Security under Message-Derived Keys: Signcryption in iMessage

Mihir Bellare[1] and Igors Stepanovs[2]

[1] Department of Computer Science and Engineering, University of California San Diego, USA. mihir@eng.ucsd.edu
[2] Department of Computer Science, ETH Zürich, Switzerland. istepanovs@inf.ethz.ch

Abstract. At the core of Apple's iMessage is a signcryption scheme that involves symmetric encryption of a message under a key that is derived from the message itself. This motivates us to formalize a primitive we call Encryption under Message-Derived Keys (EMDK). We prove security of the EMDK scheme underlying iMessage. We use this to prove security of the signcryption scheme itself, with respect to definitions of signcryption we give that enhance prior ones to cover issues peculiar to messaging protocols. Our provable-security results are quantitative, and we discuss the practical implications for iMessage.

1 Introduction

Apple's iMessage app works across iOS (iPhone, iPad) and OS X (MacBook) devices. Laudably, it aims to provide end-to-end security. At its heart is a signcryption scheme.

The current scheme —we refer to the version in iOS 9.3 onwards, revised after the attacks of GGKMR [26] on the iOS 9.0 version— is of interest on two fronts. (1) *Applied*: iMessage encrypts (according to an Internet estimate) 63 quadrillion messages per year. It is important to determine whether or not the scheme provides the security expected by its users. (2) *Theoretical*: The scheme involves (symmetric) encryption of a message under a key that is derived from the message itself, an uncommon and intriguing technique inviting formalization and a foundational treatment.

Contributions in brief. *Signcryption theory*: We extend the prior Signcryption definitions of ADR [3] to capture elements particular to messaging systems, and give general results that simplify the analysis of the candidate schemes. *EMDK*: We introduce, and give definitions (syntax and security) for, Encryption under Message Derived Keys. *iMessage EMDK scheme*: We extract from iMessage an EMDK scheme and prove its security in the random-oracle model. *Composition and iMessage Signcryption*: We give a way to compose EMDK, PKE and signatures to get signcryption, prove it works, and thereby validate the iMessage signcryption scheme for appropriate parameter choices.

$\underline{\mathsf{iMsg1.Enc}(pk_r, sk_s, M)}$	$\underline{\mathsf{iMsg2.Enc}(pk_r, sk_s, M)}$
1. $K \leftarrow\!\!\$\ \{0,1\}^{128}$	1. $L \leftarrow\!\!\$\ \{0,1\}^{88}$
2. $C_1 \leftarrow \mathsf{AES\text{-}CTR.Enc}(K, M)$	2. $h \leftarrow \mathsf{HMAC}(L, pk_s \Vert pk_r \Vert M)[1..40]$
3. $C_2 \leftarrow \mathsf{RSA\text{-}OAEP.Enc}(pk_r, K)$	3. $K \leftarrow L \Vert h$
4. $H \leftarrow \mathsf{SHA1}(C_1 \Vert C_2)$	4. $C_1 \leftarrow \mathsf{AES\text{-}CTR.Enc}(K, M)$
5. $S \leftarrow \mathsf{EC\text{-}DSA.Sign}(sk_s, H)$	5. $C_2 \leftarrow \mathsf{RSA\text{-}OAEP.Enc}(pk_r, K)$
6. Return $((C_1, C_2), S)$	6. $H \leftarrow \mathsf{SHA1}(C_1 \Vert C_2)$
	7. $S \leftarrow \mathsf{EC\text{-}DSA.Sign}(sk_s, H)$
	8. Return $((C_1, C_2), S)$

Fig. 1: Encryption in iMsg1 (left) and iMsg2 (right). Here pk_r is the recipient's public RSA encryption key, sk_s is the sender's ECDSA secret signing key and pk_s is the sender's ECDSA public verification key. Our analysis and proofs consider general schemes of which the above emerge as instantiations corresponding to particular choices of primitives and parameters.

Background. By default, the iMessage chatting app encrypts communications between any two iMessage users. The encryption is end-to-end, under keys stored on the devices, meaning Apple itself cannot decrypt. In this way, iMessage joins Signal, WhatsApp and other secure messaging apps as a means to counter mass surveillance, but the cryptography used is quite different, and while the cryptography underlying Signal and WhatsApp, namely ratcheting, has received an extensive theoretical treatment [19, 12, 28, 33, 2, 29, 22], that underlying iMessage has not.

In 2016, Garman, Green, Kaptchuk, Miers and Rushanan (GGKMR) [26] gave chosen-ciphertext attacks on the then current, iOS 9 version, of iMessage that we will denote iMsg1. Its encryption algorithm is shown on the left in Figure 1. In response Apple acknowledged the attack as CVE-2016-1788 [20], and revised the protocol for iOS 9.3. We'll denote this version iMsg2. Its encryption algorithm is shown on the right in Figure 1. It has been stable since iOS 9.3. It was this revision that, for the specific purpose of countering the GGMKR attack, introduced (symmetric) encryption with message-derived keys: message M at line 4 is encrypted under a key K derived, via lines 1–3, from M itself. The question we ask is, does the fix work?

Identifying the goal. To meaningfully answer the above question we must first, of course, identify the formal primitive and security goal being targeted. Neither Apple's iOS Security Guide [4], nor GGKMR [26], explicitly do so. We suggest that it is signcryption. Introduced by Zheng [36], signcryption aims to simultaneously provide privacy of the message (under the receiver's public encryption key) and authenticity (under the sender's secret signing key), and can be seen as the asymmetric analogue of symmetric authenticated encryption. A formalization was given by An, Dodis and Rabin (ADR) [3]. They distinguish between outsider security (the adversary is not one of the users) and the stronger insider security (the adversary could be a sender or receiver).

Identifying the iMessage goal as signcryption gives some perspective on, and understanding of, the schemes and history. The iMessage schemes can be seen as using some form of ADR's Encrypt-then-Sign ($\mathcal{E}t\mathcal{S}$) method. The iMsg1 scheme turns out to be a simple scheme from ADR [3]. It may be outsider-secure, but ADR give an attack that shows it is not insider secure. (The adversary queries the sender encryption oracle to get a ciphertext $((C_1, C_2), S)$, substitutes S with a signature S' of $H = \mathsf{SHA1}(C_1 \| C_2)$ under its own signing key, which it can do as an insider, and then queries this modified ciphertext to the recipient decryption oracle to get back the message underlying the original ciphertext.) The GGKMR [26] attack on iMsg1 is a clever improvement and real-world rendition of the ADR attack. That Apple acknowledged the GGKMR attack, and modified the scheme to protect against it, indicates that they want insider security, not just outsider security, for their modified iMsg2 scheme. So the question becomes whether this goal is achieved.

SIGNCRYPTION THEORY EXTENDED. We could answer the above question relative to ADR's (existing) definitions of insider-secure signcryption, but we do more, affirming the iMsg2 signcryption scheme under stronger definitions that capture elements particular to messaging systems, making our results of more applied value.

When you send an iMessage communication to Alice, it is encrypted to *all* her devices (her iPhone, MacBook, iPad, ...), so that she can chat seamlessly across them. To capture this, we *enhance signcryption syntax*, making the encryption algorithm multi-recipient. (It takes not one, but a list of receiver public encryption keys.) We also allow associated data as in symmetric authenticated encryption [35].

We give, like in prior work [3], a privacy definition (priv) and an authenticity definition (auth); but, unlike prior work, we also give a strong, unified definition (sec) that implies auth+priv. We show that (under certain conditions) sec is implied by auth+priv, mirroring analogous results for symmetric authenticated encryption [15, 9]. Proving that a scheme satisfies sec (the definition more intuitively capturing the practical setting) now reduces to the simpler tasks of separately showing it satisfies auth and priv. These definitions and results are for both insider and outsider security, and parameterized by choices of *relaxing relations* that allow us to easily capture variants reflecting issues like plaintext or ciphertext integrity [8], gCCA2 [3] and RCCA [18].

EMDK DEFINITIONS. Recall that a scheme for conventional symmetric encryption specifies a key-generation algorithm that is run once, a prioi, to return a key k; the encryption algorithm then takes k and message m to return a ciphertext. In our definition of a scheme for (symmetric) Encryption under Message-Derived Keys (EMDK), there is no dedicated key-generation algorithm. Encryption algorithm EMDK.Enc takes only a message m, returning both a key k and a ciphertext c, so that k may depend on m. Decryption algorithm EMDK.Dec takes k —in the overlying signcryption scheme, this is communicated to the receiver via asymmetric encryption— and c to return either m or $\perp$.

We impose two security requirements on an EMDK scheme. (1) The first, called ae, adapts the authenticated encryption requirement of symmetric encryption [35]. (Our game formalizing ae is in Figure 8.) (2) The second, called rob, is a form of robustness or wrong-key detection [1, 17, 23, 24]. (Our game formalizing rob is also in Figure 8.) Of course one may define many other and alternative security goals for EMDK, so why these? We have focused on these simply because they suffice for our results.

EMDK is different from both (Symmetric) Encryption of Key-Dependent Messages (EKDM) [14, 16] and (Symmetric) Encryption secure against Related-Key Attack (ERKA) [7]. To begin with, these definitions apply to *syntactically different objects*. Namely, both EKDM and ERKA are security metrics for the standard symmetric encryption syntax where the encryption algorithm takes a key and message as input and returns a ciphertext, while in EMDK the encryption algorithm takes only a message and itself produces a key along with the ciphertext. (Note that the latter is also different from the syntax of a Key-Encapsulation mechanism, where encryption does produce a key and ciphertext, but takes no input message.) These syntactic differences make comparison moot, but one can still discuss intuitively how the security requirements relate. In the security games for EKDM there is an honestly and randomly chosen target key k, and challenge messages to be encrypted may depend on k, but in our security games for EMDK, the key is not chosen honestly and could depend on the message being encrypted. In ERKA also, like EKDM but unlike EMDK, a target key k is chosen honestly and at random. One can now have the game apply the encryption algorithm under a key k' derived from k, but this does not capture the encryption algorithm not taking a key as input but itself producing it as a function of the message, as in EKDM.

Deconstructing iMessage. Equipped with the above, we show how to cast the iMsg2 signcryption scheme as the result of a general transform (that we specify and call IMSG-SC) on a particular EMDK scheme (that we specify) and some standard auxiliary primitives (that we also specify). In Section 5, we prove that IMSG-SC works, reducing insider security (priv, auth, sec) of the signcryption scheme to the security of the constituents, leaving us with what is the main technical task, namely showing security of the EMDK scheme.

In more detail, IMSG-SC takes a scheme EMDK for encryption under message-derived keys, a public-key encryption scheme PKE and a digital signature scheme DS to return a signcryption scheme SC = IMSG-SC[EMDK, PKE, DS]. (In the body of the paper, this is done in two steps, with a multi-recipient public-key encryption scheme [6] as an intermediate point, but for simplicity we elide this here.) Both iMessage signcryption schemes (i.e. iMsg1 and iMsg2) can be seen as results of this transform. The two make the same choices of PKE and DS, namely RSA-OAEP and EC-DSA respectively, differing only in their choice of EMDK, which for iMsg1 is a trivial scheme that we call the basic scheme, and for iMsg2 a more interesting scheme that we denote IMSG-EMDK[F, SE] and discuss below. Our Section 5 result is that signcryption scheme SC = IMSG-SC[EMDK,

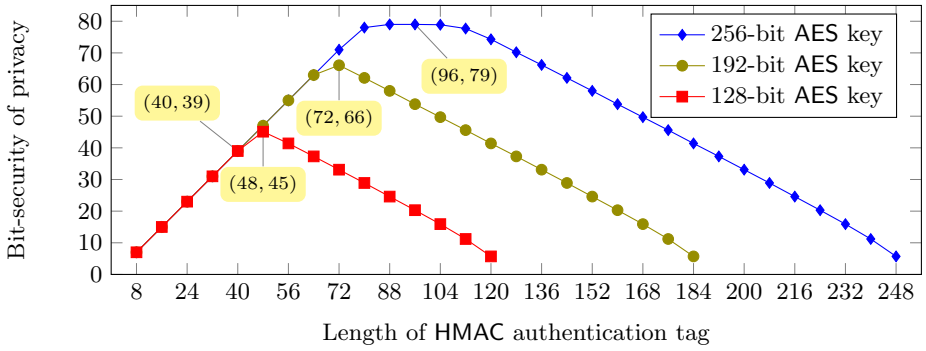


Fig. 2: Lower bounds for the bit-security of privacy achieved by iMessage, depending on the key size of AES-CTR and the length of the authentication tag returned by HMAC. iMessage 10 uses 128-bit AES key and 40-bit long HMAC authentication tag, and hence guarantees at least 39 bits of security for privacy. (Any choice of parameters guarantees 71 bits of security for authenticity.)

PKE, DS] provides insider security (priv, auth, sec) assuming ae- and rob-security of EMDK and under standard assumptions on PKE and DS.

EMDK RESULTS. In Figure 10 we specify an EMDK scheme IMSG-EMDK[F, SE] constructed from a given function family F and a given, ordinary one-time (assumed deterministic) symmetric encryption scheme SE. Setting F to HMAC and SE to AES-CTR recovers the EMDK scheme underlying iMsg2 signcryption. This EMDK scheme captures the heart of iMsg2 signcryption, namely lines 1–4 of the right side of Figure 1.

The security analysis of IMSG-EMDK[F, SE] is somewhat complex. We prove ae-security of this EMDK scheme assuming F is a random oracle and SE has the following properties: one-time IND-CPA privacy, a property we define called uniqueness, and partial key recovery security. The latter strengthens key recovery security to say that, not only is it hard to recover the key, but it is hard to recover even a prefix, of a certain prescribed length, of this key. We prove rob-security of the EMDK scheme assuming F is a random oracle and SE satisfies uniqueness and weak robustness. The properties assumed of SE appear to be true for the AES-CTR used in iMessage, and could be shown in idealized models.

PRACTICAL IMPLICATIONS FOR IMESSAGE. What we have proved is that iMsg2 signcryption is secure in principle, in the sense that the underlying template is sound. (That is, the signcryption scheme given by our IMSG-SC transform is secure assuming the underlying primitives are secure.) For the practical implications, we must consider the quantitative security guaranteed by our theorems based on the particular choices of parameters and primitives made in iMsg2 signcryption scheme. Here, things seem a bit borderline, because iMsg2 signcryption has made some specific parameter choices that seem dangerous. Considering

again the right side of Figure 1, the 128-bit AES key K at line 3 has only 88 bits of entropy —all the entropy is from the choice of L at line 1— which is not only considered small in practice but also is less than for iMsg1. (On the left side of the Figure we see that line 1 selects an AES key K with the full 128 bits of entropy.) Also the tag h produced at line 2 of the right-hand-side of the Figure is only 40 bits, shorter than recommended lengths for authentication tags. To estimate the impact of these choices, we give concrete attacks on the scheme. They show that the bounds in our theorems are tight, but do not contradict our provable-security results.

Numerical estimates based on our provable-security results say that iMessage 10 guarantees at least 39 bits of security for privacy, and 71 bits of security for authenticity, if HMAC and AES are modeled as ideal primitives. Fig. 2 shows the guaranteed bit-security of privacy for different choices of AES key length and HMAC tag length. For the small parameter choices made in iMsg2 signcryption, the attacks do approach feasibility in terms of computational effort, but we wouldn't claim they are practical, for two reasons. First, they only violate the very stringent security goals that are the target of our proofs. Second, following the GGKMR [26] attacks, Apple has implemented decryption-oracle throttling that will also curtail our attacks.

Still, ideally, a practical scheme would implement cryptography that meets even our stringent security goals without recourse to extraneous measures like throttling. We suggest that parameter and primitive choices in iMessage signcryption be revisited, for if they are chosen properly, our results do guarantee that the scheme provides strong security properties.

Discussion. When a new primitive (like EMDK) is defined, the first question of a theoretical cryptographer is often, does it exist, meaning, can it be built, and under what assumptions? At least in the random-oracle model [10] in which our results are shown, it is quite easy to build, under standard assumptions, an EMDK scheme that provides the ae+rob-security we define, and we show such a scheme in Figure 9. The issue of interest for us is less existence (to build some secure EMDK scheme) and more the security of the *particular* IMSG-EMDK[F, SE] scheme underlying iMsg2 signcryption. The motivation is mainly applied, stemming from this scheme running in security software (iMessage) that is used by millions.

But, one may then ask, WHY did Apple use their (strange) EMDK scheme instead of one like that in Figure 9, which is simpler and provable under weaker assumptions? We do not know. In that vein, one may even ask, why did Apple use EMDK at all? The literature gives Signcryption schemes that are efficient and based on standard assumptions. Why did they not just take one of them? Again, we do not know for sure, but we can speculate. The EMDK-based template that we capture in our IMSG-SC transform provides *backwards decryption compatibility*; an iMsg1 implementation can decrypt an iMsg2 ciphertext. (Of course, security guarantees revert to those of the iMsg1 scheme under such usage, but this could be offset by operational gains.) Moving to an entirely new signcryption scheme would *not* provide this backwards compatibility. But we

stress again that this is mere speculation; we did not find any Apple documents giving reasons for their choices.

Related work. We have discussed some related work above. However, signcryption is a big research area with a lot of work. We overview this in [13].

2 Preliminaries

In [13] we provide the following standard definitions. We state syntax, correctness and security definitions for function families, symmetric encryption, digital signatures, public-key encryption, and multi-recipient public-key encryption. We define the random oracle model, the ideal cipher model, and provide the birthday attack bounds. In this section we introduce the basic notation and conventions we use throughout the paper.

Basic notation and conventions. Let $\mathbb{N} = \{1, 2, \ldots\}$ be the set of positive integers. For $i \in \mathbb{N}$ we let $[i]$ denote the set $\{1, \ldots, i\}$. If X is a finite set, we let $x \xleftarrow{\$} X$ denote picking an element of X uniformly at random and assigning it to x. Let ε denote the empty string. By $x \,\|\, y$ we denote the concatenation of strings x and y. If $x \in \{0,1\}^*$ is a string then $|x|$ denotes its length, $x[i]$ denotes its i-th bit, and $x[i..j] = x[i] \ldots x[j]$ for $1 \le i \le j \le |x|$. If mem is a table, we use $\mathsf{mem}[i]$ to denote the element of the table that is indexed by i. We use a special symbol $\perp$ to denote an empty table position; we also return it as an error code indicating an invalid input to an algorithm or an oracle, including invalid decryption. We assume that adversaries never pass $\perp$ as input to their oracles.

Uniquely decodable encoding. We write $\langle a, b, \ldots \rangle$ to denote a string that is a uniquely decodable encoding of $a, b, \ldots$, where each of the encoded elements can have an arbitrary type (e.g. string or set). For any $n \in \mathbb{N}$ let $x_1, \ldots, x_n$ and $y_1, \ldots, y_n$ be two sequences of elements such that for each $i \in [n]$ the following holds: either $x_i = y_i$, or both x_i and y_i are strings of the same length. Then we require that $|\langle x_1, \ldots, x_n \rangle| = |\langle y_1, \ldots, y_n \rangle|$, and that $\langle x_1, \ldots, x_{i-1}, x_i, x_{i+1}, \ldots, x_n \rangle \oplus \langle x_1, \ldots, x_{i-1}, y_i, x_{i+1}, \ldots, x_n \rangle = \langle x_1, \ldots, x_{i-1}, (x_i \oplus y_i), x_{i+1}, \ldots, x_n \rangle$ for all $i \in [n]$.

Algorithms and adversaries. Algorithms may be randomized unless otherwise indicated. Running time is worst case. If A is an algorithm, we let $y \leftarrow A(x_1, \ldots; r)$ denote running A with random coins r on inputs $x_1, \ldots$ and assigning the output to y. We let $y \xleftarrow{\$} A(x_1, \ldots)$ be the result of picking r at random and letting $y \leftarrow A(x_1, \ldots; r)$. We let $[A(x_1, \ldots)]$ denote the set of all possible outputs of A when invoked with inputs $x_1, \ldots$. Adversaries are algorithms.

Security games and reductions. We use the code based game playing framework of [11]. (See Fig. 5 for an example.) We let $\Pr[\mathrm{G}]$ denote the probability that game G returns true. In the security reductions, we omit specifying the running times of the constructed adversaries when they are roughly the same as the running time of the initial adversary.

IMPLICIT INITIALIZATION VALUES. In algorithms and games, uninitialized integers are assumed to be initialized to 0, Booleans to false, strings to the empty string, sets to the empty set, and tables are initially empty.

BIT-SECURITY OF CRYPTOGRAPHIC PRIMITIVES. Let prim be any cryptographic primitive, and let sec be any security notion defined for this primitive. We say that prim has n bits of security with respect to sec (or n bits of sec-security) if for every adversary $\mathcal{A}$ that has advantage $\epsilon_{\mathcal{A}}$ and runtime $T_{\mathcal{A}}$ against sec-security of prim it is true that $\epsilon_{\mathcal{A}}/T_{\mathcal{A}} < 2^{-n}$. In other words, if there exists an adversary $\mathcal{A}$ with advantage $\epsilon_{\mathcal{A}}$ and runtime $T_{\mathcal{A}}$ against sec-security of prim, then prim has at most $-\log_2(\epsilon_{\mathcal{A}}/T_{\mathcal{A}})$ bits of security with respect to sec. This is the folklore definition of bit-security for cryptographic primitives. Micciancio and Walter [31] recently proposed an alternative definition for bit-security.

BIT-SECURITY LOWER BOUNDS. Let $\mathcal{BS}(\mathsf{prim}, \mathsf{sec})$ denote the bit-security of cryptographic primitive prim with respect to security notion sec. Consider any security reduction showing $\mathsf{Adv}^{\mathsf{sec}}_{\mathsf{prim}}(\mathcal{A}) \leq \sum_i \mathsf{Adv}^{\mathsf{sec_i}}_{\mathsf{prim}_i}(\mathcal{B}_i^{\mathcal{A}})$ by constructing for any adversary $\mathcal{A}$ and for each i a new adversary $\mathcal{B}_i^{\mathcal{A}}$ with runtime roughly $T_{\mathcal{A}}$. Then we can lower bound the bit-security of prim with respect to sec as

$$\mathcal{BS}(\mathsf{prim}, \mathsf{sec}) = \min_{\forall \mathcal{A}} -\log_2\left(\frac{\epsilon_{\mathcal{A}}}{T_{\mathcal{A}}}\right) \geq \min_{\forall \mathcal{A}} -\log_2\left(\frac{\sum_i \mathsf{Adv}^{\mathsf{sec_i}}_{\mathsf{prim}_i}(\mathcal{B}_i^{\mathcal{A}})}{T_{\mathcal{A}}}\right)$$

$$\geq -\log_2\left(\sum_i 2^{-\mathcal{BS}(\mathsf{prim}_i, \mathsf{sec}_i)}\right).$$

3 Signcryption

In this section we define syntax, correctness and security notions for multi-recipient signcryption schemes. We assume that upon generating any signcryption key pair (pk, sk), it gets associated to some identity id. This captures a system where users can indepenently generate their cryptographic keys prior to registering them with a public-key infrastructure. We require that all identities are distinct values in $\{0,1\}^*$. Depending on the system, each identity id serves as a label that uniquely identifies a device or a user. Note that pk cannot be used in place of the identity, because different devices can happen to use the same public keys (either due to generating the same key pairs by chance, or due to maliciously claiming someone's else public key). We emphasize that our syntax is not meant to capture identity-based signcryption, where a public key would have to depend on the identity. In [13] we provide an extensive summary of prior work on signcryption.

We focus on authenticity and privacy of signcryption in the *insider* setting, meaning that the adversary is allowed to adaptively compromise secret keys of any identities as long as that does not enable the adversary to trivially win the security games. Our definitions can also capture the *outsider* setting by considering limited classes of adversaries. We define our security notions with respect

$\pi \leftarrow\!\!\$\ \mathsf{SC.Setup}$
$(pk, sk) \leftarrow\!\!\$\ \mathsf{SC.Kg}(\pi)$
$\mathcal{C} \leftarrow\!\!\$\ \mathsf{SC.SigEnc}(\pi, id_s, pk_s, sk_s, \mathcal{R}, m, ad)$
$m \leftarrow \mathsf{SC.VerDec}(\pi, id_s, pk_s, id_r, pk_r, sk_r, c, ad)$

Fig. 3: Syntax of the constituent algorithms of signcryption scheme SC.

$\underline{\mathsf{R_m}.\mathsf{Vf}(z_0, z_1)}$	$\underline{\mathsf{R_{id}}.\mathsf{Vf}(z_0, z_1)}$
$(x_0, y_0) \leftarrow z_0\ ;\ (x_1, y_1) \leftarrow z_1$	Return $z_0 = z_1$
Return $x_0 = x_1$	

Fig. 4: Relaxing relations $\mathsf{R_m}$ and $\mathsf{R_{id}}$.

to *relaxing relations*. This allows us to capture a number of weaker security notions in a fine-grained way, by choosing an appropriate relaxing relation in each case. In [13] we define a *combined* security notion for signcryption that simultaneously encompasses authenticity and privacy, and prove that it is equivalent to the separate notions under certain conditions.

Multi-recipient signcryption schemes. A multi-recipient signcryption scheme SC specifies algorithms SC.Setup, SC.Kg, SC.SigEnc, SC.VerDec, where SC.VerDec is deterministic. Associated to SC is an identity space SC.ID. The setup algorithm SC.Setup returns public parameters π. The key generation algorithm SC.Kg takes π to return a key pair (pk, sk), where pk is a public key and sk is a secret key. The signcryption algorithm SC.SigEnc takes π, sender's identity $id_s \in \mathsf{SC.ID}$, sender's public key pk_s, sender's secret key sk_s, a set $\mathcal{R}$ of pairs (id_r, pk_r) containing recipient identities and public keys, a plaintext $m \in \{0,1\}^*$, and associated data $ad \in \{0,1\}^*$ to return a set $\mathcal{C}$ of pairs (id_r, c), each denoting that signcryption ciphertext c should be sent to the recipient with identity id_r. The unsigncryption algorithm SC.VerDec takes π, sender's identity id_s, sender's public key pk_s, recipient's identity id_r, recipient's public key pk_r, recipient's secret key sk_r, signcryption ciphertext c, and associated data ad to return $m \in \{0,1\}^* \cup \{\perp\}$, where $\perp$ indicates a failure to recover plaintext. The syntax used for the constituent algorithms of SC is summarized in Fig. 3.

Correctness of signcryption. The correctness of a signcryption scheme SC requires that for all $\pi \in [\mathsf{SC.Setup}]$, all $n \in \mathbb{N}$, all $(pk_0, sk_0), \ldots, (pk_n, sk_n) \in [\mathsf{SC.Kg}(\pi)]$ all $id_0 \in \mathsf{SC.ID}$, all *distinct* $id_1, \ldots, id_n \in \mathsf{SC.ID}$, all $m \in \{0,1\}^*$, and all $ad \in \{0,1\}^*$ the following conditions hold. Let $\mathcal{R} = \{(id_i, pk_i)\}_{1 \le i \le n}$.We require that for all $\mathcal{C} \in [\mathsf{SC.SigEnc}(\pi, id_0, pk_0, sk_0, \mathcal{R}, m, ad)]$: (i) $|\mathcal{C}| = |\mathcal{R}|$; (ii) for each $i \in \{1, \ldots, n\}$ there exists a unique $c \in \{0,1\}^*$ such that $(id_i, c) \in \mathcal{C}$; (iii) for each $i \in \{1, \ldots, n\}$ and each c such that $(id_i, c) \in \mathcal{C}$ we have $m = \mathsf{SC.VerDec}(\pi, id_0, pk_0, id_i, pk_i, sk_i, c, ad)$.

Relaxing relations. A relaxing relation $\mathsf{R} \subseteq \{0,1\}^* \times \{0,1\}^*$ is a set containing pairs of arbitrary strings. Associated to a relaxing relation R is a membership

Games $\mathrm{G}^{\mathsf{auth}}_{\mathsf{SC},\mathsf{R},\mathcal{F}}$

$\pi \xleftarrow{\$} \mathsf{SC.Setup}$; $\mathcal{F}^{\textsc{NewH},\textsc{NewC},\textsc{Exp},\textsc{SigEnc},\textsc{VerDec}}(\pi)$; Return win

$\textsc{NewH}(id)$

If $\mathsf{initialized}[id]$ then return $\bot$
$\mathsf{initialized}[id] \leftarrow \mathsf{true}$; $(pk, sk) \xleftarrow{\$} \mathsf{SC.Kg}(\pi)$; $\mathsf{pk}[id] \leftarrow pk$; $\mathsf{sk}[id] \leftarrow sk$; Return pk

$\textsc{NewC}(id, pk, sk)$

If $\mathsf{initialized}[id]$ then return $\bot$
$\mathsf{initialized}[id] \leftarrow \mathsf{true}$; $\mathsf{exp}[id] \leftarrow \mathsf{true}$; $\mathsf{pk}[id] \leftarrow pk$; $\mathsf{sk}[id] \leftarrow sk$; Return true

$\textsc{Exp}(id)$

If not $\mathsf{initialized}[id]$ then return $\bot$
$\mathsf{exp}[id] \leftarrow \mathsf{true}$; Return $\mathsf{sk}[id]$

$\textsc{SigEnc}(id_s, \mathcal{I}, m, \mathrm{ad})$

If (not $\mathsf{initialized}[id_s]$) or ($\exists id \in \mathcal{I}$: not $\mathsf{initialized}[id]$) then return $\bot$
$\mathcal{R} \leftarrow \emptyset$; For each $id \in \mathcal{I}$ do $\mathcal{R} \leftarrow \mathcal{R} \cup \{(id, \mathsf{pk}[id])\}$
$\mathcal{C} \xleftarrow{\$} \mathsf{SC.SigEnc}(\pi, id_s, \mathsf{pk}[id_s], \mathsf{sk}[id_s], \mathcal{R}, m, \mathrm{ad})$
For each $(id_r, c) \in \mathcal{C}$ do $Q \leftarrow Q \cup \{((id_s, id_r, m, \mathrm{ad}), c)\}$
Return $\mathcal{C}$

$\textsc{VerDec}(id_s, id_r, c, \mathrm{ad})$

If (not $\mathsf{initialized}[id_s]$) or (not $\mathsf{initialized}[id_r]$) then return $\bot$
$m \leftarrow \mathsf{SC.VerDec}(\pi, id_s, \mathsf{pk}[id_s], id_r, \mathsf{pk}[id_r], \mathsf{sk}[id_r], c, \mathrm{ad})$; If $m = \bot$ then return $\bot$
$z_0 \leftarrow ((id_s, id_r, m, \mathrm{ad}), c)$; If $\exists z_1 \in Q$: $\mathsf{R.Vf}(z_0, z_1)$ then return m
$\mathsf{cheated} \leftarrow \mathsf{exp}[id_s]$; If not $\mathsf{cheated}$ then $\mathsf{win} \leftarrow \mathsf{true}$
Return m

Fig. 5: Game defining authenticity of signcryption scheme SC with respect to relaxing relation R.

verification algorithm $\mathsf{R.Vf}$ that takes inputs $z_0, z_1 \in \{0,1\}^*$ to return a decision in $\{\mathsf{true}, \mathsf{false}\}$ such that $\forall z_0, z_1 \in \{0,1\}^*$: $\mathsf{R.Vf}(z_0, z_1) = \mathsf{true}$ iff $(z_0, z_1) \in \mathsf{R}$. We will normally define relaxing relations by specifying their membership verification algorithms. Two relaxing relations that will be used throughout the paper are defined in Fig. 4.

We define our security notions for signcryption with respect to relaxing relations. Relaxing relations are used to restrict the queries that an adversary is allowed to make to its unsigncryption oracle. The choice of different relaxing relations can be used to capture a variety of different security notions for signcryption in a fine-grained way. We will use relaxing relations $\mathsf{R_{id}}$ and $\mathsf{R_m}$ to capture strong vs. standard authenticity (or unforgeability) of signcryption, and IND-CCA vs. RCCA [18, 27] style indistinguishability of signcryption. In Section 5.3 we will aso define unforgeability of digital signatures with respect to relaxing relations, allowing to capture standard and strong unforgeability notions in a unified way.

Authenticity of signcryption. Consider game $\mathrm{G}^{\mathsf{auth}}$ of Fig. 5 associated to a signcryption scheme SC, a relaxing relation R and an adversary $\mathcal{F}$. The advantage of adversary $\mathcal{F}$ in breaking the AUTH-security of SC with respect to R is defined as $\mathsf{Adv}^{\mathsf{auth}}_{\mathsf{SC},\mathsf{R}}(\mathcal{F}) = \Pr[\mathrm{G}^{\mathsf{auth}}_{\mathsf{SC},\mathsf{R},\mathcal{F}}]$. Adversary $\mathcal{F}$ has access to oracles NewH, NewC, Exp, SigEnc, and VerDec. The oracles can be called in any order. Oracle NewH generates a key pair for a new honest identity id. Oracle NewC associates a key pair (pk, sk) of adversary's choice to a new corrupted identity id; it permits malformed keys, meaning sk should not necessarily be a valid secret key that matches with pk. Oracle Exp can be called to expose the secret key of any identity. The game maintains a table exp to mark which identities are exposed; all corrupted identities that were created by calling oracle NewC are marked as exposed right away. The signcryption oracle SigEnc returns ciphertexts produced by sender identity id_s to each of the recipient identities contained in set $\mathcal{I}$, encrypting message m with associated data ad. Oracle VerDec returns the plainext obtained as the result of unsigncrypting the ciphertext c sent from sender id_s to recipient id_r, with associated data ad. The goal of adversary $\mathcal{F}$ is to forge a valid signcryption ciphertext, and query it to oracle VerDec. The game does not let adversary win by querying oracle VerDec with a forgery that was produced for an exposed sender identity id_s, since the adversary could have trivially produced a valid ciphertext due to its knowledge of the sender's secret key. Certain choices of relaxing relation R can lead to another trivial attack.

A choice of relaxing relation for authenticity. When adversary $\mathcal{F}$ in game $\mathrm{G}^{\mathsf{auth}}_{\mathsf{SC},\mathsf{R},\mathcal{F}}$ calls oracle SigEnc on inputs $id_s, \mathcal{I}, m, ad$, then for each ciphertext c produced for a recipient $id_r \in \mathcal{I}$ the game adds a tuple $((id_s, id_r, m, ad), c)$ to set Q. This set is then used inside oracle VerDec. Oracle VerDec constructs $z_0 = ((id_s, id_r, m, ad), c)$ and prevents the adversary from winning the game if $\mathsf{R.Vf}(z_0, z_1)$ is true for any $z_1 \in Q$. If the relaxing relation is empty (meaning $\mathsf{R} = \emptyset$ and hence $\mathsf{R.Vf}(z_0, z_1) = \mathsf{false}$ for all $z_0, z_1 \in \{0,1\}^*$) then an adversary is allowed to trivially win the game by calling oracle SigEnc and claiming any of the resulting ciphertexts as a forgery (without changing the sender and recipient identities). Let us call this a "ciphertext replay" attack.

In order to capture a meaningful security notion, the AUTH-security of SC should be considered with respect to a relaxing relation that prohibits the above trivial attack. The strongest such security notion is achieved by considering AUTH-security of SC with respect to the relaxing relation R_{id} that is defined in Fig. 4; this relaxing relation prevents *only* the ciphertext replay attack. The resulting security notion captures the strong authenticity (or unforgeability) of signcryption. Alternatively, one could think of this notion as capturing the ciphertext integrity of signcryption.

Note that a relaxing relation R prohibits the ciphertext replay attack iff $\mathsf{R}_{\mathsf{id}} \subseteq \mathsf{R}$. Now consider the relaxing relation R_{m} as defined in Fig. 4; it is a proper superset of R_{id}. The AUTH-security of SC with respect to R_{m} captures the standard authenticity (or unforgeability, or plaintext integrity) of signcryption. The resulting security notion does not let adversary win by merely replaying an

encryption of (m, ad) from id_s to id_r for any fixed (id_s, id_r, m, ad), even if the adversary can produce a new ciphertext that was not seen before.

Capturing outsider authenticity. Game $\mathrm{G}^{\mathsf{auth}}_{\mathsf{SC},\mathsf{R},\mathcal{F}}$ captures the authenticity of SC in the *insider* setting, because it allows adversary to win by producing a forgery from an honest sender identity to an *exposed* recipient identity. This, in particular, implies that SC assures non-repudiation, meaning that the sender cannot deny the validity of a ciphertext it sent to a recipient (since the knowledge of the recipient's secret key does not help to produce a forgery). In contrast, the *outsider* authenticity only requires SC to be secure when both the sender and the recipient are honest. Our definition can capture the notion of outsider authenticity by considering a class of *outsider* adversaries that never query VerDec(id_s, id_r, c, ad) when $\mathsf{exp}[id_r] = \mathsf{true}$.

Privacy of signcryption. Consider game $\mathrm{G}^{\mathsf{priv}}$ of Fig. 6 associated to a signcryption scheme SC, a relaxing relation R and an adversary $\mathcal{D}$. The advantage of adversary $\mathcal{D}$ in breaking the PRIV-security of SC with respect to R is defined as $\mathsf{Adv}^{\mathsf{priv}}_{\mathsf{SC},\mathsf{R}}(\mathcal{D}) = 2\Pr[\mathrm{G}^{\mathsf{priv}}_{\mathsf{SC},\mathsf{R},\mathcal{D}}] - 1$. The game samples a challenge bit $b \in \{0,1\}$, and the adversary is required to guess it. Adversary $\mathcal{D}$ has access to oracles NewH, NewC, Exp, LR, and VerDec. The oracles can be called in any order. Oracles NewH, NewC, and Exp are the same as in the authenticity game (with the exception of oracle Exp also checking table ch, which is explained below). Oracle LR encrypts challenge message m_b with associated data ad, produced by sender identity id_s to each of the recipient identities contained in set $\mathcal{I}$. Oracle LR aborts if $m_0 \neq m_1$ and if the recipient set $\mathcal{I}$ contains an identity id_r that is exposed. Otherwise, the adversary would be able to trivially win the game by using the exposed recipient's secret key to decrypt a challenge ciphertext produced by this oracle. If $m_0 \neq m_1$ and none of the recipient identities is exposed, then oracle LR uses table ch to mark each of the recipient identities; the game will no longer allow to expose any of these identities by calling oracle Exp. Oracle VerDec returns the plaintext obtained as the result of unsigncrypting the ciphertext c sent from id_s to id_r with associated data ad. We discuss the choice of a relaxing relation R below. However, note that oracle LR updates the set Q (used by relaxing relation) only when $m_0 \neq m_1$. This is because the output of LR does not depend on the challenge bit when $m_0 = m_1$, and hence such queries should not affect the set of prohibited queries to oracle VerDec.

Outputs of oracle VerDec. The output of oracle VerDec in game $\mathrm{G}^{\mathsf{priv}}$ is a pair containing the plaintext (or the incorrect decryption symbol $\perp$) as its first element, and the status message as its second element. This ensures that the adversary can distinguish whether VerDec returned $\perp$ because it failed to decrypt the ciphertext (yields error message "dec"), or because the relaxing relation prohibits the query (yields error message "priv"). Giving more information to the adversary results in a stronger security definition, and will help us prove equivalence between the joint and separate security notions of signcryption in [13]. Note that an adversary can distinguish between different output branches of all other oracles used in our authenticity and privacy games.

Game $\mathrm{G}^{\mathsf{priv}}_{\mathsf{SC},\mathsf{R},\mathcal{D}}$

$b \leftarrow\!\!\$\ \{0,1\}$; $\pi \leftarrow\!\!\$\ \mathsf{SC.Setup}$; $b' \leftarrow\!\!\$\ \mathcal{D}^{\textsc{NewH},\textsc{NewC},\textsc{Exp},\textsc{LR},\textsc{VerDec}}(\pi)$; Return $b' = b$

$\textsc{NewH}(id)$

If $\mathsf{initialized}[id]$ then return $\bot$
$\mathsf{initialized}[id] \leftarrow \mathsf{true}$; $(pk, sk) \leftarrow\!\!\$\ \mathsf{SC.Kg}(\pi)$; $\mathsf{pk}[id] \leftarrow pk$; $\mathsf{sk}[id] \leftarrow sk$; Return pk

$\textsc{NewC}(id, pk, sk)$

If $\mathsf{initialized}[id]$ then return $\bot$
$\mathsf{initialized}[id] \leftarrow \mathsf{true}$; $\mathsf{exp}[id] \leftarrow \mathsf{true}$; $\mathsf{pk}[id] \leftarrow pk$; $\mathsf{sk}[id] \leftarrow sk$; Return true

$\textsc{Exp}(id)$

If (not $\mathsf{initialized}[id]$) or $\mathsf{ch}[id]$ then return $\bot$
$\mathsf{exp}[id] \leftarrow \mathsf{true}$; Return $\mathsf{sk}[id]$

$\mathrm{LR}(id_s, \mathcal{I}, m_0, m_1, ad)$

If (not $\mathsf{initialized}[id_s]$) or ($\exists id \in \mathcal{I}$: not $\mathsf{initialized}[id]$) or $|m_0| \neq |m_1|$ then return $\bot$
If $m_0 \neq m_1$ then
  If $\exists id \in \mathcal{I}$: $\mathsf{exp}[id]$ then return $\bot$
  For each $id \in \mathcal{I}$ do $\mathsf{ch}[id] \leftarrow \mathsf{true}$
$\mathcal{R} \leftarrow \emptyset$; For each $id \in \mathcal{I}$ do $\mathcal{R} \leftarrow \mathcal{R} \cup \{(id, \mathsf{pk}[id])\}$
$\mathcal{C} \leftarrow\!\!\$\ \mathsf{SC.SigEnc}(\pi, id_s, \mathsf{pk}[id_s], \mathsf{sk}[id_s], \mathcal{R}, m_b, ad)$
For each $(id_r, c) \in \mathcal{C}$ do
  If $m_0 \neq m_1$ then
    $Q \leftarrow Q \cup \{((id_s, id_r, m_0, ad), c)\}$
    $Q \leftarrow Q \cup \{((id_s, id_r, m_1, ad), c)\}$
Return $\mathcal{C}$

$\textsc{VerDec}(id_s, id_r, c, ad)$

If (not $\mathsf{initialized}[id_s]$) or (not $\mathsf{initialized}[id_r]$) then return ($\bot$, "init")
$m \leftarrow \mathsf{SC.VerDec}(\pi, id_s, \mathsf{pk}[id_s], id_r, \mathsf{pk}[id_r], \mathsf{sk}[id_r], c, ad)$
If $m = \bot$ then return ($\bot$, "dec")
$z_0 \leftarrow ((id_s, id_r, m, ad), c)$; If $\exists z_1 \in Q$: $\mathsf{R.Vf}(z_0, z_1)$ then return ($\bot$, "priv")
Return (m, "ok")

Fig. 6: Games defining privacy of signcryption scheme SC with respect to relaxing relation R.

A choice of relaxing relation for privacy. Consider relaxing relations $\mathsf{R_{id}}$ and $\mathsf{R_m}$ that are defined in Fig. 4. We recover IND-CCA security of SC as the PRIV-security of SC with respect to $\mathsf{R_{id}}$. And we capture the RCCA security of SC as the PRIV-security of SC with respect to $\mathsf{R_m}$. Recall that the intuition behind the RCCA security [18, 27] is to prohibit the adversary from querying its decryption oracle with ciphertexts that encrypt a previously queried challenge message. In particular, this is the reason that two elements are added to set Q during each call to oracle LR, one for each of m_0 and m_1. Our definition of RCCA security for SC is very similar to that of IND-gCCA2 security as proposed by An,

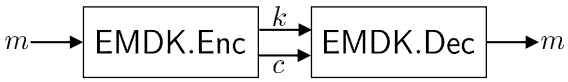


Fig. 7: Constituent algorithms of encryption scheme under message derived keys EMDK.

Dodis and Rabin [3]. The difference is that our definition passes the decrypted message as input to the relation, whereas IND-gCCA2 instead allows relations that take public keys of sender and recipient as input. It is not clear that having the relation take the public key would make our definition meaningfully stronger.

Capturing outsider privacy. Game $\mathrm{G}^{\mathsf{priv}}_{\mathsf{SC},\mathsf{R},\mathcal{D}}$ captures the privacy of SC in the *insider* setting, meaning that the adversary is allowed to request challenge encryptions from id_s to id_r even when id_s is exposed. This implies some form of forward security because exposing the sender's key does not help the adversary win the indistinguishability game. To recover the notion of *outsider* privacy, consider a class of *outsider* adversaries that never query $\mathrm{LR}(id_s, \mathcal{I}, m_0, m_1, ad)$ when $\mathsf{exp}[id_s] = \mathsf{true}$.

4 Encryption under message derived keys

We now define Encryption under Message Derived Keys (EMDK). It can be thought of as a special type of symmetric encryption allowing to use keys that depend on the messages to be encrypted. This type of primitive will be at the core of analyzing the security of iMessage-based signcryption scheme. In Section 4.1 we define syntax, correctness and basic security notions for EMDK schemes. In Section 4.2 we define the iMessage-based EMDK scheme and analyse its security.

4.1 Syntax, correctness and security of EMDK

We start by defining the syntax and correctness of encryption schemes under message derived keys. The interaction between constituent algorithms of EMDK is shown in Fig. 7. The main security notions for EMDK schemes are AE (authenticated encryption) and ROB (robustness). We also define the IND (indistinguishability) notion that will be used in Section 4.2 for an intermediate result towards showing the AE-security of the iMessage-based EMDK scheme.

Encryption schemes under message derived keys. An encryption scheme under message derived keys EMDK specifies algorithms EMDK.Enc and EMDK.Dec, where EMDK.Dec is deterministic. Associated to EMDK is a key length $\mathsf{EMDK.kl} \in \mathbb{N}$. The encryption algorithm EMDK.Enc takes a message $m \in \{0,1\}^*$ to return a key $k \in \{0,1\}^{\mathsf{EMDK.kl}}$ and a ciphertext $c \in \{0,1\}^*$. The decryption algorithm EMDK.Dec takes k, c to return message $m \in \{0,1\}^* \cup \{\perp\}$, where $\perp$ denotes incorrect decryption. Decryption correctness requires that $\mathsf{EMDK.Dec}(k, c) = m$ for all $m \in \{0,1\}^*$, and all $(k, c) \in [\mathsf{EMDK.Enc}(m)]$.

Game $\mathrm{G}^{\mathsf{ind}}_{\mathsf{EMDK},\mathcal{D}}$	Game $\mathrm{G}^{\mathsf{ae}}_{\mathsf{EMDK},\mathcal{D}}$	Game $\mathrm{G}^{\mathsf{rob}}_{\mathsf{EMDK},\mathcal{G}}$
$b \leftarrow\!\!\$\ \{0,1\}\ ;\ b' \leftarrow\!\!\$\ \mathcal{D}^{\mathrm{LR}}$ Return $b = b'$	$b \leftarrow\!\!\$\ \{0,1\}\ ;\ b' \leftarrow\!\!\$\ \mathcal{D}^{\mathrm{LR},\textsc{Dec}}$ Return $b = b'$	$(i,k) \leftarrow\!\!\$\ \mathcal{G}^{\textsc{Enc}}$ If $i \notin [n]$ then return false $m \leftarrow \mathsf{EMDK.Dec}(k, \mathsf{c}[i])$ $\mathsf{win}_1 \leftarrow (m \neq \perp)$ $\mathsf{win}_2 \leftarrow (m \neq \mathsf{m}[i])$ Return win_1 and win_2
$\underline{\mathrm{LR}(m_0, m_1)}$ If $\lvert m_0\rvert \neq \lvert m_1\rvert$ then return $\perp$ $(k,c) \leftarrow\!\!\$\ \mathsf{EMDK.Enc}(m_b)$ Return c	$\underline{\mathrm{LR}(m_0, m_1)}$ If $\lvert m_0\rvert \neq \lvert m_1\rvert$ then return $\perp$ $n \leftarrow n + 1$ $(\mathsf{k}[n], \mathsf{c}[n]) \leftarrow\!\!\$\ \mathsf{EMDK.Enc}(m_b)$ Return $(n, \mathsf{c}[n])$	$\underline{\textsc{Enc}(m)}$ $(k,c) \leftarrow\!\!\$\ \mathsf{EMDK.Enc}(m)$ $n \leftarrow n + 1\ ;\ \mathsf{m}[n] \leftarrow m\ ;$ $\mathsf{c}[n] \leftarrow c$ Return (k,c)
	$\underline{\textsc{Dec}(i,c)}$ If $i \notin [n]$ or $\mathsf{c}[i] = c$ then return $\perp$ $m \leftarrow \mathsf{EMDK.Dec}(\mathsf{k}[i], c)$ If $b = 1$ then return m Else return $\perp$	

Fig. 8: Games defining indistinguishability, authenticated encryption security, and robustness of encryption scheme under message derived keys EMDK.

Indistinguishability of EMDK. Consider game $\mathrm{G}^{\mathsf{ind}}$ of Fig. 8, associated to an encryption scheme under message derived keys EMDK, and to an adversary $\mathcal{D}$. The advantage of $\mathcal{D}$ in breaking the IND security of EMDK is defined as $\mathsf{Adv}^{\mathsf{ind}}_{\mathsf{EMDK}}(\mathcal{D}) = 2 \cdot \Pr[\mathrm{G}^{\mathsf{ind}}_{\mathsf{EMDK},\mathcal{D}}] - 1$. The game samples a random challenge bit b and requires the adversary to guess it. The adversary has access to an encryption oracle LR that takes two challenge messages m_0, m_1 to return an EMDK encryption of m_b.

Authenticated encryption security of EMDK. Consider game G^{ae} of Fig. 8, associated to an encryption scheme under message derived keys EMDK, and to an adversary $\mathcal{D}$. The advantage of $\mathcal{D}$ in breaking the AE security of EMDK is defined as $\mathsf{Adv}^{\mathsf{ae}}_{\mathsf{EMDK}}(\mathcal{D}) = 2 \cdot \Pr[\mathrm{G}^{\mathsf{ae}}_{\mathsf{EMDK},\mathcal{D}}] - 1$. Compared to the indistinguishability game from above, game G^{ae} saves the keys and ciphertexts produced by oracle LR, and also provides a decryption oracle Dec to adversary $\mathcal{D}$. The decryption oracle allows to decrypt a ciphertext with any key that was saved by oracle Enc, returning either the actual decryption m (if $b = 1$) or the incorrect decryption symbol $\perp$ (if $b = 0$). To prevent trivial wins, the adversary is not allowed to query oracle Dec with a key-ciphertext pair that were produced by the same LR query.

Robustness of EMDK. Consider game $\mathrm{G}^{\mathsf{rob}}$ of Fig. 8, associated to an encryption scheme under message derived keys EMDK, and to an adversary $\mathcal{G}$. The advantage of $\mathcal{G}$ in breaking the ROB security of EMDK is defined as $\mathsf{Adv}^{\mathsf{rob}}_{\mathsf{EMDK}}(\mathcal{G}) = \Pr[\mathrm{G}^{\mathsf{rob}}_{\mathsf{EMDK},\mathcal{G}}]$. To win the game, adversary $\mathcal{G}$ is required to find (c, k_0, k_1, m_0, m_1) such that c decrypts to m_0 under key k_0, and c decrypts to m_1 under key k_1, but $m_0 \neq m_1$. Furthermore, the game requires that the ciphertext (along with one

$\underline{\mathsf{EMDK.Enc}^{\mathrm{RO}}(m)}$	$\underline{\mathsf{EMDK.Dec}^{\mathrm{RO}}(k,c)}$	$\underline{\mathrm{RO}(z,\ell)}$
$k \leftarrow\!\!\$\ \{0,1\}^{\mathsf{EMDK.kl}}$; $\ell \leftarrow \lvert m\rvert$	$(x,h) \leftarrow c$; $\ell \leftarrow \lvert x\rvert$	If $T[z,\ell] = \perp$ then
$x \leftarrow m \oplus \mathrm{RO}(k,\ell)$	$m \leftarrow x \oplus \mathrm{RO}(k,\ell)$	$\quad T[z,\ell] \leftarrow\!\!\$\ \{0,1\}^{\ell}$
$h \leftarrow \mathrm{RO}(k \,\Vert\, m,\ell)$	$h' \leftarrow \mathrm{RO}(k \,\Vert\, m,\ell)$	Return $T[z,\ell]$
$c \leftarrow (x,h)$	If $h \neq h'$ then return $\perp$	
Return (k,c)	Else return m	

Fig. 9: Sample EMDK scheme $\mathsf{EMDK} = \mathsf{SIMPLE\text{-}EMDK}$ in the ROM.

$\underline{\mathsf{EMDK.Enc}(m)}$	$\underline{\mathsf{EMDK.Dec}(k,c_{se})}$
$r_0 \leftarrow\!\!\$\ \{0,1\}^{\mathsf{F.kl}}$; $r_1 \leftarrow \mathsf{F.Ev}(r_0,m)$	$m \leftarrow \mathsf{SE.Dec}(k,c_{se})$; If $m = \perp$ then return $\perp$
$k \leftarrow r_0 \,\Vert\, r_1$; $c_{se} \leftarrow\!\!\$\ \mathsf{SE.Enc}(k,m)$	$r_0 \leftarrow k[1 \ldots \mathsf{F.kl}]$; $r_1 \leftarrow k[\mathsf{F.kl}+1 \ldots \mathsf{SE.kl}]$
Return (k,c_{se})	If $r_1 \neq \mathsf{F.Ev}(r_0,m)$ then return $\perp$
	Return m

Fig. 10: iMessage-based EMDK scheme $\mathsf{EMDK} = \mathsf{IMSG\text{-}EMDK}[\mathsf{F},\mathsf{SE}]$.

of the keys) was produced during a call to oracle Enc that takes a message m as input to return the output (k,c) of running $\mathsf{EMDK.Enc}(m)$ with honestly generated random coins. The other key can be arbitrarly chosen by the adversary. In the symmetric encryption setting, a similar notion called *wrong-key detection* was previously defined by Canetti et al. [17]. The notion of robustness for public-key encryption was formalized by Abdalla et al. [1] and further extended by Farshim et al. [23].

Sample EMDK scheme SIMPLE-EMDK. It is easy to build an EMDK scheme that is both AE-secure and ROB-secure. One example of such scheme is the construction SIMPLE-EMDK in the random oracle model (ROM) that is defined in Fig. 9. In the next section we will define the EMDK scheme used iMessage; it looks convoluted, and its security is hard to prove even in the ideal models. In [13] we define the EMDK scheme that was initially used in iMessage; it was replaced with the current EMDK scheme in order to fix a security flaw in the iMessage design. We believe that the design of the currently used EMDK scheme was chosen based on a requirement to maintain backward-compatibility across the initial and the current versions of iMessage protocol.

4.2 iMessage-based EMDK scheme

In this section we define the EMDK scheme IMSG-EMDK that is used as the core building block in the construction of iMessage (we use it to specify the iMessage-based signcryption scheme in Section 5). We will provide reductions showing the AE-security and the ROB-security of IMSG-EMDK. These security reductions will first require us to introduce two new security notions for symmetric encryption schemes: *partial key recovery* and *weak robustness*.

Game $\mathrm{G}^{\mathsf{pkr}}_{\mathsf{SE},\ell,\mathcal{P}}$	Game $\mathrm{G}^{\mathsf{wrob}}_{\mathsf{SE},\ell,\mathcal{G}}$
$\mathcal{P}^{\textsc{Enc},\textsc{GuessKey}}$; Return win	$\mathcal{G}^{\textsc{Enc}}$; Return win
$\underline{\textsc{Enc}(m)}$	$\underline{\textsc{Enc}(r_0, m)}$
$k \leftarrow\!\!\$\ \{0,1\}^{\mathsf{SE.kl}}$; $c \leftarrow\!\!\$\ \mathsf{SE.Enc}(k, m)$ $n \leftarrow n + 1$; $\mathsf{k}[n] \leftarrow k[1 \ldots \ell]$; Return c	$r_1 \leftarrow\!\!\$\ \{0,1\}^{\ell}$; $k \leftarrow r_0 \,\|\, r_1$ $c \leftarrow \mathsf{SE.Enc}(k, m)$
$\underline{\textsc{GuessKey}(p)}$	If $\exists (m', c) \in W$: $m' \neq m$ then $\quad \mathsf{win} \leftarrow \mathsf{true}$
If $\exists i \in [n]$: $\mathsf{k}[i] = p$ then $\mathsf{win} \leftarrow \mathsf{true}$	$W \leftarrow W \cup \{(m, c)\}$; Return r_1

Fig. 11: Games defining partial key recovery security of symmetric encryption scheme SE with respect to prefix length ℓ, and weak robustness of deterministic symmetric encryption scheme SE with respect to randomized key-suffix length ℓ.

EMDK scheme IMSG-EMDK. Let SE be a symmetric encryption scheme. Let F be a function family with $\mathsf{F.In} = \{0,1\}^*$ such that $\mathsf{F.kl} + \mathsf{F.ol} = \mathsf{SE.kl}$. Then $\mathsf{EMDK} = \mathsf{IMSG\text{-}EMDK}[\mathsf{F}, \mathsf{SE}]$ is the EMDK scheme as defined in Fig. 10, with key length $\mathsf{EMDK.kl} = \mathsf{SE.kl}$.

Informally, the encryption algorithm $\mathsf{EMDK.Enc}(m)$ samples a hash function key r_0 and computes hash $r_1 \leftarrow\!\!\$\ \mathsf{F.Ev}(r_0, m)$. It then encrypts m by running $\mathsf{SE.Enc}(k, m)$, where $k = r_0 \,\|\, r_1$ is a message-derived key. The decryption algorithm splits k into r_0 and r_1 and – upon recovering m – checks that $r_1 = \mathsf{F.Ev}(r_0, m)$. In the iMessage construction, SE is instantiated with AES-CTR using 128-bit keys and a fixed IV=1, whereas F is instantiated with HMAC-SHA256 using $\mathsf{F.kl} = 88$ and $\mathsf{F.ol} = 40$.

Partial key recovery security of SE. Consider game $\mathrm{G}^{\mathsf{pkr}}$ of Fig. 11, associated to a symmetric encryption scheme SE, a prefix length $\ell \in \mathbb{N}$ and an adversary $\mathcal{P}$. The advantage of $\mathcal{P}$ in breaking the PKR-security of SE with respect to ℓ is defined as $\mathsf{Adv}^{\mathsf{pkr}}_{\mathsf{SE},\ell}(\mathcal{P}) = \Pr[\mathrm{G}^{\mathsf{pkr}}_{\mathsf{SE},\ell,\mathcal{P}}]$. The adversary $\mathcal{P}$ has access to oracle Enc that takes a message m and encrypts it under a uniformly random key k (independently sampled for each oracle call). The goal of the adversary is to recover the first ℓ bits of *any* secret key that was used in prior Enc queries.

Weak robustness of deterministic SE. Consider game $\mathrm{G}^{\mathsf{wrob}}$ of Fig. 11, associated to a deterministic symmetric encryption scheme SE, a randomized key-suffix length $\ell \in \mathbb{N}$, and an adversary $\mathcal{G}$. The advantage of $\mathcal{G}$ in breaking the WROB-security of SE with respect to ℓ is defined as $\mathsf{Adv}^{\mathsf{wrob}}_{\mathsf{SE},\ell}(\mathcal{G}) = \Pr[\mathrm{G}^{\mathsf{wrob}}_{\mathsf{SE},\ell,\mathcal{G}}]$. The adversary has access to oracle Enc. The oracle takes a prefix of an encryption key $r_0 \in \{0,1\}^{\mathsf{SE.kl}-\ell}$ and message m as input. It then randomly samples the suffix of the key $r_1 \in \{0,1\}^{\ell}$ and returns it to the adversary. The adversary wins if it succeeds to query Enc on some inputs (r_0, m) and (r_0', m') such that $m \neq m'$ yet the oracle mapped both queries to the same ciphertext c. In other words, the goal of the adversary is to find k_0, m_0, k_1, m_1 such that

$\mathsf{SE.Enc}(k_0, m_0) = \mathsf{SE.Enc}(k_1, m_1)$ and $m_0 \neq m_1$ (which also implies $k_0 \neq k_1$), and the adversary has only a partial control over the choice of k_0 and k_1. Note that this assumption can be validated in the ideal cipher model.

Security reductions for IMSG-EMDK. We now provide the reductions for AE-security and ROB-security of IMSG-EMDK. The former is split into Theorem 1 and Theorem 2, whereas the latter is provided in Theorem 3. Note that in [13] we provide the standard definitions for the random oracle model, the UNIQUE-security and the OTIND-security of symmetric encryption, and the TCR-security of function families. The proofs of Theorem 1, Theorem 2 and Theorem 3 are in the full version [13].

Theorem 1. *Let* SE *be a symmetric encryption scheme. Let* F *be a function family with* $\mathsf{F.In} = \{0,1\}^*$*, such that* $\mathsf{F.kl} + \mathsf{F.ol} = \mathsf{SE.kl}$*. Let* EMDK $=$ IMSG-EMDK$[\mathsf{F}, \mathsf{SE}]$*. Let* $\mathcal{D}_{\mathrm{AE}}$ *be an adversary against the* AE*-security of* EMDK*. Then we build an adversary* $\mathcal{U}$ *against the* UNIQUE*-security of* SE*, an adversary* $\mathcal{H}$ *against the* TCR*-security of* F*, and an adversary* $\mathcal{D}_{\mathrm{IND}}$ *against the* IND*-security of* EMDK *such that*

$$\mathsf{Adv}^{\mathsf{ae}}_{\mathsf{EMDK}}(\mathcal{D}_{\mathrm{AE}}) \leq 2 \cdot \mathsf{Adv}^{\mathsf{unique}}_{\mathsf{SE}}(\mathcal{U}) + 2 \cdot \mathsf{Adv}^{\mathsf{tcr}}_{\mathsf{F}}(\mathcal{H}) + \mathsf{Adv}^{\mathsf{ind}}_{\mathsf{EMDK}}(\mathcal{D}_{\mathrm{IND}}).$$

Theorem 2. *Let* SE *be a symmetric encryption scheme. Let* F *be a function family with* $\mathsf{F.In} = \{0,1\}^*$ *and* $\mathsf{F.kl} + \mathsf{F.ol} = \mathsf{SE.kl}$*, defined by* $\mathsf{F.Ev}^{\mathrm{RO}}(r, m) = \mathrm{RO}(\langle r, m\rangle, \mathsf{F.ol})$ *in the random oracle model. Let* EMDK $=$ IMSG-EMDK$[\mathsf{F}, \mathsf{SE}]$*. Let* $\mathcal{D}_{\mathsf{EMDK}}$ *be an adversary against the* IND*-security of* EMDK *that makes* q_{LR} *queries to its* LR *oracle and* q_{RO} *queries to random oracle* RO*. Then we build an adversary* $\mathcal{P}$ *against the* PKR*-security of* SE *with respect to* F.kl*, and an adversary* $\mathcal{D}_{\mathsf{SE}}$ *against the* OTIND*-security of* SE*, such that*

$$\mathsf{Adv}^{\mathsf{ind}}_{\mathsf{EMDK}}(\mathcal{D}_{\mathsf{EMDK}}) \leq 2 \cdot \gamma + 2 \cdot \mathsf{Adv}^{\mathsf{pkr}}_{\mathsf{SE},\mathsf{F.kl}}(\mathcal{P}) + \mathsf{Adv}^{\mathsf{otind}}_{\mathsf{SE}}(\mathcal{D}_{\mathsf{SE}}),$$

where

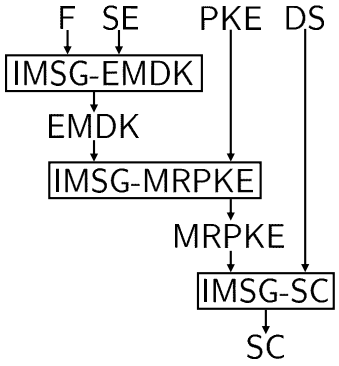
$$\gamma = \frac{(2 \cdot q_{\mathrm{RO}} + q_{\mathrm{LR}} - 1) \cdot q_{\mathrm{LR}}}{2^{\mathsf{F.kl}+1}}.$$

Theorem 3. *Let* SE *be a deterministic symmetric encryption scheme. Let* F *be a function family with* $\mathsf{F.In} = \{0,1\}^*$ *and* $\mathsf{F.kl} + \mathsf{F.ol} = \mathsf{SE.kl}$*, defined by* $\mathsf{F.Ev}^{\mathrm{RO}}(r, m) = \mathrm{RO}(\langle r, m\rangle, \mathsf{F.ol})$ *in the random oracle model. Let* EMDK $=$ IMSG-EMDK$[\mathsf{F}, \mathsf{SE}]$*. Let* $\mathcal{G}_{\mathsf{EMDK}}$ *be an adversary against the* ROB*-security of* EMDK*. Then we build an adversary* $\mathcal{U}$ *against the* UNIQUE*-security of* SE*, and an adversary* $\mathcal{G}_{\mathsf{SE}}$ *against the* WROB*-security of* SE *with respect to* F.ol *such that*

$$\mathsf{Adv}^{\mathsf{rob}}_{\mathsf{EMDK}}(\mathcal{G}_{\mathsf{EMDK}}) \leq \mathsf{Adv}^{\mathsf{unique}}_{\mathsf{SE}}(\mathcal{U}) + \mathsf{Adv}^{\mathsf{wrob}}_{\mathsf{SE},\mathsf{F.ol}}(\mathcal{G}_{\mathsf{SE}}).$$

5 Design and security of iMessage

In this section we define a signcryption scheme that models the current design of iMessage protocol for end-to-end encrypted messaging, and we analyze its security. All publicly available information about the iMessage protocol is provided by Apple in *iOS Security Guide* [4] that is regularly updated but is very limited



Scheme	Construction	Figure
EMDK	IMSG-EMDK[F, SE]	10
MRPKE	IMSG-MRPKE[EMDK, PKE]	14
SC	IMSG-SC[MRPKE, DS]	13

Scheme	Instantiation
F	HMAC-SHA256 (F.kl = 88, F.ol = 40)
SE	AES-CTR with 128-bit key and IV=1
PKE	RSA-OAEP with 1280-bit key
DS	ECDSA with NIST P-256 curve

Fig. 12: Modular design of iMessage-based signcryption scheme. The boxed nodes in the diagram denote transforms that build a new cryptographic scheme from two underlying primitives.

and vague. So in addition to the iOS Security Guide, we also reference work that attempted to reverse-engineer [34, 32] and attack [26] the prior versions of iMessage. A message-recovery attack against iMessage was previously found and implemented by Garman et al. [26] in 2016, and subsequently fixed by Apple starting from version 9.3 of iOS, and version 10.11.4 of Mac OS X. The implemented changes to the protocol prevented the attack, but also made the protocol design less intuitive. It appears that one of the goals of the updated protocol design was to preserve backward-compatibility, and that could be the reason why the current design is a lot more more sophisticated than otherwise necessary. Apple has not formalized any claims about the security achieved by the initial or the current iMessage protocol, or the assumptions that are required from the cryptographic primitives that serve as the building blocks. We fill in the gap by providing precise claims about the security of iMessage design when modeled by our signcryption scheme. In this section we focus only on the current protocol design of iMessage. In [13] we provide the design of the initial iMessage protocol, we explain the attack proposed by Garman et al. [26], and we introduce the goal of backward-compatibility for signcryption schemes.

5.1 iMessage-based signcryption scheme IMSG-SC

Identifying signcryption as the goal. The design of iMessage combines multiple cryptographic primitives to build an end-to-end encrypted messaging protocol. It uses HMAC-SHA256, AES-CTR, RSA-OAEP and ECDSA as the underlying primitives. Apple's *iOS Security Guide* [4] and prior work on reverse-engineering and analysis of iMessage [34, 32, 26] does not explicitly indicate what type of cryptographic scheme is built as the result of combining these primitives. We identify it as a signcryption scheme. We define the iMessage-based signcryp-

$\underline{\mathsf{SC.Setup}}$
$\pi \leftarrow \mathsf{MRPKE.Setup}$; Return π

$\underline{\mathsf{SC.SigEnc}(\pi, id_s, pk_s, sk_s, \mathcal{R}, m, ad)}$
$\mathcal{I} \leftarrow \emptyset$; $\mathcal{R}_{pke} \leftarrow \emptyset$; $\mathcal{C} \leftarrow \emptyset$
For each $(id_r, pk_r) \in \mathcal{R}$ do
  $(vk_r, ek_r) \leftarrow pk_r$
  $\mathcal{I} \leftarrow \mathcal{I} \cup \{id_r\}$
  $\mathcal{R}_{pke} \leftarrow \mathcal{R}_{pke} \cup \{(id_r, ek_r)\}$
$m_{pke} \leftarrow \langle m, id_s, \mathcal{I}\rangle$
$\mathcal{C}_{pke} \leftarrow\!\!\$\ \mathsf{MRPKE.Enc}(\pi, \mathcal{R}_{pke}, m_{pke})$
$(tk_s, dk_s) \leftarrow sk_s$
For each $(id_r, c_{pke}) \in \mathcal{C}_{pke}$ do
  $\sigma \leftarrow\!\!\$\ \mathsf{DS.Sig}(tk_s, \langle c_{pke}, ad\rangle)$
  $c \leftarrow (c_{pke}, \sigma)$; $\mathcal{C} \leftarrow \mathcal{C} \cup \{(id_r, c)\}$
Return $\mathcal{C}$

$\underline{\mathsf{SC.Kg}(\pi)}$
$(vk, tk) \leftarrow\!\!\$\ \mathsf{DS.Kg}$
$(ek, dk) \leftarrow\!\!\$\ \mathsf{MRPKE.Kg}(\pi)$
$pk \leftarrow (vk, ek)$; $sk \leftarrow (tk, dk)$
Return (pk, sk)

$\underline{\mathsf{SC.VerDec}(\pi, id_s, pk_s, id_r, pk_r, sk_r, c, ad)}$
$(c_{pke}, \sigma) \leftarrow c$; $(vk_s, ek_s) \leftarrow pk_s$
$(vk_r, ek_r) \leftarrow pk_r$; $(tk_r, dk_r) \leftarrow sk_r$
$d \leftarrow \mathsf{DS.Ver}(vk_s, \langle c_{pke}, ad\rangle, \sigma)$
If not d then return $\perp$
$m_{pke} \leftarrow \mathsf{MRPKE.Dec}(\pi, ek_r, dk_r, c_{pke})$
If $m_{pke} = \perp$ then return $\perp$
$\langle m, id_s^*, \mathcal{I}\rangle \leftarrow m_{pke}$
If $id_s \neq id_s^*$ or $id_r \notin \mathcal{I}$ then return $\perp$
Return m

Fig. 13: Signcryption scheme $\mathsf{SC} = \mathsf{IMSG\text{-}SC}[\mathsf{MRPKE}, \mathsf{DS}]$.

tion scheme IMSG-SC in a modular way that facilitates its security analysis. Fig. 12 shows the order in which the underlying primitives are combined to build IMSG-SC, while also providing intermediate constructions along the way. We now explain this step by step.

Modular design of IMSG-SC. Our construction starts from choosing a function family F and a symmetric encryption scheme SE (instantiated with HMAC-SHA256 and AES-CTR in iMessage). It combines them to build an encryption scheme under message derived keys $\mathsf{EMDK} = \mathsf{IMSG\text{-}EMDK}[\mathsf{F}, \mathsf{SE}]$. The resulting EMDK scheme is combined with public-key encryption scheme PKE (instantiated with RSA-OAEP in iMessage) to build a multi-recipient public-key encryption scheme $\mathsf{MRPKE} = \mathsf{IMSG\text{-}MRPKE}[\mathsf{EMDK}, \mathsf{PKE}]$ (syntax and correctness of MRPKE schemes is defined in [13]). Finally, MRPKE and digital signature scheme DS (instantiated with ECDSA in iMessage) are combined to build the iMessage-based signcryption scheme $\mathsf{SC} = \mathsf{IMSG\text{-}SC}[\mathsf{MRPKE}, \mathsf{DS}]$. The definition of IMSG-EMDK was provided in Section 4.2. We now define IMSG-SC and IMSG-MRPKE.

Signcryption scheme IMSG-SC. Let MRPKE be a multi-recipient public-key encryption scheme. Let DS be a digital signature scheme. Then $\mathsf{SC} = \mathsf{IMSG\text{-}SC}[\mathsf{MRPKE}, \mathsf{DS}]$ is the signcryption scheme as defined in Fig. 13, with $\mathsf{SC.ID} = \{0,1\}^*$. In order to produce a signcryption of message m with associated data ad, algorithm SC.SigEnc performs the following steps. It builds a new message $m_{pke} = \langle m, id_s, \mathcal{I}\rangle$ as the unique encoding of $m, id_s, \mathcal{I}$, where $\mathcal{I}$ is the set of recipients. It then calls MRPKE.Enc to encrypt the same message m_{pke} for every recipient. Algorithm MRPKE.Enc returns a set $\mathcal{C}_{pke}$ containing pairs (id_r, c_{pke}), each indicating that an MRPKE ciphertext c_{pke} was produced for recipient id_r. For each recipient, the corresponding ciphertext c_{pke} is then

$\underline{\mathsf{MRPKE.Setup}}$	$\underline{\mathsf{MRPKE.Kg}(\pi)}$
$\pi \leftarrow \varepsilon$; Return π	$(ek, dk) \leftarrow\!\!\$\ \mathsf{PKE.Kg}$; Return (ek, dk)
$\underline{\mathsf{MRPKE.Enc}(\pi, \mathcal{R}, m)}$	$\underline{\mathsf{MRPKE.Dec}(\pi, ek, dk, c)}$
$\mathcal{C} \leftarrow \emptyset$; $(k, c_{se}) \leftarrow\!\!\$\ \mathsf{EMDK.Enc}(m)$	$(c_{se}, c_{pke}) \leftarrow c$
For each $(id_r, ek_r) \in \mathcal{R}$ do	$k \leftarrow \mathsf{PKE.Dec}(ek, dk, c_{pke})$
$\quad c_{pke} \leftarrow\!\!\$\ \mathsf{PKE.Enc}(ek_r, k)$	If $k = \perp$ then return $\perp$
$\quad c \leftarrow (c_{se}, c_{pke})$; $\mathcal{C} \leftarrow \mathcal{C} \cup \{(id_r, c)\}$	$m \leftarrow \mathsf{EMDK.Dec}(k, c_{se})$
Return $\mathcal{C}$	Return m

Fig. 14: Multi-recipient public-key encryption scheme $\mathsf{MRPKE} = \mathsf{IMSG\text{-}MRPKE}[\mathsf{EMDK}, \mathsf{PKE}]$.

encoded with the associated data ad into $\langle c_{pke}, ad\rangle$ and signed using the signing key tk_s of sender identity id_s, producing a signature σ. The pair $(id_r, (c_{pke}, \sigma))$ is then added to the output set of algorithm $\mathsf{SC.SigEnc}$. When running the unsigncryption of ciphertext c sent from id_s to id_r, algorithm $\mathsf{SC.VerDec}$ ensures that the recovered MRPKE plaintext $m_{pke} = \langle m, id_s^*, \mathcal{I}\rangle$ is consistent with $id_s = id_s^*$ and $id_r \in \mathcal{I}$.

Multi-recipient public-key encryption scheme $\mathsf{IMSG\text{-}MRPKE}$. Let EMDK be an encryption scheme under message derived keys. Let PKE be a public-key encryption scheme with $\mathsf{PKE.In} = \{0,1\}^{\mathsf{EMDK.kl}}$. Then $\mathsf{MRPKE} = \mathsf{IMSG\text{-}MRPKE}[\mathsf{EMDK}, \mathsf{PKE}]$ is the multi-recipient public-key encryption scheme as defined in Fig. 14. Algorithm $\mathsf{MRPKE.Enc}$ first runs $(k, c_{se}) \leftarrow\!\!\$\ \mathsf{EMDK.Enc}(m)$ to produce an EMDK ciphertext c_{se} that encrypts m under key k. The obtained key k is then independently encrypted for each recipient identity id_r using its PKE encryption key ek_r, and the corresponding tuple $(id_r, (c_{se}, c_{pke}))$ is added to the output set of algorithm $\mathsf{MRPKE.Enc}$.

Combining everything together. Let SC be the iMessage-based signcryption scheme that is produced by combining all of the underlying primitives described above. Then the data flow within the fully expanded algorithms $\mathsf{SC.SigEnc}$ and $\mathsf{SC.VerDec}$ is schematically displayed in Fig. 15. For simplicity, the diagrams show the case when a message m is sent to a single recipient id_r.

5.2 Parameter-choice induced attacks on privacy of iMessage

The iMessage-based signcryption scheme SC uses the EMDK scheme $\mathsf{EMDK} = \mathsf{IMSG\text{-}EMDK}[\mathsf{F}, \mathsf{SE}]$ as one of its underlying primitives. Recall that in order to encrypt a payload $m' = \langle m, id_s, \mathcal{I}\rangle$, the EMDK scheme samples a function key $r_0 \leftarrow\!\!\$\ \{0,1\}^{\mathsf{F.kl}}$, computes a hash of m' as $r_1 \leftarrow \mathsf{F.Ev}(r_0, m')$, sets the encryption key $k \leftarrow r_0 \,\|\, r_1$, and produces a ciphertext as $c_{se} \leftarrow\!\!\$\ \mathsf{SE.Enc}(k, m')$. The implementation of iMessage uses parameters $\mathsf{F.kl} = 88$ and $\mathsf{F.ol} = 40$. In this section we provide three adversaries against the privacy of SC whose success depends on the choice of $\mathsf{F.kl}$ and $\mathsf{F.ol}$. In next sections we will provide security proofs for SC.

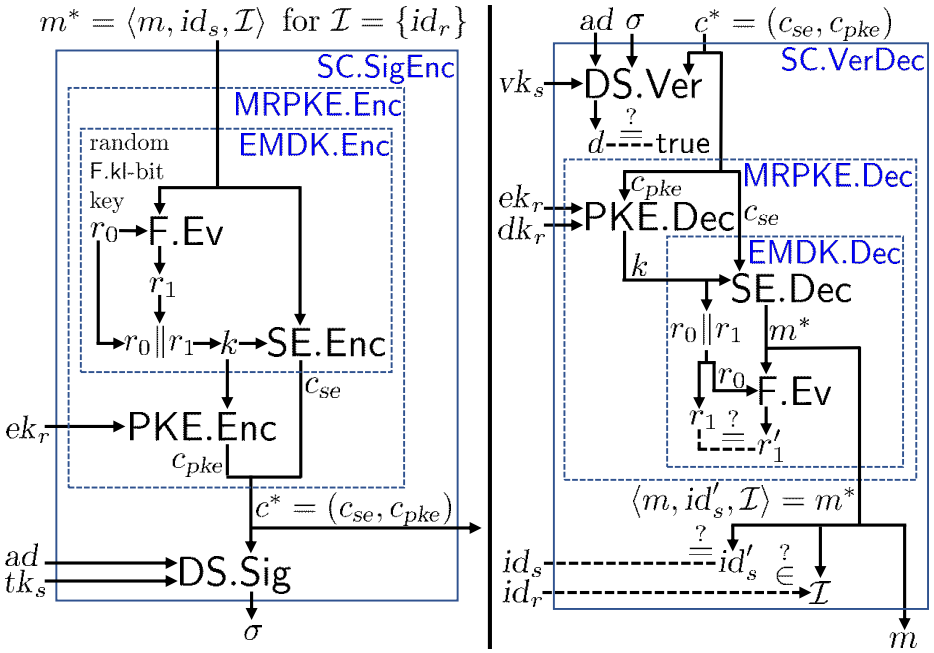


Fig. 15: Algorithms SC.SigEnc (left panel) and SC.VerDec (right panel) for SC = IMSG-SC[MRPKE, DS], where MRPKE = IMSG-MRPKE[EMDK, PKE] and EMDK = IMSG-EMDK[F, SE]. For simplicity, we let id_r be the only recipient, and we do not show how to parse inputs and combine outputs for the displayed algorithms. The dotted lines inside SC.VerDec denote equality check, and the dotted arrow denotes membership check.

We will show that each adversary in this section arises from an attack against a different step in our security proofs. We will be able to conclude that these are roughly the best attacks that arise from the choice of EMDK parameters. We will also explain why it is hard to construct any adversaries against the authenticity of SC. Now consider the adversaries of Fig. 16. The full version of this paper [13] provides a detailed explanation for each adversary.

Formal claims and analysis. We provide the number of queries, the runtime complexity and the advantage of each adversary in Fig. 17. The assumptions necessary to prove the advantage are stated in Lemma 4 below. Note that $\mathcal{D}_{\mathsf{birthday}}$ represents a purely theoretical attack, but both $\mathcal{D}_{\mathsf{exhaustive}}$ and $\mathcal{D}_{\mathsf{ADR02}}$ can lead to practical message-recovery attacks (the latter used by Garman et al. [26]).

Let EMDK = IMSG-EMDK[F, SE]. Adversary $\mathcal{D}_{\mathsf{ADR02}}$ shows that EMDK can have at most F.ol bits of security with respect to PRIV, and adversary $\mathcal{D}_{\mathsf{birthday}}$ shows that EMDK can have at most $\approx \mathsf{F.kl}/2 + \log_2 \mathsf{F.kl}$ bits of security with respect to PRIV. It follows that setting $\mathsf{F.ol} \approx \mathsf{F.kl}/2$ is a good initial guideline, and roughly corresponds to the parameter choices made in iMessage. We will

$\underline{\mathcal{D}^{\text{NewH,NewC,Exp,LR,VerDec}}_{\mathsf{exhaustive},n}(\pi)}$
$id_s \leftarrow$ "send" ; $pk_s \leftarrow\!\!\$\ \text{NewH}(id_s)$
$id_r \leftarrow$ "recv" ; $pk_r \leftarrow\!\!\$\ \text{NewH}(id_r)$
$\mathcal{I} \leftarrow \{id_r\}$; $ad \leftarrow \varepsilon$
$m_0 \leftarrow 0^n$; $m_1 \leftarrow\!\!\$\ \{0,1\}^n$
$\mathcal{C} \leftarrow\!\!\$\ \text{LR}(id_s, \mathcal{I}, m_0, m_1, ad)$
$\{(id_r, c)\} \leftarrow \mathcal{C}$; $((c_{se}, c_{pke}), \sigma) \leftarrow c$
$m'_1 \leftarrow \langle m_1, id_s, \mathcal{I}\rangle$
For each $r_0 \in \{0,1\}^{\mathsf{F.kl}}$ do
  $r_1 \leftarrow \mathsf{F.Ev}(r_0, m'_1)$; $k \leftarrow r_0 \,\|\, r_1$
  If $\mathsf{SE.Dec}(k, c_{se}) = m'_1$ then return 1
Return 0

$\underline{\mathcal{D}^{\text{NewH,NewC,Exp,LR,VerDec}}_{\mathsf{birthday}}(\pi)}$
$id_s \leftarrow$ "send" ; $pk_s \leftarrow\!\!\$\ \text{NewH}(id_s)$
$id_r \leftarrow$ "recv" ; $pk_r \leftarrow\!\!\$\ \text{NewH}(id_r)$
$\mathcal{I} \leftarrow \{id_r\}$; $ad \leftarrow \varepsilon$
$S \leftarrow \emptyset$; $p \leftarrow \lceil \mathsf{F.kl}/2 \rceil$; $m_1 \leftarrow 0^p$
For each $m_0 \in \{0,1\}^p$ do
  $\mathcal{C} \leftarrow\!\!\$\ \text{LR}(id_s, \mathcal{I}, m_0, m_1, ad)$
  $\{(id_r, c)\} \leftarrow \mathcal{C}$; $((c_{se}, c_{pke}), \sigma) \leftarrow c$
  If $c_{se} \in S$ then return 1
  $S \leftarrow S \cup \{c_{se}\}$
Return 0

$\underline{\mathcal{D}^{\text{NewH,NewC,Exp,LR,VerDec}}_{\mathsf{ADR02}}(\pi)}$
$id_s \leftarrow 0^{128}$; $pk_s \leftarrow\!\!\$\ \text{NewH}(id_s)$; $id_r \leftarrow 1^{128}$; $pk_r \leftarrow\!\!\$\ \text{NewH}(id_r)$
$\mathcal{I} \leftarrow \{id_r\}$; $m_0 \leftarrow 0^{128}$; $m_1 \leftarrow 1^{128}$; $ad \leftarrow \varepsilon$
$\mathcal{C} \leftarrow\!\!\$\ \text{LR}(id_s, \mathcal{I}, m_0, m_1, ad)$; $\{(id_r, c)\} \leftarrow \mathcal{C}$; $((c_{se}, c_{pke}), \sigma) \leftarrow c$
$id_c \leftarrow 0^{64}1^{64}$; $(pk_c, sk_c) \leftarrow\!\!\$\ \mathsf{SC.Kg}(\pi)$; $\text{NewC}(id_c, pk_c, sk_c)$; $(tk_c, dk_c) \leftarrow sk_c$
$m'_1 \leftarrow \langle m_1, id_s, \{id_r\}\rangle$; $m''_1 \leftarrow \langle m_1, id_c, \{id_r\}\rangle$; $c'_{se} \leftarrow c_{se} \oplus (m'_1 \oplus m''_1)$
$\sigma' \leftarrow\!\!\$\ \mathsf{DS.Sig}(tk_c, \langle (c'_{se}, c_{pke}), ad\rangle)$; $c' \leftarrow ((c'_{se}, c_{pke}), \sigma')$
$(m, \mathsf{err}) \leftarrow \text{VerDec}(id_c, id_r, c', ad)$; If $m = m_1$ then return 1 else return 0

Fig. 16: Adversaries $\mathcal{D}_{\mathsf{exhaustive},n}$, $\mathcal{D}_{\mathsf{birthday}}$ and $\mathcal{D}_{\mathsf{ADR02}}$ against the PRIV-security of $\mathsf{SC} = \mathsf{IMSG\text{-}SC}[\mathsf{MRPKE}, \mathsf{DS}]$, where $\mathsf{MRPKE} = \mathsf{IMSG\text{-}MRPKE}[\mathsf{EMDK}, \mathsf{PKE}]$ and $\mathsf{EMDK} = \mathsf{IMSG\text{-}EMDK}[\mathsf{F}, \mathsf{SE}]$. Adversary $\mathcal{D}_{\mathsf{ADR02}}$ requires that SE is AES-CTR with a fixed IV.

Adversary	q_{LR}	q_{NewC}	q_{VerDec}	Runtime complexity	Advantage
$\mathcal{D}_{\mathsf{exhaustive},n}$	1	0	0	$2^{\mathsf{F.kl}}$ evaluations of $\mathsf{F.Ev}$, $\mathsf{SE.Enc}$	$\geq 1 - 2^{\mathsf{SE.kl}-n}$
$\mathcal{D}_{\mathsf{birthday}}$	$2^{\lceil \mathsf{F.kl}/2 \rceil}$	0	0	$2^p \cdot p$ for $p = \lceil \mathsf{F.kl}/2 \rceil$	$> 1/8 - 2^{\mathsf{F.kl}-128}$
$\mathcal{D}_{\mathsf{ADR02}}$	1	1	1	1 evaluation of $\mathsf{SC.Kg}$, $\mathsf{DS.Sig}$	$= 2^{-\mathsf{F.ol}}$

Fig. 17: The resources used by adversaries $\mathcal{D}_{\mathsf{exhaustive},n}$, $\mathcal{D}_{\mathsf{birthday}}$ and $\mathcal{D}_{\mathsf{ADR02}}$, and the advantage achieved by each of them. Columns labeled q_{O} denote the number of queries an adversary makes to oracle O. All adversaries make 2 queries to oracle NewH, and 0 queries to oracle Exp. See Lemma 4 for necessary assumptions.

provide a more detailed analysis in Section 5.5. The proof of Lemma 4 is in the full version [13].

Lemma 4. *Let* SE *be a symmetric encryption scheme. Let* F *be a function family with* $\mathsf{F.In} = \{0,1\}^*$ *such that* $\mathsf{F.kl} + \mathsf{F.ol} = \mathsf{SE.kl}$. *Let* $\mathsf{EMDK} = \mathsf{IMSG\text{-}EMDK}[\mathsf{F}, \mathsf{SE}]$. *Let* PKE *be a public-key encryption scheme with* $\mathsf{PKE.In} = \{0,1\}^{\mathsf{SE.kl}}$. *Let* $\mathsf{MRPKE} = \mathsf{IMSG\text{-}MRPKE}[\mathsf{EMDK}, \mathsf{PKE}]$. *Let* DS *be a digital signature scheme. Let* $\mathsf{SC} = \mathsf{IMSG\text{-}SC}[\mathsf{MRPKE}, \mathsf{DS}]$. *Let* $\mathsf{R} \subseteq \{0,1\}^* \times \{0,1\}^*$ *be*

any relaxing relation. Then for any $n > \mathsf{SE.kl}$,

$$\mathsf{Adv}^{\mathsf{priv}}_{\mathsf{SC,R}}(\mathcal{D}_{\mathsf{exhaustive},n}) \geq 1 - 2^{\mathsf{SE.kl}-n}.$$

Furthermore, for any $1 \leq \mathsf{F.kl} \leq 124$, *if* SE *is AES-CTR with a fixed IV, and if AES is modeled as the ideal cipher, then*

$$\mathsf{Adv}^{\mathsf{priv}}_{\mathsf{SC,R}}(\mathcal{D}_{\mathsf{birthday}}) > 1/8 - 2^{\mathsf{F.kl}-128}.$$

Let $\mathsf{R_m}$ *be the relaxing relation defined in Fig. 4. If* SE *is AES-CTR with a fixed IV, and if* F *is defined as* $\mathsf{F.Ev}^{\mathrm{RO}}(r, m) = \mathrm{RO}(\langle r, m\rangle, \mathsf{F.ol})$ *in the random oracle model, then*

$$\mathsf{Adv}^{\mathsf{priv}}_{\mathsf{SC,R_m}}(\mathcal{D}_{\mathsf{ADR02}}) = 2^{-\mathsf{F.ol}}.$$

5.3 Authenticity of iMessage

In this section we reduce the authenticity of the iMessage-based signcryption scheme SC to the security of its underlying primitives. First we reduce the authenticity of SC = IMSG-SC[MRPKE, DS] to the unforgeability of DS and to the robustness of MRPKE. And then we reduce the robustness of MRPKE = IMSG-MRPKE[EMDK, PKE] to the robustness of *either* PKE or EMDK; it is sufficient that only one of the two is robust.

Reduction showing authenticity of IMSG-SC. Recall that an SC ciphertext is a pair (c_{pke}, σ) that consists of an MRPKE ciphertext c_{pke} (encrypting some $\langle m, id_s, \mathcal{I}\rangle$) and a DS signature σ of $\langle c_{pke}, ad\rangle$. Intuitively, the authenticity of SC requires some type of unforgeability from DS in order to prevent the adversary from producing a valid signature on arbitrary c_{pke} and ad of its own choice. However, the unforgeability of DS is not a sufficient condition, because the adversary is allowed to win the game $\mathrm{G}^{\mathsf{auth}}$ by forging an SC ciphertext for a corrupted recipient identity that uses maliciously chosen SC keys. So an additional requirement is that the adversary should not be able to find an SC key pair (pk, sk) that succesfully decrypts an honestly produced SC ciphertext (c_{pke}, σ) to an *unintended* message. To ensure this, we require that MRPKE is robust (as defined in the full version of this paper [13]). Note that finding a new key pair that decrypts the ciphertext to the *original* message will not help the adversary to win the game because then the decryption will fail by not finding the corrupted recipient's identity in recipient set $\mathcal{I}$.

We define unforgeability UF of a digital signature scheme with respect to a relaxing relation R, such that the standard unforgeability is captured with respect to $\mathsf{R_m}$ and the strong unforgeability is captured with respect to $\mathsf{R_{id}}$. The formal definition is in the full version [13]. We show that if DS is UF-secure with respect to a relaxing relation $\mathsf{R}^* \in \{\mathsf{R_m}, \mathsf{R_{id}}\}$ then SC is AUTH-secure with respect to the corresponding parameterized relaxing relation IMSG-AUTH-REL[R^*], which we define below. ECDSA signatures are not strongly unforgeable [25], so iMessage is AUTH-secure with respect to IMSG-AUTH-REL[$\mathsf{R_m}$].

Relaxing relation IMSG-AUTH-REL. Let $\mathsf{R_m}$ and $\mathsf{R_{id}}$ be the relaxing relations defined in Section 3. Let $\mathsf{R}^* \in \{\mathsf{R_m}, \mathsf{R_{id}}\}$. Then IMSG-AUTH-REL[$\mathsf{R}^*$] is the

$\underline{\mathsf{IMSG\text{-}AUTH\text{-}REL}[\mathsf{R}^*].\mathsf{Vf}(z, z^*)}$
$((id_s, id_r, m, ad), (c_{pke}, \sigma)) \leftarrow z\ ;\ z_0 \leftarrow ((id_s, id_r, m, ad, c_{pke}), \sigma)$
$((id_s^*, id_r^*, m^*, ad^*), (c_{pke}^*, \sigma^*)) \leftarrow z^*\ ;\ z_1 \leftarrow ((id_s^*, id_r^*, m^*, ad^*, c_{pke}^*), \sigma^*)$
Return $\mathsf{R}^*.\mathsf{Vf}(z_0, z_1)$

Fig. 18: Relaxing relation $\mathsf{IMSG\text{-}AUTH\text{-}REL}[\mathsf{R}^*]$.

relaxing relation as defined in Fig. 18. Note that

$$\mathsf{R_{id}} = \mathsf{IMSG\text{-}AUTH\text{-}REL}[\mathsf{R_{id}}] \subset \mathsf{IMSG\text{-}AUTH\text{-}REL}[\mathsf{R_m}] \subset \mathsf{R_m},$$

where AUTH-security with respect to $\mathsf{R_{id}}$ captures the stronger security definition due to imposing the least number of restrictions regarding which queries are permitted to oracle VerDec. Relaxing relation $\mathsf{IMSG\text{-}AUTH\text{-}REL}[\mathsf{R_m}]$ does not allow adversary to win the authenticity game by only mauling the signature σ and not changing anything else.

Theorem 5. *Let* MRPKE *be a multi-recipient public-key encryption scheme. Let* DS *be a digital signature scheme. Let* $\mathsf{SC} = \mathsf{IMSG\text{-}SC}[\mathsf{MRPKE}, \mathsf{DS}]$. *Let* $\mathsf{R}^* \in \{\mathsf{R_m}, \mathsf{R_{id}}\}$. *Let* $\mathcal{F}_{\mathsf{SC}}$ *be an adversary against the* AUTH*-security of* SC *with respect to relaxing relation* $\mathsf{R} = \mathsf{IMSG\text{-}AUTH\text{-}REL}[\mathsf{R}^*]$. *Then we build an adversary* $\mathcal{F}_{\mathsf{DS}}$ *against the* UF*-security of* DS *with respect to* R^*, *and an adversary* $\mathcal{G}$ *against the* ROB*-security of* MRPKE *such that*

$$\mathsf{Adv}^{\mathsf{auth}}_{\mathsf{SC},\mathsf{R}}(\mathcal{F}_{\mathsf{SC}}) \leq \mathsf{Adv}^{\mathsf{uf}}_{\mathsf{DS},\mathsf{R}^*}(\mathcal{F}_{\mathsf{DS}}) + \mathsf{Adv}^{\mathsf{rob}}_{\mathsf{MRPKE}}(\mathcal{G}).$$

The proof of Theorem 5 is in the full version [13].

Reduction showing robustness of MRPKE. The ciphertext of $\mathsf{MRPKE} = \mathsf{IMSG\text{-}MRPKE}[\mathsf{EMDK}, \mathsf{PKE}]$ is a pair (c_{se}, c_{pke}), where c_{se} is an EMDK ciphertext encrypting some $m^* = \langle m, id_s, \mathcal{I} \rangle$, and c_{pke} is a PKE ciphertext encrypting the corresponding EMDK key k. The decryption algorithm of MRPKE first uses the PKE key pair (ek, dk) to decrypt c_{pke}, and then uses the recovered EMDK key k to decrypt c_{se}. We show that just one of PKE and EMDK being robust implies that MRPKE is also robust. Our definition of robustness for public-key encryption requires that it is hard to find a key pair (ek, dk) that decrypts an honestly produced ciphertext to a plaintext that is different from the originally encrypted message. If this condition holds for PKE, then clearly MRPKE is robust regardless of whether EMDK is robust. On the other hand, if PKE is not robust, then the robustness of EMDK (as defined in Section 4) would guarantee that the adversary is unlikely to decrypt c_{se} to a message other than m^* even if it has full control over the choice of EMDK key k. It is not known whether RSA-OAEP is robust, so our concrete security analysis of iMessage in Section 5.5 will rely entirely on the robustness of $\mathsf{EMDK} = \mathsf{IMSG\text{-}EMDK}$. The formal definition of robustness for PKE and the proof of Theorem 6 are in the full version [13].

Theorem 6. *Let* EMDK *be an encryption scheme under message derived keys. Let* PKE *be a public-key encryption scheme with* $\mathsf{PKE.In} = \{0,1\}^{\mathsf{EMDK.kl}}$. *Let*

$\underline{\mathsf{IMSG\text{-}PRIV\text{-}REL.Vf}(z, z^*)}$
$((id_s, id_r, m, ad), (c_{pke}, \sigma)) \leftarrow z\ ;\ ((id_s^*, id_r^*, m^*, ad^*), (c_{pke}^*, \sigma^*)) \leftarrow z^*$
Return $(id_s, id_r, m, c_{pke}) = (id_s^*, id_r^*, m^*, c_{pke}^*)$

Fig. 19: Relaxing relation IMSG-PRIV-REL.

$\mathsf{MRPKE} = \mathsf{IMSG\text{-}MRPKE}[\mathsf{EMDK}, \mathsf{PKE}]$. *Let* $\mathcal{G}_{\mathsf{MRPKE}}$ *be an adversary against the* ROB*-security of* MRPKE*. Then we build an adversary* $\mathcal{G}_{\mathsf{EMDK}}$ *against the* ROB*-security of* EMDK *such that*

$$\mathsf{Adv}^{\mathsf{rob}}_{\mathsf{MRPKE}}(\mathcal{G}_{\mathsf{MRPKE}}) \leq \mathsf{Adv}^{\mathsf{rob}}_{\mathsf{EMDK}}(\mathcal{G}_{\mathsf{EMDK}}),$$

and an adversary $\mathcal{G}_{\mathsf{PKE}}$ *against the* ROB*-security of* PKE *such that*

$$\mathsf{Adv}^{\mathsf{rob}}_{\mathsf{MRPKE}}(\mathcal{G}_{\mathsf{MRPKE}}) \leq \mathsf{Adv}^{\mathsf{rob}}_{\mathsf{PKE}}(\mathcal{G}_{\mathsf{PKE}}).$$

5.4 Privacy of iMessage

In this section we reduce the PRIV-security of $\mathsf{SC} = \mathsf{IMSG\text{-}SC}[\mathsf{MRPKE}, \mathsf{DS}]$ to the INDCCA-security of MRPKE, then reduce the INDCCA-security of $\mathsf{MRPKE} = \mathsf{IMSG\text{-}MRPKE}[\mathsf{EMDK}, \mathsf{PKE}]$ to the AE-security of EMDK and the INDCCA-security of PKE. The reductions are straightforward.

An adversary attacking the PRIV-security of SC is allowed to query oracle LR and get a challenge ciphertext from an exposed sender as long as the recipient is honest. This means that the adversary can use the sender's DS signing key to arbitrarily change associated data ad and signature σ of any challenge ciphertext prior to querying it to oracle VerDec. Our security reduction for PRIV-security of SC will be with respect to a relation that prohibits the adversary from trivially winning this way. Note that if IMSG-SC was defined to instead put ad inside $\langle m, id_s, \mathcal{I}\rangle$, then our security reduction would be able to show the PRIV-security of SC with respect to R_{id} assuming DS had *unique* signatures. However, ECDSA does not have this property (for the same reason it is not strongly unforgeable, as explained in [25]).

Relaxing relation IMSG-PRIV-REL. Let IMSG-PRIV-REL be the relaxing relation defined in Fig. 19. It first discards the associated data ad and the signature σ, and then compares the resulting tuples against each other. This reflects the intuition that an adversary can trivially change the values of ad and σ in any challenge ciphertext when attacking the PRIV-security of IMSG-SC.

Theorem 7. *Let* MRPKE *be a multi-recipient public-key encryption scheme. Let* DS *be a digital signature scheme. Let* $\mathsf{SC} = \mathsf{IMSG\text{-}SC}[\mathsf{MRPKE}, \mathsf{DS}]$*. Let* $\mathcal{D}_{\mathsf{SC}}$ *be an adversary against the* PRIV*-security of* SC *with respect to the relaxing relation* $\mathsf{R} = \mathsf{IMSG\text{-}PRIV\text{-}REL}$*. Then we build an adversary* $\mathcal{D}_{\mathsf{MRPKE}}$ *against the* INDCCA*-security of* MRPKE *such that*

$$\mathsf{Adv}^{\mathsf{priv}}_{\mathsf{SC},\mathsf{R}}(\mathcal{D}_{\mathsf{SC}}) \leq \mathsf{Adv}^{\mathsf{indcca}}_{\mathsf{MRPKE}}(\mathcal{D}_{\mathsf{MRPKE}}).$$

Theorem 8. *Let* EMDK *be an encryption scheme under message derived keys. Let* PKE *be a public-key encryption scheme with input set* $\mathsf{PKE.In} = \{0,1\}^{\mathsf{EMDK.kl}}$*. Let* $\mathsf{MRPKE} = \mathsf{IMSG\text{-}MRPKE}[\mathsf{EMDK}, \mathsf{PKE}]$*. Let* $\mathcal{D}_{\mathsf{MRPKE}}$ *be an adversary against the* INDCCA*-security of* MRPKE*. Then we build an adversary* $\mathcal{D}_{\mathsf{PKE}}$ *against the* INDCCA*-security of* PKE*, and an adversary* $\mathcal{D}_{\mathsf{EMDK}}$ *against the* AE*-security of* EMDK *such that*

$$\mathsf{Adv}^{\mathsf{indcca}}_{\mathsf{MRPKE}}(\mathcal{D}_{\mathsf{MRPKE}}) \leq 2 \cdot \mathsf{Adv}^{\mathsf{indcca}}_{\mathsf{PKE}}(\mathcal{D}_{\mathsf{PKE}}) + \mathsf{Adv}^{\mathsf{ae}}_{\mathsf{EMDK}}(\mathcal{D}_{\mathsf{EMDK}}).$$

The proofs of Theorem 7 and Theorem 8 are in the full version [13].

5.5 Concrete security of iMessage

In this section we summarize the results concerning the security of our iMessage-based signcryption scheme. For simplicity, we use the constructions and primitives from all across our work without formally redefining each of them.

Corollary for abstract schemes. Let SC be the iMessage-based signcryption scheme, defined based on the appropriate underlying primitives. Let $\mathsf{R}_{\mathsf{auth}} = \mathsf{IMSG\text{-}AUTH\text{-}REL}[\mathsf{R}^*]$ and $\mathsf{R}_{\mathsf{priv}} = \mathsf{IMSG\text{-}PRIV\text{-}REL}$. Then for any adversary $\mathcal{F}_{\mathsf{SC}}$ attacking the AUTH-security of SC we can build new adversaries such that:

$$\mathsf{Adv}^{\mathsf{auth}}_{\mathsf{SC},\mathsf{R}_{\mathsf{auth}}}(\mathcal{F}_{\mathsf{SC}}) \leq \mathsf{Adv}^{\mathsf{uf}}_{\mathsf{DS},\mathsf{R}^*}(\mathcal{F}_{\mathsf{DS}}) + \min(\mathsf{Adv}^{\mathsf{rob}}_{\mathsf{PKE}}(\mathcal{G}_{\mathsf{PKE}}), \alpha),$$

where

$$\alpha = \mathsf{Adv}^{\mathsf{unique}}_{\mathsf{SE}}(\mathcal{U}_0) + \mathsf{Adv}^{\mathsf{wrob}}_{\mathsf{SE},\mathsf{F.ol}}(\mathcal{G}_{\mathsf{SE}}).$$

For any adversary $\mathcal{D}_{\mathsf{SC}}$ attacking the PRIV-security of SC, making q_{LR} queries to LR oracle and q_{RO} queries to RO oracle, we build new adversaries such that:

$$\mathsf{Adv}^{\mathsf{priv}}_{\mathsf{SC},\mathsf{R}_{\mathsf{priv}}}(\mathcal{D}_{\mathsf{SC}}) \leq 2 \cdot (\beta + \gamma) + \mathsf{Adv}^{\mathsf{otind}}_{\mathsf{SE}}(\mathcal{D}_{\mathsf{SE}}),$$

where

$$\beta = \mathsf{Adv}^{\mathsf{indcca}}_{\mathsf{PKE}}(\mathcal{D}_{\mathsf{PKE}}) + \mathsf{Adv}^{\mathsf{unique}}_{\mathsf{SE}}(\mathcal{U}_1) + \mathsf{Adv}^{\mathsf{tcr}}_{\mathsf{F}}(\mathcal{H}) + \mathsf{Adv}^{\mathsf{pkr}}_{\mathsf{SE},\mathsf{F.kl}}(\mathcal{P}),$$

$$\gamma = \frac{(2 \cdot q_{\mathrm{RO}} + q_{\mathrm{LR}} - 1) \cdot q_{\mathrm{LR}}}{2^{\mathsf{F.kl}+1}}.$$

Bit-security of iMessage. We now assess the concrete security of iMessage when the abstract schemes that constitute SC are instantiated with real-world primitives. First, note that $\mathsf{Adv}^{\mathsf{unique}}_{\mathsf{SE}}(\mathcal{U}) = 0$ for any $\mathcal{U}$ when SE is AES-CTR. We will approximate the bit-security of SC based on the other terms above.

We assume that ECDSA with 256-bit keys (on the NIST P-256 curve) has 128 bits of UF-security with respect to R_{m} [21, 5]. We assume that RSA-OAEP with 1280-bit keys has 80 bits of INDCCA-security [21, 30]. SE is AES-CTR with key length $\mathsf{SE.kl}$; we assume that SE has $\mathsf{SE.kl}$ bits of OTIND-security.

For every other term used above, we approximate the corresponding bit-security based on the advantage ϵ and the runtime T of the best adversary we can come up with. For simplicity, we model F as the random oracle and we model SE as the ideal cipher. This simplifies the task of finding the "best possible" adversary against each security notion and then calculating its advantage. In each

SE.kl	F.kl	F.ol	PRIV bit-security	AUTH bit-security
128	88	40	39	71
	80	48	45	
	72	56	41	
192	128	64	63	
	120	72	66	
	112	80	62	
256	168	88	79	
	160	96	79	

Fig. 20: Lower bounds for bit-security of SC across different parameter choices.

case we consider either a constant-time adversary making a single guess in its security game (achieving some advantage ϵ in time $T \approx 1$), or an adversary that runs a birthday attack (achieving advantage $\epsilon \geq 0.3 \cdot \frac{q \cdot (q-1)}{N}$ in time $T \approx q \cdot \log_2 q$ for $q = \sqrt{2N}$). We use the following adversaries:

(i) Assume SE is AES-CTR where AES modeled as the ideal cipher with block length 128. In game $\mathrm{G}^{\mathsf{wrob}}_{\mathsf{SE},\mathsf{F.ol},\mathcal{G}}$ consider an adversary $\mathcal{G}$ that repeatedly queries its oracle Enc on inputs (r_0, m) where all $r_0 \in \{0,1\}^{\mathsf{F.kl}}$ are distinct and all $m \in \{0,1\}^{128}$ are distinct. The adversary wins if a collision occurs across the 128-bit outputs of $\mathsf{SE.Enc}$. Then $\epsilon = \mathsf{Adv}^{\mathsf{wrob}}_{\mathsf{SE},\mathsf{F.ol}}(\mathcal{G}_{\mathsf{SE}}) \geq 0.3 \cdot \frac{q_{\mathrm{Enc}} * (q_{\mathrm{Enc}}-1)}{2^{128}}$ and $T = q_{\mathrm{Enc}} \cdot \log_2 q_{\mathrm{Enc}}$ for $q_{\mathrm{Enc}} = \sqrt{2^{128+1}}$.

(ii) In game $\mathrm{G}^{\mathsf{tcr}}_{\mathsf{F},\mathcal{H}}$ consider an adversary $\mathcal{H}$ that queries its oracle NewKey(x_0) for any $x_0 \in \{0,1\}^*$ and then makes a guess $(1, x_1)$ for any $x_0 \neq x_1$. Then $\epsilon = \mathsf{Adv}^{\mathsf{tcr}}_{\mathsf{F}}(\mathcal{H}) = 2^{-\mathsf{F.ol}}$ and $T \approx 1$ in the random oracle model.

(iii) In game $\mathrm{G}^{\mathsf{pkr}}_{\mathsf{SE},\mathsf{F.kl},\mathcal{P}}$ consider an adversary $\mathcal{P}$ that makes a single call to Enc and then randomly guesses any key prefix $p \in \{0,1\}^{\mathsf{F.kl}}$. Then $\epsilon = \mathsf{Adv}^{\mathsf{pkr}}_{\mathsf{SE},\mathsf{F.kl}}(\mathcal{P}) = 2^{-\mathsf{F.kl}}$ and $T \approx 1$ in the ideal cipher model.

(iv) The term γ upper bounds the probability of an adversary finding a collision when running the birthday attack (in the random oracle model). The corresponding lower bound (for $q_{\mathrm{RO}} = 0$) is $\epsilon \geq 0.3 \cdot \frac{q_{\mathrm{LR}} \cdot (q_{\mathrm{LR}}-1)}{2^{\mathsf{F.kl}}}$ with $T = q_{\mathrm{LR}} \cdot \log_2 q_{\mathrm{LR}}$ and $q_{\mathrm{LR}} = \sqrt{2^{\mathsf{F.kl}+1}}$.

We wrote a script that combines all of the above to find the lower bound for the bit-security of SC (with respect to PRIV and AUTH security notions) for different choices of SE.kl, F.kl and F.ol. This assumes that the above adversaries are optimal, and computes the lower bound according to Section 2. Fig. 2 (in Section 1) shows the bit-security lower bounds with respect to privacy, depending on the choice of symmetric key length SE.kl and authentication tag length F.ol. Fig. 20 shows the choices of F.kl and F.ol that yield the best lower bounds for the bit-security of PRIV for each $\mathsf{SE.kl} \in \{128, 192, 256\}$. According to our results, the security of the iMessage-based signcryption scheme would slightly improve if the value of F.ol was chosen to be 48 instead of 40. The bit-security of SC with respect to AUTH is constant because it does not depend on the values of

$\mathsf{SE.kl}$, $\mathsf{F.kl}$, $\mathsf{F.ol}$. The assumption that RSA-OAEP with 1280-bit long keys has 80 bits of INDCCA-security limits the bit-security that can be achieved when $\mathsf{SE.kl} = 256$; otherwise, the PRIV bit-security for $\mathsf{SE.kl} = 256$ would allow a lower bound of 86 bits. But note that using $\mathsf{SE.kl} \in \{192, 256\}$ is likely not possible while maintaining the backward-compatibility of iMessage.

Acknowledgments

The authors were supported in part by NSF grant CNS-1717640 and a gift from Microsoft. Igors Stepanovs's work was done while at UCSD. We thank Adina Wollner, Wei Dai and Joseph Jaeger for discussions and insights.

References

1. M. Abdalla, M. Bellare, and G. Neven. Robust encryption. *TCC 2010*.
2. J. Alwen, S. Coretti, and Y. Dodis. The double ratchet: Security notions, proofs, and modularization for the signal protocol. *EUROCRYPT 2019*.
3. J. H. An, Y. Dodis, and T. Rabin. On the security of joint signature and encryption. *EUROCRYPT 2002*.
4. Apple. iOS security: iOS 12.3. Technical whitepaper, `https://www.apple.com/business/docs/site/iOS_Security_Guide.pdf`, May 2019.
5. E. Barker. Recommendation for key management part 1: General (revision 5). *NIST special publication*, 800(57):1–174, 2019.
6. M. Bellare, A. Boldyreva, K. Kurosawa, and J. Staddon. Multirecipient encryption schemes: How to save on bandwidth and computation without sacrificing security. *IEEE Trans. Information Theory*, 53(11):3927–3943, 2007.
7. M. Bellare and T. Kohno. A theoretical treatment of related-key attacks: RKA-PRPs, RKA-PRFs, and applications. *EUROCRYPT 2003*.
8. M. Bellare and C. Namprempre. Authenticated encryption: Relations among notions and analysis of the generic composition paradigm. *ASIACRYPT 2000*.
9. M. Bellare, R. Ng, and B. Tackmann. Nonces are noticed: AEAD revisited. *CRYPTO 2019*.
10. M. Bellare and P. Rogaway. Random oracles are practical: A paradigm for designing efficient protocols. *ACM CCS 1993*.
11. M. Bellare and P. Rogaway. The security of triple encryption and a framework for code-based game-playing proofs. *EUROCRYPT 2006*.
12. M. Bellare, A. C. Singh, J. Jaeger, M. Nyayapati, and I. Stepanovs. Ratcheted encryption and key exchange: The security of messaging. *CRYPTO 2017*.
13. M. Bellare and I. Stepanovs. Security under message-derived keys: Signcryption in imessage. Cryptology ePrint Archive, Report 2020/224, 2020.
14. J. Black, P. Rogaway, and T. Shrimpton. Encryption-scheme security in the presence of key-dependent messages. *SAC 2002*.
15. P. Bose, V. T. Hoang, and S. Tessaro. Revisiting AES-GCM-SIV: Multi-user security, faster key derivation, and better bounds. *EUROCRYPT 2018*.
16. J. Camenisch and A. Lysyanskaya. An efficient system for non-transferable anonymous credentials with optional anonymity revocation. *EUROCRYPT 2001*.
17. R. Canetti, Y. T. Kalai, M. Varia, and D. Wichs. On symmetric encryption and point obfuscation. *TCC 2010*.

18. R. Canetti, H. Krawczyk, and J. B. Nielsen. Relaxing chosen-ciphertext security. *CRYPTO 2003*.
19. K. Cohn-Gordon, C. Cremers, B. Dowling, L. Garratt, and D. Stebila. A formal security analysis of the Signal messaging protocol. *Proc. IEEE European Symposium on Security and Privacy (EuroS&P) 2017*.
20. Common Vulnerabilities and Exposures system. Cve-2016-1788. Available at `https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2016-1788`.
21. Damien Giry. Cryptographic key length recommendation. Available at `https://www.keylength.com`.
22. F. B. Durak and S. Vaudenay. Bidirectional asynchronous ratcheted key agreement with linear complexity. *IWSEC 19*.
23. P. Farshim, B. Libert, K. G. Paterson, and E. A. Quaglia. Robust encryption, revisited. *PKC 2013*.
24. P. Farshim, C. Orlandi, and R. Roşie. Security of symmetric primitives under incorrect usage of keys. *IACR Trans. Symm. Cryptol.*, 2017(1):449–473, 2017.
25. M. Fersch, E. Kiltz, and B. Poettering. On the provable security of (EC)DSA signatures. *ACM CCS 2016*.
26. C. Garman, M. Green, G. Kaptchuk, I. Miers, and M. Rushanan. Dancing on the lip of the volcano: Chosen ciphertext attacks on apple iMessage. *USENIX Security 2016*.
27. J. Groth. Rerandomizable and replayable adaptive chosen ciphertext attack secure cryptosystems. *TCC 2004*.
28. J. Jaeger and I. Stepanovs. Optimal channel security against fine-grained state compromise: The safety of messaging. *CRYPTO 2018*.
29. D. Jost, U. Maurer, and M. Mularczyk. Efficient ratcheting: Almost-optimal guarantees for secure messaging. *EUROCRYPT 2019*.
30. A. K. Lenstra. Key length. Contribution to the handbook of information security. 2004.
31. D. Micciancio and M. Walter. On the bit security of cryptographic primitives. *EUROCRYPT 2018*.
32. OpenIM wiki. iMessage. Available at `https://wiki.imfreedom.org/wiki/IMessage`.
33. B. Poettering and P. Rösler. Towards bidirectional ratcheted key exchange. *CRYPTO 2018*.
34. Quarkslab. iMessage privacy. Available at `https://blog.quarkslab.com/imessage-privacy.html`, Oct. 2013.
35. P. Rogaway. Authenticated-encryption with associated-data. *ACM CCS 2002*.
36. Y. Zheng. Digital signcryption or how to achieve cost(signature & encryption) $\ll$ cost(signature) + cost(encryption). *CRYPTO 1997*.