathcal{B}$ is further reduced by at most $\varepsilon_{\mathsf{OTE}}$.

We note that, in the final step, $B$ cannot have any advantage over guessing, since the challenge ciphertext is random and independent of the challenge messages. Therefore, by this hybrid argument, $B$ has a total advantage at most $\varepsilon_{\mathsf{SC-CCA2}} + \varepsilon_{\mathsf{OTE}}$ in the original game, and the proof is complete. □