Interactive Encryption, Message Authentication, and Anonymous Key Exchange

Yevgeniy Dodis[1] and Dario Fiore[2⋆]

[1] Department of Computer Science, New York University, USA
dodis@cs.nyu.edu
[2] IMDEA Software Institute, Spain
dario.fiore@imdea.org

Abstract. Public-Key Encryption (PKE) and Message Authentication (PKMA, aka as digital signatures) are fundamental cryptographic primitives. Traditionally, both notions are defined as non-interactive (i.e., single-message). In this work, we initiate rigorous study of (possibly) *interactive* PKE and PKMA schemes. We obtain the following results demonstrating the power of interaction to resolve questions which are either open or impossible in the non-interactive setting.

Efficiency/Assumptions. One of the most well known open questions in the area of PKE is to build, in a "black-box way", so called chosen ciphertext attack (CCA-) secure PKE from chosen plaintext attack (CPA-) secure PKE. In contrast, we show a simple 2-round CCA-secure PKE from any (non-interactive) CPA-secure PKE (in fact, these primitives turn out to be equivalent). Similarly, although non-interactive PKMA schemes can be inefficiently built from any one-way function, no efficient signature schemes are known from many popular number-theoretic assumptions, such as factoring, CDH or DDH. In contrast, we show an efficient 2-round PKMA from most popular assumptions, including factoring, CDH and DDH.

Advanced Properties. It is well known that no non-interactive signature (resp. encryption) scheme can be *deniable* (resp. *forward-secure*), since the signature (resp. ciphertext) can later "serve as an evidence of the sender's consent" (resp. "be decrypted if the receiver's key is compromised"). We also formalize a related notion of *replay-secure* (necessarily) interactive PKMA (resp. PKE) schemes, where the verifier (resp. encryptor) is assured that the "current" message can only be authenticated (resp. decrypted) by the secret key owner *now*, as opposed to some time in the past (resp. future). We observe that our 2-round PKMA scheme is both replay-secure and (passively) deniable, and our 2-round PKE scheme is both replay- and forward-secure. We also define and construct stronger forms of necessarily interactive PKE/PKMA schemes, called *confirmed encryption* and *confidential authentication*.

Anonymous Key Exchange. We extend our definitional framework for interactive PKE and PKMA schemes to give definitions and constructions of (necessarily interactive) *anonymous key exchange* (1-KE), where an anonymous (unkeyed) party establishes a key with an authenticated (keyed) party. Unlike the prior work, defining 1-KE by "downgrading" the hairy and complex definition of *mutually authenticated* key exchange (2-KE), our definition is very "short" and easy to understand. We also show simple and general connections between anonymous KE and (interactive) confirmed PKE/confidential PKMA schemes. As a result, we obtain old and new schemes for anonymous KE in a clean and modular manner.

Keywords: Public Key Encryption, Digital Signatures, Chosen Ciphertext Security, Man-in-the-Middle Attacks, Anonymous Key Exchange.

⋆ Work partially done while postdoc at NYU.

Table of Contents

1 Introduction

Digital signatures and public-key encryption (PKE) schemes are two of the most fundamental and well studied cryptographic primitives. Traditionally, both notions are defined as *non-interactive* (i.e., single message). Aside from obvious convenience — both the sender and receiver need not keep any state — such non-interactive "one-and-done" philosophy is also critical in many applications. Coupled with the fact that we now have many (often efficient) candidates for both signature and encryption schemes, it might appear that there is little value in extending the syntax of signature/encryption schemes to allow for (possibly) interactive realizations.

In this work we challenge this point of view, and initiate rigorous study of (possibly) *interactive signature and public-key encryption schemes*. For the former, we will actually use the term *Public-Key Message Authentication* (PKMA) scheme, as the term "signature" often comes with expectations of "non-repudiation", which is orthogonal to the standard notion of "unforgeability" the moment interaction is allowed. First, although we agree that some applications might critically rely on the non-interactivity of PKE/PKMA schemes, we believe that many other applications, including *arguably the most basic one of sending/receiving private/authentic messages*, might not so be fixated on non-interactivity. For such applications, it appears natural to allow the sender and the receiver to interact, especially if they are involved in a conversation anyway.

Second, in this work we will show that, by allowing a single extra message (i.e., a 2-round protocol), we can "resolve" two arguably most important open problems in the area of non-interactive PKE/PKMA schemes:[3] (a) "black-box" 2-round chosen-ciphertext attack (CCA-) secure PKE from any (non-interactive) chosen-plaintext attack (CPA-) secure PKE; (b) *efficient* 2-round strongly unforgeable PKMA scheme from a variety of simple assumptions, such as factoring and DDH.

Third, we point out several useful advanced properties of PKE/PKMA schemes which are *impossible to achieve without interaction*. While some of these properties (such as *deniable* PKMA [21,23,22,17]) were already extensively studied in the past, most others (such as interactive *forward-secure* PKE, *confirmed* PKE, *confidential* PKMA and *replay-secure* PKE/PKMA) appear to be new.

Finally, in this work we introduce another reason for developing a definitional framework for interactive PKE/PKMA schemes. It comes from the desire to build a smoother transition between the clean non-interactive notions of signature/encryption schemes and the much more "hairy" setting of (public-key authenticated) *key exchange* (KE) [7,11]. The latter setting not only involves multiple parties (each running multiple sessions), but also inherently requires interactive solutions (e.g., to prevent "replay attacks" mentioned earlier). As a result, formal definitions for KE take pages to define, making it virtually impossible to include formal treatment of KE protocols in introductory cryptography courses. As we show in this work, by properly defining *interactive* variants of encryption and authentication, and then naturally extending them to the (necessarily interactive) setting of KE protocols, we can arrive at clean and simple definitions/constructions. This gives a much more "incremental" path from PKE/PKMA to KE, which can be easily digested by an undergraduate student.

Related Work. Although our work is the first to offer a detailed and comprehensive study of interactive PKE/PKMA schemes, it is certainly not the first to consider these notions. The most related prior work in this regard is the famous "DDN-paper" on non-malleable cryptography [20,21]. This seminal work had many extremely important and influential results. Among them, it also considered non-malleable, interactive encryption and authentication, and briefly sketched[4] elegant constructions for both primitives. We discuss more in detail the relation with our work in the next section, when we describe our improvements over the DDN paper.

[3] Of course, we obtain this by changing the model to allow for interaction, which is the reason for the quotation marks.

[4] Due to the massive scope of [21], the DDN paper did not give formal definitions and proofs for the encryption results (saying they are "outside the scope" of their paper; see page 32), and only sketched the definition/proof for the authentication case.

To the best of our knowledge, the only other work providing a related definition (only for encryption) is the one of Katz [34]. However, our definition is stronger, as we place more restriction on the attacker to declare that the attacker 'acts as a wire'. Moreover, the solutions given in [34] use so called timing assumptions, while our constructions are in the standard model.

In the area of key exchange, a vast majority of papers (whose survey is beyond the scope of this work; see [7,11,36]) considered the mutually authenticated case, where *both* the server and the client have keys. As we discuss later, in this work we only consider a simpler variant of general KE protocols, called *anonymous* (or, sometimes, server-only or unilateral) KE [48,29,26,36], where only *one* of the parties has a key, and the other party is "anonymous".[5] To emphasize this distinction, we will denote anonymous KE as *1-KE*, and mutually authenticated KE as *2-KE*, and notice that all prior definitions for 1-KE were obtained by slightly "downgrading" long and hairy definitions of 2-KE to the anonymous setting. As such, the resulting definitions were still pretty complex and hard to digest. In contrast, our definition is obtained by slightly "upgrading" our (short and simple) definitions for interactive PKE/PKMA, resulting (in our opinion) in a much more intuitive and easier to digest definition.[6] Given the importance of the anonymous setting on its own right, and the fact that it already introduces many of the subtleties arising in the 2-KE literature, we find our simplifications for this setting significant.

1.1 Our Results

We now describe our motivations and results in more detail.

Definitional Framework. Our first goal, thinking ahead to the KE setting, was to extend the short and elegant definitions of non-interactive encryption/signatures to the interactive setting, without making the definitions long and tedious. Unfortunately, in the interactive setting things *are* more complicated, as issues of concurrency and state, among others, must be dealt with. The way we managed to achieve our goal, was to split our definitions into two parts. The first (somewhat boring) part is *independent* of the particular primitive (e.g., PKE/PKMA), and simply introduces the bare minimum of notions/notation to deal with interaction. For example (see Section 2.1 for details), we define (a) what it means to have (concurrent) oracle access to an *interactive party* under attack; and (b) what it means to 'act as a wire' between two honest parties (for brevity, we call this trivial, but unavoidable, attack a 'ping-pong' attack). Once the notation is developed, however, our actual definitions of possibly interactive PKE/PKMA are *as short and simple as in the non-interactive setting* (see Definitions 5 and 6). E.g., in the PKMA setting (Definition 6) the attacker $\mathcal{A}$ has (concurrent) oracle access to the honest signer (as defined in (a)), and simultaneously tries to convince an honest verifier (i.e., "challenger"). $\mathcal{A}$ wins if the challenger accepts, and $\mathcal{A}$'s forgery was not a 'ping-pong' of one of its conversations with the signer (as defined in (b)).[7] Overall, the definition consists of the same couple of lines as in the non-interactive setting! And the same holds for the encryption case in Definition 5, which naturally generalizes the notion of CCA-security to the interactive setting.

We also show that our simple definitional framework (one generic/reusable 'long-and-boring' part followed by many application-specific 'short-and-intuitive' parts) easily lends itself to various extensions. Examples include various notions of privacy and/or authenticity which are *impossible* in the non-interactive setting (such as forward-secure PKE and replay-secure PKE/PKMA, discussed later in this section), and the already mentioned notion of anonymous KE, which we briefly discuss now. Indeed, our definition of anonymous KE (see Definition 8) is very simple. The attacker $\mathcal{A}$ has (concurrent) oracle access to the honest secret key owner (the "server"), and simultaneously tries to establish a session key

[5] We note that the anonymous setting is very important *on its own right*, as in the majority of real-life applications ordinary clients indeed do not have keys/certificates!

[6] We stress, we are not suggesting that we can similarly simplify the more complicated 'long-and-hairy' definitions of 2-KE, leaving this important question to future work.

[7] This generalizes the notion of *strong* unforgeability, as opposed to regular unforgeability, as was done in DDN [21].

with a honest anonymous client (the "challenger"). If the challenger rejects, then $\mathcal{A}$ 'lost'.[8] If it accepts and the session is *not* a ping-pong of one its conversations with the server, then $\mathcal{A}$ 'won', since it 'fooled' the challenger without trivially forwarding messages from the honest server. Otherwise, if $\mathcal{A}$ established a valid key with the challenger by a ping-pong attack, $\mathcal{A}$ 'wins' if it can distinguish a (well-defined!) 'real' session key from a completely random key.[9]

Better Efficiency/Assumptions via Interaction. Turning from definitions to constructions, we show how a *single* extra round of interaction (i.e., a 2-round protocol) can help "solve" (in a sense explained in Footnote 3) two of the arguably toughest open problems in the areas of non-interactive PKE and PKMA, respectively.

In the area of PKE, the question is to build a CCA-secure PKE from a CPA-secure PKE. In principle, such constructions are known using an appropriately-chosen notion of non-interactive zero-knowledge proofs [21,43,44,46,37]. However, all these constructions are generally inefficient and considered somewhat unsatisfactory, since they use the code of the given CPA-secure encryption scheme. In particular, the question of finding so called *black-box* constructions of CCA-encryption from CPA-encryption remains open. In fact, although several partial progress along both positive (e.g., [14,13,39]) and negative (e.g., [28]) fronts was made, the general question remains elusive. In contrast, we show a relatively simple, black-box, 2-round CCA-secure encryption from CPA-secure encryption (in fact, the two primitives are *equivalent*). We notice that the DDN paper [21] itself already made a similar conclusion, by presenting a 3-round black-box protocol from any CPA-secure PKE.[10] Thus, aside from presenting a formal model for interactive CCA-secure encryption, our result can be viewed as improving the round efficiency of the DDN paper from 3 to 2.

In the area of PKMA, the "theory" question was settled pretty quickly, by showing that the strongest security notion for signature schemes — strong existential unforgeability against chosen message attack — can be realized assuming the mere existence of one-way functions [30,42,4,5,45]. Unfortunately, the generic constructions were primarily of theoretical interest, and did not result in practical enough signature schemes. In fact, practical signature schemes (outside of the random oracle model [8,25,47,31]) are only constructed from a handful of "not-too-standard" number-theoretic assumptions, such as 'strong RSA' [16,27] and 'Bilinear-Diffie-Hellman' [49], and even these 'practical' constructions were generally somewhat slow, requiring generation of primes, long keys or bilinear maps. In particular, one of the main open questions is to build an *efficient* digital signature scheme from a standard assumption, such as factoring, CDH or DDH.

In contrast, we show very efficient, 2-round, strongly unforgeable PKMA schemes from virtually all standard assumptions, including factoring, CDH and DDH. In fact, although we are not aware of any paper explicitly claiming this result, *it follows in a relatively simple manner by combining various prior works.*[11] Let us explain. With a different motivation in mind, the DDN paper [21] showed a simple 3-round[12] PKMA scheme from any CCA-secure encryption. At the time, CCA-secure PKE was considered a very 'advanced' primitive, so the construction was not considered 'efficient'. Over the years, though, many truly efficient CCA-secure schemes were constructed from virtually all popular assumptions, including factoring [32], CDH [50] and DDH [15] (despite the fact that no such efficient signature schemes are known from these assumptions!). Thus, these results, if combined, immediately yield an efficient 3-round PKMA from all these assumptions. Moreover, it is well known (e.g., see [35,1]) that one can reduce the number of rounds from 3 to 2 by also using a message authentication code

[8] Notice, since anybody can establish a key with the server, to succeed $\mathcal{A}$ must establish the key with a honest client.

[9] Notice, for elegance sake our basic definition does not demand advanced properties, such as forward security or deniability, but (as we show) can be easily extended to do so. We also stress that our goal was not to get the most 'advanced' KE definition, but rather to get a strong and useful definition which is short, intuitive, and easy to digest!

[10] Although they do not give a formal definition/proof, their construction is easily seen to be secure in our model (see Appendix B).

[11] Except only establishing regular unforgeability, but the actual constructions are easily seen to be strongly unforgeable.

[12] The construction becomes 2-round if the verifier knows the authenticated message in advance.

(MAC), in addition to CCA-secure encryption. Of course, in theory, a MAC is implied by a CCA-secure PKE, albeit in an inefficient manner. Moreover, until recently, even direct efficient MAC constructions from concrete assumptions, such as DDH [40] and factoring [41], required long keys (quadratic in security parameter). Fortunately, Dodis et al. [19] recently observed an elementary efficient (probabilistic) MAC construction from any CCA-secure scheme. This gives an efficient 2-round PKMA scheme from any CCA-secure encryption. We also manage to further optimize the resulting construction, and obtain a really simple (new!) 2-round protocol, depicted in Figure 3. In turn, this gives efficient 2-round PKMA from a variety of standard assumptions, including factoring, CDH and DDH.[13]

Duality between *Interactive* PKE and PKMA. Interestingly, our 2-round CCA-secure PKE uses a *signing* key as its long-term "decryption secret" (and generates several ephemeral keys for the CPA-secure scheme), while our 2-round strongly unforgeable PKMA scheme uses a *decryption* key for a CCA-secure encryption as its long term "authentication secret". We show that this duality is not a coincidence. In fact, our 2-round results follow as corollaries of two more general schemes, depicted in Figures 1 and 2: an interactive CCA-secure scheme from any (interactive or not) strongly unforgeable PKMA scheme (plus any CPA-secure PKE[14]), and an interactive strongly unforgeable PKMA scheme from any (interactive or not) CCA-secure PKE. The 'duality' of our results (authentication using encryption and vice-versa) shows that, perhaps, the practical/theoretical distinction between *interactive* encryption and authentication is not as great as one could have guessed by looking at what is known in the *non-interactive* setting.

Overcoming Impossibility Results. We already mentioned that interactive PKE/PKMA might achieve advanced security properties which are impossible in the non-interactive setting. One such (well studied [21,23,22,17]) notion is that of deniable authentication, which was actually the original motivation of the DDN paper. Since this is not the main topic of this work, we only define the weakest notion of *passive deniability* (and its extension called 'passive forward deniability' [17]), and observe that our optimized 2-round variant from non-interactive CCA-secure PKE is passively forward deniable.[15]

Another example is the notion of *forward security*, which (intuitively) states that 'old' message transmissions should remain private even if the party's long term secret key is later compromised. Prior to our work, forward security has been extensively studied in the KE literature (in fact, in many cases being a mandatory part of a 'secure' KE). On the other hand, forward security is (obviously) impossible for non-interactive PKE without "changing the model"; e.g. by introducing global time periods, and periodically refreshing the secret key [10]. In contrast, no such impossibility exists for interactive PKE, and, indeed, our interactive PKE schemes are forward-secure, since all of them use ephemeral keys to actually encrypt the message.

Yet another limitation of non-interactive PKE/PKMA schemes is that they necessarily suffer from what we informally (for now) term "replay" attacks. In the case of encryption, for example, an attacker can always record an 'old' ciphertext, and then manage to decrypt it much later. Similarly, a verifier can always pass an 'old' signature to another verifier in the future. Motivated by this impossibility, we formalize (to the best of our knowledge, for the first time) the notion of *replay-secure* (necessarily) interactive PKMA/PKE schemes. For the former, a honest verifier is assured that the "current" message is actually being authenticated by the secret key owner "now", as opposed to some time in the past. For the latter, a honest encryptor is similarly assured that the "current" message can only be decrypted by the secret key owner "now", as opposed to some time in the future. We then show that *any* interactive

[13] Of course, in practice one should not use CCA-secure encryption to build a MAC (instead, one should use practical MACs such as CBC-MAC or HMAC), but here we use it to establish *efficient feasibility results* from concrete number-theoretic assumptions.

[14] For the sake of elegance and modularity, we use a slightly stronger notion of so called "1-bounded CCA-secure PKE", but the latter can be built from any CPA-secure PKE [14].

[15] The construction can be made 'actively deniable', with more rounds, using the techniques developed by [23].

PKE/PKMA scheme which has at least 2 rounds is already replay-secure.[16] For example, we automatically get replay-secure PKE/PKMA schemes, by using the 2-round solutions in this paper.[17] We also notice that a very special case of our replay-secure PKMA, when the message space has cardinality 1, essentially corresponds to the strongest security notion for identification schemes, called impersonation security under concurrent attacks [6]. Here the attacker has concurrent oracle access to the prover (i.e., 'signer of a fixed message'), then loses this oracle access, and, finally, has to convince an honest verifier. In fact, our 2-round PKMA protocols, when specialized to this trivial case, essentially "collapse" to well-known challenge-response identification protocols from CCA-encryption and signature schemes, respectively. Of course, by having an extra "non-trivial" message, we think that replay-secure PKMA schemes should have more applications than concurrently secure identification schemes.

Finally, we define another two strengthenings of PKE/PKMA which inherently require interaction: *confirmed encryption* and *confidential authentication*. In brief, confirmed encryption is an extension of our interactive encryption in which the sender gets a confirmation that the receiver obtained the correct (encrypted) message, and thus accepts/rejects accordingly. Confidential authentication, instead, adds a privacy property to PKMA protocols in such a way that no information about the message is leaked to adversaries controlling the communication channel (and, yet, the receiver gets the message!). Clearly, both notions require interaction, and we show both can be realized quite naturally with two rounds of interaction. Moreover, as we comment below, these two notions provide two modular and "dual" ways to build secure anonymous KE protocols.

Key Exchange Results. As we mentioned, our anonymous KE (1-KE) definition is a natural extension of our interactive PKE/PKMA definitions. As a result, we show two simple and *very natural* constructions of 1-KE protocols: from any possibly interactive PKE scheme, depicted in Figure 5, and from any possibly interactive PKMA scheme and CPA-secure key encapsulation mechanism (KEM), depicted in Figure 6. By plugging various non-interactive or 2-round PKE/PKMA schemes (and KEMs, such as the classical Diffie-Hellman KE), we get a variety of simple and natural 1-KE protocols. For example, we re-derive the A-DHKE-1 protocol from [48] and the unilateral version of the SKEME protocol [35].

In Section 4.5, we also further abstract our 1-KE constructions in Figures 5 and 6 by using the notions of confirmed PKE and confidential PKMA mentioned above. Namely, we show that "confirmed encryption of random K" and "confidential authentication of random K" both yield secure 1-KE protocols. Overall, we believe that our work gives a very intuitive path from traditional non-interactive PKE/PKMA schemes, to interactive PKE/PKMA, to (interactive) confirmed PKE/confidential PKMA, to anonymous KE. Given that anonymous KE, aside from independent interest, already introduces many of the subtleties of mutually authenticated KE (2-KE), we hope our work can also simplify the introduction of 2-KE to students, but leave it as an interesting open problem if the "last jump" from 1-KE to 2-KE can be made without completely redoing the definitional framework.

Simplicity in Retrospect. We notice that, after developing our definitional framework, all our results are derived as simple corollaries/extensions of a handful of extremely *general and intuitive* protocols: interactive PKE from any PKMA (and CPA-secure encryption; Figure 1), interactive PKMA from any PKE (Figure 2), anonymous KE from any PKE (Figure 5), anonymous KE from any PKMA (and CPA-secure KEM; Figure 6), and round-extension for PKE/PKMA schemes (Figure 4). This simplicity should be viewed as a big *advantage* of our definitional framework: (1) we *want* definitions lending themselves to natural and intuitive realizations, both for the sake of aesthetics as well as practicality (as complicated solutions are unlikely to be used); and (2) we *want* to be able to derive non-trivial and possibly surprising conclusions (like 2-round CCA-secure PKE from CPA-secure PKE, or efficient

[16] For encryption, we also define a stronger notion of replay-security, which requires at least 3 rounds, and is realized by any 3-round CCA secure scheme.

[17] This includes already mentioned 2-round PKMA from CCA-encryption and 2-round PKE from signatures, as well as simple 2-round PKE/PKMA obtained by "extending" 1-round PKE/PKMA schemes.

2-round PKMA from factoring) in a modular manner, by using a sequence of seemingly elementary transformations. Most importantly, we believe all our results (including KE!) can be easily taught in an undergraduate cryptography course.

1.2 Organization of the paper

The paper is organized as follows. In Section 2 we introduce our definitional framework for message transmission protocols with the security notions of (possibly interactive) iCCA-secure PKE and iCMA-secure PKMA. Moreover, we use our framework to define anonymous key exchange. Next, in Section 3, we focus on realizations of these protocols, notably, iCCA-secure PKE from iCMA-secure PKMA and CPA-secure encryption, iCMA-secure PKMA from iCCA-secure PKE, and anonymous KE based on iCCA and iCMA security. Finally, Section 4 discusses advanced security properties for (interactive) PKE and PKMA, and for anonymous KE. We postpone to the appendix notation and standard definitions (Appendix A), the 3-round interactive encryption of Dolev, Dwork and Naor [21], the construction of a 1-bounded-$\mathsf{IND-CCA}$-secure PKE scheme from an $\mathsf{IND-CPA}$-secure one (Appendix C), the extension of (interactive) PKE to support "labels" and its application to PKMA (Appendices D–E), and a few simple proofs (Appendix F).

2 Defining Message Transmission and Anonymous Key-Exchange Protocols

In our paper we use relatively standard notation recalled in Appendix A. This section is devoted to introducing the syntax and the security notions for the primitives considered by our work. First, in Section 2.1, we focus on *message transmission protocols*: we define their syntax as well as suitable notions of confidentiality (called iCCA security) and authenticity (called iCMA security). Second, in Section 2.4, we move to *anonymous key-exchange*. We provide its syntax and a security notion, which builds upon the same definitional framework of message transmission protocols.

2.1 Message Transmission Protocols

We give a generic definition of message transmission protocols involving two parties: a sender S and a receiver R, such that the goal of S is to send a message m to R while preserving certain security properties on m. In particular, in the next two sections we consider arguably the most basic security properties: the confidentiality/authenticity of the messages sent by S to R. Formally, a message transmission protocol Π consists of algorithms $(\mathsf{Setup}, \mathsf{S}, \mathsf{R})$ defined as follows:

$\mathsf{Setup}(1^\lambda)$: on input the security parameter λ, the setup algorithm generates a pair of keys $(\mathsf{sendk}, \mathsf{recvk})$. In particular, these keys contain an implicit description of the message space $\mathcal{M}$.

$\mathsf{S}(\mathsf{sendk}, m)$: is a possibly interactive algorithm that is run with the sender key sendk and a message $m \in \mathcal{M}$ as private inputs.

$\mathsf{R}(\mathsf{recvk})$: is a possibly interactive algorithm that takes as private input the receiver key recvk, and whose output is a message $m \in \mathcal{M}$ or an error symbol $\perp$.

When S and R are jointly run, they exchange messages in a specific order, e.g., S starts by sending M_1, R sends M_2, S sends M_3, and so on and so forth until they both terminate. We say that Π is an n-round protocol if the number of messages exchanged between S and R during a run of the protocol is n. If Π is 1-round, then we say that Π is *non-interactive*. Since the sender gets no output, we assume without loss of generality that the *sender always speaks last*. Thus, in an n-round protocol, R (resp. S) speaks first if n is even (resp. odd). For compact notation, we denote with $\langle \mathsf{S}(\mathsf{sendk}, m), \mathsf{R}(\mathsf{recvk}) \rangle = m'$ the process of running S and R on inputs (sendk, m) and recvk respectively, and assigning R's output to m'. In our notation, we will use $m \in \mathcal{M}$ for messages (aka plaintexts), and capital M for protocol messages.

Definition 1 (Correctness). *A message transmission protocol* $\Pi = (\mathsf{Setup}, \mathsf{S}, \mathsf{R})$ *is* correct *if for all honestly generated keys* $(\mathsf{sendk}, \mathsf{recvk}) \xleftarrow{\$} \mathsf{Setup}(1^\lambda)$*, and all messages* $m \in \mathcal{M}$*, we have that* $\langle \mathsf{S}(\mathsf{sendk}, m), \mathsf{R}(\mathsf{recvk})\rangle = m$ *holds with all but negligible probability.*

Defining Security: Man-in-the-Middle Adversaries. In our work, we assume that the sender and the receiver speak in the presence of powerful adversaries that have full control of the communication channel, i.e., the adversary can eavesdrop the content of the communication, and it can stop/delay/alter the messages passing over the channel. Roughly speaking, the goal of an adversary is to violate a given security property (say confidentiality or authenticity) in a run of the protocol that we call the *challenge session*. Formally, this session is a protocol execution $\langle \mathsf{S}(\mathsf{sendk}, m), \mathcal{A}^{\mathsf{R}(\mathsf{recvk})}\rangle$ or $\langle \mathcal{A}^{\mathsf{S}(\mathsf{sendk},\cdot)}, \mathsf{R}(\mathsf{recvk})\rangle$ where the adversary runs with a honest party (S or R). By writing $\mathcal{A}^{\mathsf{P}}$, we mean that the adversary has oracle access to *multiple* honest copies of party P (where $\mathsf{P} = \mathsf{R}$ or $\mathsf{P} = \mathsf{S}$), i.e., $\mathcal{A}$ can start as many copies of P as it wishes, and it can run the message transmission protocol with each of these copies. This essentially formalizes the fact that the adversary can "sit in the middle of two honest parties" relaying messages between them in an active way. Sometimes, we also write $\mathcal{A}^{\mathsf{P}_k}$ to denote that the adversary can start at most k copies of party P. In our model we assume that whenever $\mathcal{A}$ sends a message to the oracle P, then $\mathcal{A}$ always obtains P's output. In particular, in the case of the receiver oracle, when $\mathcal{A}$ sends the last protocol message to R, $\mathcal{A}$ obtains the (private) output of the receiver, i.e., a message m or $\perp$.

Since all these protocol sessions can be run in a concurrent way, the adversary might entirely replay the challenge session by using its oracle. This is something that we would like to prevent in our definitions. To formalize this idea, we take an approach similar to the one introduced by Bellare and Rogaway [7] in the context of key exchange, which is based on the idea of "matching conversations". First of all, we introduce a notion of time during the execution of the security experiment. We stress that this is done for ease of analysis of the security model: there is no need to keep track of global timing in the real protocols. Let t be a global counter which is progressively incremented every time a party (including the adversary) sends a message, and assume that every message sent by a party (S, R or $\mathcal{A}$) gets timestamped with the current time t. Note that this includes all messages of the sessions established by the adversary using its oracle. Using this notion of time, we define the transcript of a protocol session as follows:

Definition 2 (Protocol Transcript). *The* transcript of a protocol session *between two parties is the timestamped sequence of messages exchanged by the parties during a run of the message transmission protocol* Π*. If* Π *is* n*-round, then a transcript* T *is of the form* $T = \langle (M_1, t_1), \ldots, (M_n, t_n)\rangle$*, where* $M_1, \ldots, M_n$ *are the exchanged messages, and* $t_1, \ldots, t_n$ *are the respective timestamps.*

In a protocol run $\langle \mathsf{S}(\mathsf{sendk}, m), \mathcal{A}^{\mathsf{R}(\mathsf{recvk})}\rangle$ (resp. $\langle \mathcal{A}^{\mathsf{S}(\mathsf{sendk},\cdot)}, \mathsf{R}(\mathsf{recvk})\rangle$) we have a transcript T^* of the challenge session between S and $\mathcal{A}$ (resp. $\mathcal{A}$ and R), and Q transcripts $T_1, \ldots, T_Q$, one for each of the Q sessions established by $\mathcal{A}$ with R (resp. S) via the oracle.

While we postpone to the next two sections the definition of specific security properties of message transmission (e.g., confidentiality and authenticity), our goal here is to formalize in a generic fashion which adversaries are effective for "uninteresting"/unavoidable reasons. Namely, when the challenge session is obtained by entirely replaying one of the oracle sessions: what we call a "ping-pong" attack, that we formalize via the following notion of matching transcripts.

Definition 3 (Matching Transcripts). *Let* $T = \langle (t_1, M_1), \ldots, (t_n, M_n)\rangle$ *and* $T^* = \langle (t_1^*, M_1^*), \ldots, (t_n^*, M_n^*)\rangle$ *be two protocol transcripts. We say that* T matches T^* *(*$T \equiv T^*$*, for short) if* $\forall i = 1, \ldots, n$*,* $M_i = M_i^*$ *and the two timestamp sequences are "alternating", i.e.,* $t_1 < t_1^* < t_2^* < t_2 < t_3 < \cdots < t_{n-1} < t_n < t_n^*$ *if* R *speaks first, or* $t_1^* < t_1 < t_2 < t_2^* < t_3^* < \cdots < t_{n-1} < t_n < t_n^*$ *if* S *speaks first.*

We remark that the notion of match is not commutative.

Given all the definitions above, we can finally define the notion of ping-pong adversary:

Definition 4 (Ping-pong Adversary). *Consider a run of the protocol $\mathsf{\Pi}$ involving $\mathcal{A}$ and a honest party (it can be either $\langle \mathsf{S}(\mathsf{sendk}, m), \mathcal{A}^{\mathsf{R}(\mathsf{recvk})}\rangle$ or $\langle \mathcal{A}^{\mathsf{S}(\mathsf{sendk},\cdot)}, \mathsf{R}(\mathsf{recvk})\rangle$), and let T^* be the transcript of the challenge session, and $T_1, \ldots, T_Q$ be the transcripts of all the oracle sessions established by $\mathcal{A}$. Then we say that $\mathcal{A}$ is a* ping-pong adversary *if there is a transcript $T \in \{T_1, \ldots, T_Q\}$ such that T matches T^*, i.e., $T \equiv T^*$.*

2.2 Interactive Chosen-Ciphertext-Secure Encryption

In this section we propose a suitable notion of confidentiality for message transmission protocols that we call *interactive chosen ciphertext security* (iCCA). Our notion is designed as a very natural generalization of the classical notion of IND−CCA security to the interactive setting. In fact, IND−CCA security is a special case of iCCA security for 1-round (i.e., non-interactive) protocols (we leave this check to the reader). Roughly speaking, in the IND−CCA definition (recalled in Appendix A) the adversary has to distinguish whether a given "challenge" ciphertext encrypts a message m_0 or m_1 while having access to a decryption oracle. To make the definition non-trivial, the adversary is denied to query the decryption oracle on the challenge ciphertext. Our notion of iCCA security is obtained similarly: the adversary $\mathcal{A}$ interacts with a sender which sends either m_0 or m_1 and $\mathcal{A}$ has to tell the two cases apart while having access to the receiver (instead of the decryption oracle); the restriction on the challenge ciphertext is replaced by requiring that $\mathcal{A}$ cannot be ping-pong. The formal definition follows.

Let $\mathsf{\Pi} = (\mathsf{Setup}, \mathsf{S}, \mathsf{R})$ be a message transmission protocol and $\mathcal{A}$ be an adversary. To define iCCA security, consider the following experiment:

```
Experiment $\mathbf{Exp}^{\mathsf{iCCA}}_{\mathsf{\Pi},\mathcal{A}}(\lambda)$
    $b \xleftarrow{\$} \{0,1\}$
    $(\mathsf{sendk}, \mathsf{recvk}) \xleftarrow{\$} \mathsf{Setup}(1^\lambda)$
    $(m_0, m_1) \leftarrow \mathcal{A}^{\mathsf{R}(\mathsf{recvk})}(\mathsf{sendk})$
    $b' \leftarrow \langle \mathsf{S}(\mathsf{sendk}, m_b), \mathcal{A}^{\mathsf{R}(\mathsf{recvk})}(\mathsf{sendk})\rangle$
    If $b' = b$ and $\mathcal{A}$ is not "ping-pong", then output 1
    Else output 0.
```

Definition 5 (iCCA security). *For any $\lambda \in \mathbb{N}$, we define the advantage of an adversary $\mathcal{A}$ in breaking* iCCA *security of a message transmission protocol $\mathsf{\Pi}$ as $\mathbf{Adv}^{\mathsf{iCCA}}_{\mathsf{\Pi},\mathcal{A}}(\lambda) = \left|\Pr[\mathbf{Exp}^{\mathsf{iCCA}}_{\mathsf{\Pi},\mathcal{A}}(\lambda) = 1] - \frac{1}{2}\right|$, and we say that $\mathsf{\Pi}$ is* iCCA*-secure if for any PPT $\mathcal{A}$, $\mathbf{Adv}^{\mathsf{iCCA}}_{\mathsf{\Pi},\mathcal{A}}(\lambda)$ is negligible. We call a message transmission protocol satisfying this notion an* interactive encryption *scheme.*

As we mentioned in the introduction, it is worth noting that our definition is similar to the one proposed by Katz in [34]. The main difference is in the restrictions applied to the adversary in the security game. In [34] this is realized by means of a notion of "equality of transcripts" which considers only equality of messages (in the same order) and leaves any time constraint to the specific protocols realizations. Our security definition, instead, directly takes into account time constraints in the security experiment via the notion of matching transcripts. To see the difference between the two definitions with an example, consider an adversary who creates an oracle session having the same transcript as the one of the challenge session, but where the timestamps of the messages are not correctly alternating. Such an adversary would not be legal according to the definition of [34], but is legal (i.e., not ping-pong) according to ours.

Later, in Section 4.1, we strengthen our requirements by considering an orthogonal security property for interactive encryption that we call "Replay Security". This will allow to model MiM attacks more easily and from a different perspective. It will also turn out to be realizable under our basic iCCA notion. However, since a replay attack is *always* possible in any non-interactive solution, replay security will only be realizable by an *interactive* protocol.

q-bounded-iCCA Security. In this work we also consider a weaker notion of iCCA security, called *q-bounded*-iCCA, in which the adversary is restricted to complete at most q sessions with the oracle R, i.e., in $\mathbf{Exp}^{\mathsf{iCCA}}_{\Pi,\mathcal{A}}(\lambda)$ we run $\mathcal{A}^{\mathsf{R}_q}$. The same notion has been considered in the non-interactive setting [14], with the name of q-bounded $\mathsf{IND-CCA}$ security.

Labeled (Interactive) Encryption. We also consider an extension of interactive encryption in which both the sender and the receiver algorithms take a public string—a label—as an additional input (similarly to the non-interactive setting [9]). Very intuitively, the idea is that the receiver working with label L decrypts correctly only if the sender works with the same label L. In Appendix D we provide a full formalization of this notion and we show that it can be generically constructed from 'plain' iCCA-secure encryption.

2.3 Interactive Chosen-Message Secure Public Key Message Authentication

In this section we propose a suitable notion of authenticity for message transmission protocols that we call *interactive unforgeability under chosen message attacks* (iCMA). Our notion is designed as a very natural generalization to the interactive setting of the standard notion of strong unforgeability ($\mathsf{suf\text{-}cma}$) for digital signatures. In fact, $\mathsf{suf\text{-}cma}$ security is a special case of iCMA security for 1-round protocols. Roughly speaking, in the $\mathsf{suf\text{-}cma}$ definition (see Appendix A) the adversary has to produce a valid signature while having access to the signer. In order for the definition to be non-trivial, however, such signature has to be "new", i.e., not obtained from the signing oracle. Our notion of iCMA security naturally extends $\mathsf{suf\text{-}cma}$ as follows: the adversary $\mathcal{A}$ has to convince a receiver while having oracle access to the sender (instead of the signing oracle); the requirement that the signature must be new is replaced by requiring that $\mathcal{A}$ is not ping-pong. The formal definition follows.

Let $\Pi = (\mathsf{Setup}, \mathsf{S}, \mathsf{R})$ be a message transmission protocol and $\mathcal{A}$ be an adversary. To define iCMA security, consider the following experiment:

```
Experiment $\mathbf{Exp}^{\mathsf{iCMA}}_{\Pi,\mathcal{A}}(\lambda)$
    $(\mathsf{sendk}, \mathsf{recvk}) \xleftarrow{\$} \mathsf{Setup}(1^\lambda)$
    $m^* \leftarrow \langle \mathcal{A}^{\mathsf{S}(\mathsf{sendk},\cdot)}(\mathsf{recvk}), \mathsf{R}(\mathsf{recvk})\rangle$
    If $m^* \neq \perp$ and $\mathcal{A}$ is not "ping-pong", then output 1
    Else output 0.
```

Definition 6 (iCMA security). *For any $\lambda \in \mathbb{N}$, the advantage of $\mathcal{A}$ in breaking the* iCMA *security of a message transmission protocol Π is $\mathbf{Adv}^{\mathsf{iCMA}}_{\Pi,\mathcal{A}}(\lambda) = \Pr[\mathbf{Exp}^{\mathsf{iCMA}}_{\Pi,\mathcal{A}}(\lambda) = 1]$, and we say that Π is* iCMA-*secure if for any PPT $\mathcal{A}$, $\mathbf{Adv}^{\mathsf{iCMA}}_{\Pi,\mathcal{A}}(\lambda)$ is negligible. We call a message transmission protocol satisfying this notion a* public key message authentication *(PKMA) protocol.*

Later, we further motivate the importance of interactive PKMA protocols by considering additional security notions that can be realized *only* in the interactive setting. This is the case for our notion of Replay Security (see Section 4.1), and for the *deniability* property (see Section 4.3). The former notion intuitively captures the idea of obtaining security by denying the adversary to access the sender oracle during the challenge session. Deniability, instead, allows the authenticator to deny his active participation in a protocol, i.e., it denies the verifier to transfer the information authenticated by S.

2.4 Anonymous Key-Exchange

While the notions of iCCA/iCMA secure message transmission protocols may be interesting on their own right, here we expand an additional motivation of these notions: a smoother and clean transition from encryption/authentication towards key exchange. In particular, in this work we focus on *anonymous key-exchange* (1-KE, for short).[18] 1-KE is a weaker form of bilateral key-exchange in which only one of

[18] In the literature this primitive has been also called one-way-authenticated or unilateral KE. We use anonymous KE following Shoup [48].

the two protocol parties is authenticated. This way, the protocol does not provide any cryptographic evidence about the identity of the unauthenticated party (from which the name "anonymous KE"). 1-KE has been previously considered by Shoup [48], Goldberg *et al.* [29] (in the context of Tor), Fiore *et al.* [26] (in the identity-based setting), and Jager *et al.* [33] and Krawczyk *et al.* [36] (in the context of TLS). All these works arrived at anonymous key-exchange by following essentially the same approach: they started from (some standard definitions of) mutually-authenticated KE, and then they relaxed this notion by introducing one "dummy" user which can run the protocol without any secret (so, the anonymous party will run the protocol on behalf of such user), and by slightly changing the party-corruption condition.

In our work, we take a different approach and show how to obtain 1-KE starting from the simple notions of $\mathsf{iCCA}/\mathsf{iCMA}$-secure message transmission. Following our style of definitions, we define 1-KE as a protocol between two parties—in this case, an un-keyed (aka anonymous) user U and a keyed (aka authenticated) user T—so that, at the end of a successful protocol run, both parties (privately) output a common session key. Formally, a 1-KE protocol Π consists of algorithms $(\mathsf{KESetup}, \mathsf{U}, \mathsf{T})$ working as follows:

$\mathsf{KESetup}(1^\lambda)$: on input the security parameter λ, the setup algorithm generates a pair of keys $(\mathsf{uk}, \mathsf{tk})$. Implicitly, it also defines a session key space $\mathcal{K}$.

$\mathsf{U}(\mathsf{uk})$: is a possibly interactive algorithm that takes as input the public key uk of the authenticated user, and outputs a session key or a symbol $\perp$.

$\mathsf{T}(\mathsf{tk})$: is a possibly interactive algorithm that takes as input the private key tk, and outputs a session key K or an error symbol $\perp$.

In our security definitions we explicitly include the property that U terminates correctly (i.e., no $\perp$ output) only if U gets confirmation that T can terminate correctly. For this reason, we assume without loss of generality that T *always speaks last.* For compact notation, we denote with $\langle \mathsf{U}(\mathsf{uk}), \mathsf{T}(\mathsf{tk}) \rangle = (\mathsf{K_U}, \mathsf{K_T})$ a run of the protocol in which U and T output $\mathsf{K_U}$ and $\mathsf{K_T}$ respectively.

Definition 7 (Correctness). *An anonymous key-exchange protocol* $\Pi = (\mathsf{KESetup}, \mathsf{U}, \mathsf{T})$ *is* correct *if for all honestly generated key pairs* $(\mathsf{uk}, \mathsf{tk}) \xleftarrow{\$} \mathsf{KESetup}(1^\lambda)$ *and all session keys* $\langle \mathsf{U}(\mathsf{uk}), \mathsf{T}(\mathsf{tk}) \rangle = (\mathsf{K_U}, \mathsf{K_T})$*, we have that* $\mathsf{K_U} = \mathsf{K_T} \neq \perp$ *holds with all but negligible probability.*

For 1-KE protocols we aim at formalizing two main security properties: *authenticity* and *confidentiality.* Intuitively, authenticity says that the only way for an adversary to make the un-keyed party terminate correctly (no $\perp$ output) is to be ping-pong. Basically, this authenticity notion captures the property of *explicit key confirmation* that has been considered in the area of authenticated key exchange. Confidentiality aims to capture that, once the un-keyed party U accepted, then the adversary cannot learn any information about the session key (unless it is ping-pong up to learning the key). We formalize these two properties in a single experiment in which $\mathcal{A}$ runs a challenge session with the un-keyed party U while having access to the keyed party T. Since in 1-KE T speaks last, we allow the adversary to make one additional query to T after T generated the last message: in this case T reveals its private output $\mathsf{K_T}$. If $\mathcal{A}$ makes such an additional query in a ping-pong session, then we say that $\mathcal{A}$ is "full-ping-pong". Although the resulting experiment looks a bit more complex compared to the ones of iCCA and iCMA security, we stress that it can be seen as a natural combination of these two security notions. At a high level, the experiment consists in first running $(K_0, \cdot) \leftarrow \langle \mathsf{U}(\mathsf{uk}), \mathcal{A}^{\mathsf{T}(\mathsf{tk})}(\mathsf{uk}) \rangle$ and then analyzing U's output K_0 ($\cdot$ means that we do not care about $\mathcal{A}$'s output at this stage). If $K_0 \neq \perp$ and $\mathcal{A}$ is *not* ping-pong, then $\mathcal{A}$ wins (it broke authenticity). Otherwise, we give to $\mathcal{A}$ a real-or-random key K_b and $\mathcal{A}$ wins if it can tell these two cases apart *without*, of course, pushing the ping-pong attack up to getting K_0 revealed from the oracle T. Notice that when $K_0 = \perp$ (i.e., the honest sender did not accept in the challenge session), we also set $K_1 = \perp$. This is meant to capture that if U does not accept, then there is no common session key established by the two parties (essentially, no secure channel will be established). In this case the adversary will have no better chances of winning the game than guessing b.

```
Experiment $\mathbf{Exp}^{\mathsf{1-KE-Sec}}_{\Pi,\mathcal{A}}(\lambda)$
    $(\mathsf{uk},\mathsf{tk}) \xleftarrow{\$} \mathsf{KESetup}(1^\lambda)$;
    $b \xleftarrow{\$} \{0,1\}$
    $(K_0,\cdot) \leftarrow \langle \mathsf{U}(\mathsf{uk}), \mathcal{A}^{\mathsf{T}(\mathsf{tk})}(\mathsf{uk})\rangle$
    If $K_0 = \perp$, then $K_1 = \perp$
    Else if $K_0 \neq \perp$ and $\mathcal{A}$ is not "ping-pong", then output 1
    Else $K_1 \xleftarrow{\$} \mathcal{K}$
    $b' \leftarrow \mathcal{A}^{\mathsf{T}(\mathsf{tk})}(K_b)$
    If $b' = b$ and $\mathcal{A}$ is not "full-ping-pong", then output 1
    Else output 0.
```

Definition 8 (Security of 1-KE). *We define the advantage of an adversary $\mathcal{A}$ in breaking the security of Π as $\mathbf{Adv}^{\mathsf{1-KE-Sec}}_{\Pi,\mathcal{A}}(\lambda) = \left|\Pr[\mathbf{Exp}^{\mathsf{1-KE-Sec}}_{\Pi,\mathcal{A}}(\lambda) = 1] - \frac{1}{2}\right|$, and we say that a 1-KE protocol Π is secure if for any PPT $\mathcal{A}$, $\mathbf{Adv}^{\mathsf{1-KE-Sec}}_{\Pi,\mathcal{A}}(\lambda)$ is negligible.*

Multi-User Extension of Our Notion. While we defined anonymous key-exchange in the single-user setting, we stress that the definition easily extends to the multi-user setting. The reason is that in our notion there is only one keyed user, T. So, when considering the multi-user setting with keyed users $\mathsf{T}_1, \ldots, \mathsf{T}_n$, we can assume that an adversary attacking a given T_j could simulate the keys of all remaining users $\mathsf{T}_i \neq \mathsf{T}_j$. In contrast, such an extension is not equally straightforward in 2-KE, where, for example, the adversary could choose arbitrary keys for one of the two parties in the challenge session. We also refer the interested reader to [36] for a discussion on the multi-user extension of 1-KE.

Single-Challenge vs. Multiple Challenges. Similarly to CCA-secure encryption and other privacy primitives, our attacker has only a single challenge session. Using standard hybrid argument, this is asymptotically equivalent to the multi-challenge extension of our notion (with all challenge sessions sharing the same challenge bit b, which is also known as the "left-or-right oracle" [2]).

We stress, however, that *single-challenge does not mean single oracle access to* T. Indeed, as was the case in all our interactive notions so far, the attacker $\mathcal{A}^{\mathsf{T}}$ can start *arbitrarily many interleaved sessions with the keyed user* T, both before and after the (single) challenge K_b. In particular, any 1-KE protocol where one can recover the secret key tk given (multiple) oracle access to T will never be secure according to our definition, as then the attacker will trivially win the (single) challenge session by simulating honest T.

Relation with Existing Definitions. As we mentioned earlier in this section, the notion of 1-KE has been considered in prior work with different definitions. Notably, two recent works [33] and [36] use a definition (Server only Authenticated and Confidential Channel Establishment – SACCE) which formally captures whether a party accepts or not in a protocol session, and requires that the adversary $\mathcal{A}$ should not let the party accept if $\mathcal{A}$ does not correctly relay messages. If we compare our security definition of 1-KE given above and the SACCE notion, we then observe the following main facts. (i) Our notion of ping-pong is stronger than the notion of matching transcripts used in SACCE in that ping-pong takes into account the timing of the messages included in the transcripts. (ii) While 1-KE and SACCE are basically equivalent w.r.t. capturing the authenticity property, they instead differ w.r.t. confidentiality. In particular, our notion aims to capture indistinguishability of the keys, whereas SACCE aims to capture the security of the channel built by using the established session key. As observed in [33], the latter security notion is weaker than mere session key indistinguishability, and might thus be realized from weaker assumptions.

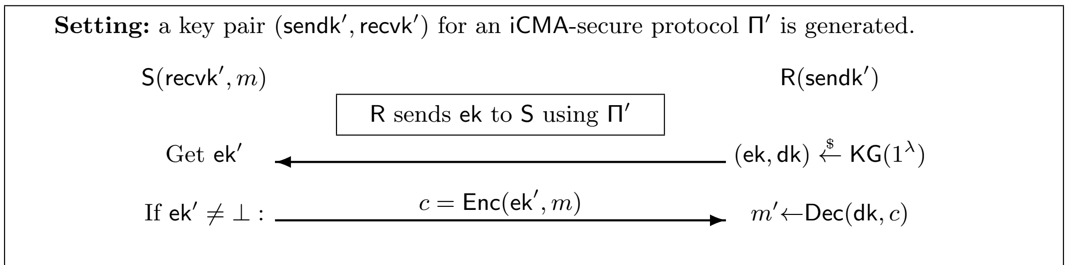

Setting: a key pair $(\mathsf{sendk}', \mathsf{recvk}')$ for an iCMA-secure protocol Π' is generated.
$\mathsf{S}(\mathsf{recvk}', m)$
$\mathsf{R}(\mathsf{sendk}')$
R sends ek to S using Π'
Get ek'
$(\mathsf{ek}, \mathsf{dk}) \xleftarrow{\$} \mathsf{KG}(1^\lambda)$
If $\mathsf{ek}' \neq \perp$:
$c = \mathsf{Enc}(\mathsf{ek}', m)$
$m' \leftarrow \mathsf{Dec}(\mathsf{dk}, c)$

Fig. 1. iCCA protocol from iCMA-secure PKMA and *1-bounded*-$\mathsf{IND{-}CCA}$ encryption.

3 Basic Constructions

In this section we propose realizations of message transmission protocols satisfying our iCCA and iCMA security notions, as well as realizations of anonymous key-exchange based on iCCA and iCMA security. Interestingly, our constructions show that iCCA security is implied by the iCMA and $\mathsf{IND{-}CPA}$ notions, whereas (somehow vice-versa) iCMA security is directly implied by iCCA. Our results thus show that in the interactive setting—and with a minimum level of interaction (i.e., 2 rounds) indeed!—the notions of confidentiality and authenticity present somewhat surprising and interesting relations unknown in the non-interactive case.

3.1 iCCA Encryption from IND−CPA Encryption and iCMA PKMA

Our result is a simple interactive encryption protocol that is based on a PKMA protocol Π' and a public key encryption scheme $\mathcal{E}$. The idea is simple and is illustrated in Figure 1: the receiver sends a "fresh" public key ek authenticated using Π', and the sender encrypts the message using ek. As we show in our theorem below, the PKMA protocol has to be iCMA-secure, while the public key encryption needs only to be *1-bounded*-$\mathsf{IND{-}CCA}$-secure. Concretely, Π' can be a strongly unforgeable signature and $\mathcal{E}$ can be constructed using a $\mathsf{IND{-}CPA}$-secure encryption (as shown by Cramer et al. [14] and recalled in Appendix C), thus yielding an *optimal* 2-round encryption protocol that is iCCA-secure based only on $\mathsf{IND{-}CPA}$ security. A more precise description follows.

Let $\Pi' = (\mathsf{Setup}', \mathsf{S}', \mathsf{R}')$ be a PKMA protocol, and $\mathcal{E} = (\mathsf{KG}, \mathsf{Enc}, \mathsf{Dec})$ be a (non-interactive) public key encryption scheme. We build protocol $\Pi = (\mathsf{Setup}, \mathsf{S}, \mathsf{R})$ as follows:

$\mathsf{Setup}(1^\lambda)$**:** run $(\mathsf{sendk}', \mathsf{recvk}') \xleftarrow{\$} \mathsf{Setup}'(1^\lambda)$ and output $\mathsf{sendk} = \mathsf{recvk}'$ and $\mathsf{recvk} = \mathsf{sendk}'$.

$\mathsf{S}(\mathsf{sendk}, m)$**:** first run the PKMA protocol Π' with R by playing the role of the receiver, i.e., S runs $\mathsf{R}'(\mathsf{recvk}')$. If Π' terminates correctly with output ek, then send $c = \mathsf{Enc}(\mathsf{ek}, m)$ to R. Otherwise, stop running.

$\mathsf{R}(\mathsf{recvk})$**:** generate $(\mathsf{ek}, \mathsf{dk}) \xleftarrow{\$} \mathsf{KG}(1^\lambda)$, run $\mathsf{S}'(\mathsf{sendk}', \mathsf{ek})$ to send ek with authenticity to S, and keep dk as private information. After Π' terminates, on input a message c from S, the receiver algorithm computes $m \leftarrow \mathsf{Dec}(\mathsf{dk}, c)$ and returns the message m as its private output.

Theorem 1. *If $\mathcal{E}$ is* 1*-bounded*-$\mathsf{IND{-}CCA}$*-secure and Π' is* iCMA*-secure, then Π is* iCCA*-secure.*

Proof. Consider the experiment $\mathbf{Exp}^{\mathsf{iCCA}}_{\mathsf{iPKE},\mathcal{A}}$ in which $T_1, \ldots, T_Q$ and T^* are the transcripts of the oracle sessions and the challenge session. Each transcript T can be written as $T = (T', T_c)$ such that T' is the transcript of the PKMA protocol Π' and T_c is the portion containing the last message c. Let Forge be the event that in $\mathbf{Exp}^{\mathsf{iCCA}}_{\mathsf{iPKE},\mathcal{A}}$ the adversary $\mathcal{A}$ is not ping-pong in the subprotocol Π' (i.e., $T'_i \not\equiv T^{*\prime}$ $\forall i = 1, \ldots, Q$), and that in the challenge session the sender accepts (i.e., R' terminates correctly returning $\mathsf{ek}^* \neq \perp$). It is not hard to see that

$$\mathbf{Adv}^{\mathsf{iCCA}}_{\mathcal{A},\Pi}(\lambda) \leq \left| \Pr[\mathbf{Exp}^{\mathsf{iCCA}}_{\mathsf{iPKE},\mathcal{A}} = 1 \mid \overline{\mathsf{Forge}}] - \frac{1}{2} \right| + \Pr[\mathsf{Forge}].$$

The security of our protocol then follows from showing that: (1) Forge occurs with negligible probability under the assumption that Π' is iCMA-secure, and (2) if Forge does not occur, then any adversary winning in $\mathbf{Exp}^{\mathsf{iCCA}}_{\mathsf{iPKE},\mathcal{A}}$ can be used to break the 1-bounded-$\mathsf{IND{-}CCA}$ security of the encryption scheme. We formally prove these facts in the following claims.

Claim 1 *If* Π' *is* iCMA*-secure, then* $\Pr[\mathsf{Forge}]$ *is negligible.*

Claim 2 *If* $\mathcal{E}$ *is 1-bounded* $\mathsf{IND{-}CCA}$*-secure, then* $\left|\Pr[\mathbf{Exp}^{\mathsf{iCCA}}_{\mathsf{iPKE},\mathcal{A}} = 1 \mid \overline{\mathsf{Forge}}] - \frac{1}{2}\right|$ *is negligible.*

The proof of Claim 1 is straightforward and appears in Appendix F.1. The proof of Claim 2 is given below.

Proof (Claim 2). Assume by contradiction there exists an efficient adversary $\mathcal{A}$ such that

$$\left|\Pr[\mathbf{Exp}^{\mathsf{iCCA}}_{\Pi,\mathcal{A}}(\lambda) = 1 \mid \overline{\mathsf{Forge}}] - \frac{1}{2}\right| \geq \epsilon$$

is non-negligible. Then we build a PPT adversary $\mathcal{B}$ that has non-negligible advantage in breaking the 1-bounded-$\mathsf{IND{-}CCA}$ security of the encryption scheme $\mathcal{E} = (\mathsf{KG}, \mathsf{Enc}, \mathsf{Dec})$.

On input a public key $\bar{\mathsf{ek}}$, $\mathcal{B}$ works as follows:

- Generate a key pair $\mathsf{sendk}', \mathsf{recvk}') \xleftarrow{\$} \mathsf{Setup}'(1^\lambda)$, set $\mathsf{sendk} = \mathsf{recvk}'$ and run $\mathcal{A}(\mathsf{sendk})$.
- Initialize a counter $j \leftarrow 0$ for the number of new sessions opened by $\mathcal{A}$ with R during the experiment.
- Choose a random index $\mu \xleftarrow{\$} \{1, \ldots, Q\}$ where $Q = poly(\lambda)$ is an upper bound on the number of sessions opened by $\mathcal{A}$ with the oracle receiver R. Since Forge does not occur, $\mathcal{A}$ is ping-pong in the subprotocol Π' and thus in the challenge session $\mathcal{A}$ will re-use one of the public keys $\mathsf{ek}_1, \ldots, \mathsf{ek}_Q$ obtained by the oracle R. Therefore, μ represents a guess for the index of such public key ek_μ.
- For every oracle query asking to interact with a new copy of R: first, increment j by 1. Now, let j be the index of the current query. If $j \neq \mu$, then $\mathcal{B}$ generates a new encryption key pair $\mathsf{ek}_j, \mathsf{dk}_j) \xleftarrow{\$} \mathsf{KG}(1^\lambda)$, and runs $\mathsf{S}'(\mathsf{sendk}', \mathsf{ek}_j)$ to simulate all the oracle answers of this session corresponding to the run of Π'. Otherwise, if $j = \mu$, it sets $\mathsf{ek}_\mu = \bar{\mathsf{ek}}$, and runs $\mathsf{S}'(\mathsf{sendk}', \bar{\mathsf{ek}})$.
- When $\mathcal{A}$ queries R with the last protocol message c_i on the i-th opened session: let ek_i be the public key previously generated in the above step. If $i \neq \mu$, then $\mathcal{B}$ knows the corresponding decryption key dk_i, ($\mathcal{B}$ generated it by itself), and it answers by computing $m_i \leftarrow \mathsf{Dec}(\mathsf{dk}, c_i)$. If $i = \mu$, $\mathcal{B}$ asks c_i to the decryption oracle, obtains $\tilde{m}$, and answers $\tilde{m}$. Notice that for $i = \mu$ such a query can occur only once, as c_i is the last message of the protocol (the session is then closed).
- Let (m_0, m_1) be the message pair returned by the adversary $\mathcal{A}$. Then the challenge session starts and $\mathcal{A}$ is expected to "speak first", by sending ek^* using Π'. Since Forge does not occur, we have that either $\mathsf{ek}^* = \bot$, or $\mathsf{ek}^* \neq \bot$ and $\mathcal{A}$ is ping-pong, i.e., $\mathsf{ek}^* \in \{\mathsf{ek}_1, \ldots, \mathsf{ek}_Q\}$. In the first case $\mathcal{B}$ returns an error (this is a correct simulation by protocol's construction). In the second case: if $\mathsf{ek}^* \neq \mathsf{ek}_\mu$, then $\mathcal{B}$ aborts the simulation and outputs a random bit. Otherwise, it continues as described below. $\mathcal{B}$ forwards (m_0, m_1) to its challenger, gets back a ciphertext c^*, and sends c^* to $\mathcal{A}$ in the challenge session.
- $\mathcal{B}$ answers oracle queries as before.
- Finally, let b' be $\mathcal{A}$'s output, then $\mathcal{B}$ outputs the same bit.

Let Abort be the event that $\mathcal{B}$ aborts during the experiment. If Abort occurs, then $\Pr[\mathbf{Exp}^{1\mathsf{-IND-CCA}}_{\mathcal{E},\mathcal{B}}(\lambda) = 1 \mid \mathsf{Abort}] = 1/2$. Moreover, as long as Abort does not occur the distribution of the public keys simulated by $\mathcal{B}$ is identical to the one in the real experiment, and thus the index μ is perfectly hidden. Hence, we have that $\Pr[\overline{\mathsf{Abort}}] = 1/Q$ and

$$\mathbf{Adv}^{1\mathsf{-IND-CCA}}_{\mathcal{E},\mathcal{B}}(\lambda) \geq \frac{1}{Q} \cdot \left|\Pr[\mathbf{Exp}^{1\mathsf{-IND-CCA}}_{\mathcal{E},\mathcal{B}}(\lambda) = 1 \mid \overline{\mathsf{Abort}}] - \frac{1}{2}\right|.$$

To complete the proof we show that in the case Abort does not occur $\mathcal{B}$'s simulation of the iCCA game to $\mathcal{A}$ is perfect. In particular, we show that as long as $\mathcal{A}$ is not ping-pong (as it must be by definition of iCCA security) $\mathcal{B}$ can answer correctly to all queries made by $\mathcal{A}$. Precisely, the tricky case that needs to be checked is that $\mathcal{B}$ can answer with the correct decryption when the adversary sends the last message on sessions that were already opened. Let $T = (T', T_c)$ be the transcript of the queried session where ek is the corresponding public key and c is the ciphertext sent by $\mathcal{A}$, and let $T^* = (T^{*'}, T_c^*)$ be the transcript of the challenge session. If $T' \not\equiv T^{*'}$, it essentially means that $\mathcal{B}$ generated ek and thus it can decrypt. If $T' \equiv T^{*'}$, since $\mathcal{A}$ is not ping-pong, then it must be that either (I) $c \neq c^*$ (in this case $\mathcal{B}$ forwards c to the decryption oracle), or (II) $c = c^*$ and the corresponding timestamps are not alternating, i.e., $t_n^* > t_n$, that is c was sent before $\mathcal{B}$ asked (and sent) the challenge ciphertext. Thus $\mathcal{B}$ could have asked c to its decryption oracle recall that in the 1-bounded-$\mathsf{IND-CCA}$ game such decryption query is legal).

In conclusion, we obtain: $\mathbf{Adv}_{\mathcal{E},\mathcal{B}}^{1-\mathsf{IND-CCA}}(\lambda) \geq \frac{\epsilon}{Q}$. □

As an interesting consequence of Theorem 1 we obtain an equivalence between the notions of $\mathsf{IND-CPA}$ and iCCA security:

Corollary 1. *2-round-iCCA encryption exists if and only if (non-interactive) $\mathsf{IND-CPA}$ PKE exists.*

Proof. The first direction ($\mathsf{IND-CPA} \Rightarrow$ 2-round-iCCA) follows by observing that: (i) 1-bounded-$\mathsf{IND-CCA}$ is implied by $\mathsf{IND-CPA}$ security (see [14] and Appendix C); (ii) iCMA security can be realized using digital signatures, and thus from one-way functions (see our Corollary 3). The second direction follows from the following simple Lemma:

Lemma 1. *2-round-$\mathsf{iCCA} \Rightarrow \mathsf{IND-CPA}$*

Proof. Let $\Pi = (\mathsf{Setup}, \mathsf{S}, \mathsf{R})$ be a 2-round iCCA protocol (recall that wlog we assume that R speaks first). We construct a non-interactive encryption scheme $\mathcal{E} = (\mathsf{KG}, \mathsf{Enc}, \mathsf{Dec})$ from Π as follows:

$\mathsf{KG}(1^\lambda)$**:** run $(\mathsf{sendk}, \mathsf{recvk}) \xleftarrow{\$} \mathsf{Setup}(1^\lambda)$, and $(R; \rho) \xleftarrow{\$} \mathsf{R}(\mathsf{recvk})$ where we denote with R the message sent by R, and by ρ its private coins. Output $\mathsf{ek} = (\mathsf{sendk}, R)$ and $\mathsf{dk} = (\mathsf{recvk}, \rho)$.

$\mathsf{Enc}(\mathsf{ek}, m)$**:** run the sender algorithm $\mathsf{S}(\mathsf{sendk}, m)$ with input message R. Let C be the message generated by S. Output C as the ciphertext.

$\mathsf{Dec}(\mathsf{dk}, C)$**:** run $m \leftarrow \mathsf{R}(\mathsf{DK})$ with input message C and private random coins ρ, and output m.

Proof of Security. Assume there exists an efficient $\mathcal{A}$ that breaks the $\mathsf{IND-CPA}$ security of $\mathcal{E}$ with non-negligible advantage ϵ, i.e., $\mathbf{Adv}_{\mathcal{A},\mathcal{E}}^{\mathsf{IND-CPA}}(\lambda) \geq \epsilon$. We build an adversary $\mathcal{B}$ that uses $\mathcal{A}$ to obtain non-negligible advantage in breaking the iCCA security of protocol Π. $\mathcal{B}$ is run on input the public sender key sendk and has oracle access to R. First, $\mathcal{B}$ queries $\mathsf{R}(\mathsf{recvk})$ to start a new session, and it obtains R. $\mathcal{B}$ sets $\mathsf{ek} = (\mathsf{sendk}, R)$ and runs $(m_0, m_1) \xleftarrow{\$} \mathcal{A}(\mathsf{ek})$. $\mathcal{B}$ outputs the same pair of messages, and then runs the challenge session with $\mathsf{S}(\mathsf{sendk}, m_b)$, to which it sends R and receives C^* back. $\mathcal{B}$ finally runs $b' \xleftarrow{\$} \mathcal{A}(C^*)$ and outputs the same bit b'. It is easy to see that the simulation of the $\mathsf{IND-CPA}$ experiment is perfect, and thus $\mathbf{Adv}_{\mathcal{B},\Pi}^{\mathsf{iCCA}}(\lambda) = \mathbf{Adv}_{\mathcal{A},\mathcal{E}}^{\mathsf{IND-CPA}}(\lambda) \geq \epsilon$. □

For the case of non-interactive public key encryption, it is an open problem to understand whether the notion of $\mathsf{IND-CPA}$ security implies the one of $\mathsf{IND-CCA}$ security (in a fully black-box sense), and there have been provided some evidences that it may not be the case [28]. In contrast, our result shows that by adding a *single* round of communication the basic notion of $\mathsf{IND-CPA}$-security is sufficient to realize 2-round iCCA-secure encryption.

It is worth noting that a 3-round encryption protocol based on $\mathsf{IND-CPA}$ encryption was earlier proposed by Dolev, Dwork and Naor [21]. While the protocol in [21] can be proven in our formalization of interactive encryption (see Appendix B), our contribution is a protocol which is optimal (2 rounds) and is based on the same weaker assumption ($\mathsf{IND-CPA}$ security).

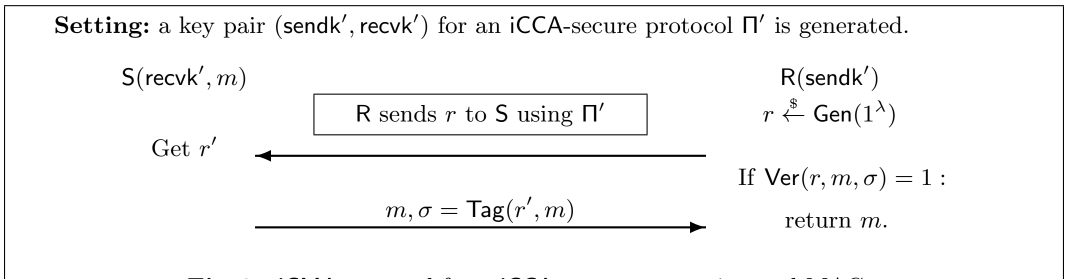

Setting: a key pair $(\mathsf{sendk}', \mathsf{recvk}')$ for an iCCA-secure protocol Π' is generated.
$\mathsf{S}(\mathsf{recvk}', m)$
$\mathsf{R}(\mathsf{sendk}')$
R sends r to S using Π'
$r \xleftarrow{\$} \mathsf{Gen}(1^\lambda)$
Get r'
If $\mathsf{Ver}(r, m, \sigma) = 1$:
$m, \sigma = \mathsf{Tag}(r', m)$
return m.

Fig. 2. iCMA protocol from iCCA-secure encryption and MACs.

3.2 iCMA-secure PKMA from iCCA security

We show how to realize iCMA-secure message transmission from any message transmission protocol that is iCCA-secure, and any strongly unforgeable MAC. The protocol is based on an old idea of realizing public key authentication via CCA-secure encryption and message authentication codes. The basic idea is that the authenticator shows its ability to decrypt: the verifier encrypts a MAC key r for the authenticator, who decrypts and sends back the MAC of message m using the key r. This protocol (briefly sketched in Figure 2) implicitly appeared first in [35] and was proven secure in [1]. Here we generalize this construction in the framework of our definitions, i.e., by using (possibly interactive) iCCA-secure encryption in place of (non-interactive) $\mathsf{IND-CCA}$-secure encryption. Furthermore, in Appendix E we generalize a 3-round protocol earlier proposed by Dolev, Dwork and Naor [21] that is based only on iCCA security.

Let $\mathsf{MAC} = (\mathsf{Gen}, \mathsf{Tag}, \mathsf{Ver})$ be a strongly unforgeable MAC with message space $\mathcal{M}$ and key space $\mathcal{K}$ (see Appendix A.4 for a formal definition of MAC), and let $\Pi' = (\mathsf{Setup}', \mathsf{S}', \mathsf{R}')$ be an encryption protocol whose message space is $\mathcal{K}$. We build a PKMA protocol $\Pi = (\mathsf{Setup}, \mathsf{S}, \mathsf{R})$ as follows.

$\mathsf{Setup}(1^\lambda)$**:** run $(\mathsf{sendk}', \mathsf{recvk}') \xleftarrow{\$} \mathsf{Setup}'(1^\lambda)$ and output $\mathsf{recvk} = \mathsf{sendk}'$ and $\mathsf{sendk} = \mathsf{recvk}'$. The message space of the protocol Π is the message space $\mathcal{M}$ of the MAC.

$\mathsf{S}(\mathsf{sendk}, m)$**:** run the encryption protocol Π' with R with reversed roles, i.e., S runs the receiver algorithm R' of Π' while R will run the sender algorithm S'. Let r be the output of R' at the end of the run of protocol Π'. Then S sends $(m, \sigma = \mathsf{Tag}(r, m))$ to R as the last message.

$\mathsf{R}(\mathsf{recvk})$**:** generate a fresh MAC key $r \xleftarrow{\$} \mathsf{Gen}(1^\lambda)$ and send r to S using the encryption protocol Π', i.e., R runs $\mathsf{S}'(\mathsf{sendk}', r)$. Once the encryption protocol is over, R waits for a message (m, σ') from S. If $\mathsf{Ver}(r, m, \sigma') = 1$ then R outputs m. Otherwise, it outputs $\perp$.

We can now state the following theorem.

Theorem 2. *If* Π' *is* iCCA*-secure and* MAC *is strongly-unforgeable, then* Π *is an* iCMA*-secure message transmission protocol.*

Proof. To prove the theorem we define the following hybrid games and we denote with G_i the event that the outcome of Game i, run with $\mathcal{A}$, is 1.

Game 0: this is the real iCMA game.

Game 1: this is the same as Game 0 except that in the challenge session the receiver generates a random key $r^* \xleftarrow{\$} \mathsf{Gen}(1^\lambda)$ but runs the encryption protocol Π' with the message 0, i.e., R runs $\mathsf{S}'(\mathsf{sendk}', 0)$. We point out that the receiver keeps using r^* as the random string associated with the run of the encryption protocol in the challenge session. This means that:

1. the receiver uses r^* to check whether $\mathsf{Ver}(r^*, m^*, \sigma^*) = 1$, where (m^*, σ^*) is the forgery returned by the adversary (i.e., the last message of $\mathcal{A}$ in the challenge session);

Setting: $(\mathsf{ek}, \mathsf{dk})$ a key-pair for a labeled encryption scheme $(\mathsf{KG}, \mathsf{Enc}^L, \mathsf{Dec}^L)$ is generated.

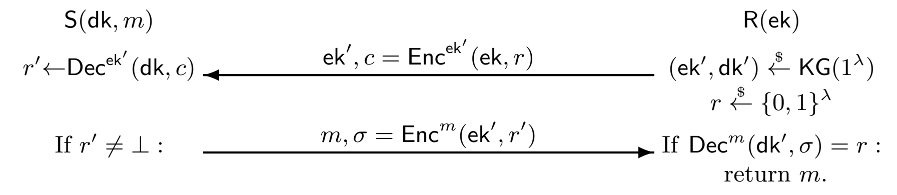

$\mathsf{S}(\mathsf{dk}, m)$
$\mathsf{R}(\mathsf{ek})$
$r' \leftarrow \mathsf{Dec}^{\mathsf{ek}'}(\mathsf{dk}, c)$
$\mathsf{ek}', c = \mathsf{Enc}^{\mathsf{ek}'}(\mathsf{ek}, r)$
$(\mathsf{ek}', \mathsf{dk}') \xleftarrow{\$} \mathsf{KG}(1^\lambda)$
$r \xleftarrow{\$} \{0,1\}^\lambda$
If $r' \neq \bot$:
$m, \sigma = \mathsf{Enc}^m(\mathsf{ek}', r')$
If $\mathsf{Dec}^m(\mathsf{dk}', \sigma) = r$:
return m.

Fig. 3. 2-round iCMA protocol from labeled $\mathsf{IND-CCA}$-secure encryption.

2. the receiver uses r^* to compute $\sigma = \mathsf{Tag}(r^*, m)$ in all those queries to $\mathsf{S}(\mathsf{sendk}, m)$ where the plaintext is m and all previous messages of the session are exactly the same as those of the challenge session (basically, the adversary replayed all messages of the encryption protocol but asked for a different plaintext $m \neq m^*$).

Via a straightforward reduction to the iCCA-security of the encryption protocol, it is possible to show that there exists an adversary $\mathcal{B}$ such that:

$$|\Pr[G_0] - \Pr[G_1]| \leq 2 \cdot \mathbf{Adv}^{\mathsf{iCCA}}_{\Pi', \mathcal{B}}(\lambda).$$

Finally, if we consider Game 1 it is not hard to see that by the strong unforgeability of the MAC we have that $\Pr[G_1] \leq \mathbf{Adv}^{\mathsf{suf\text{-}cmva}}_{\mathcal{A}, \mathsf{MAC}}(\lambda)$. □

Remark 1. While the protocol uses a fresh MAC key for every session, we stress that a one-time MAC (e.g., a pairwise independent hash function is not sufficient to prove iCMA security. Intuitively, the reason is that the adversary may fully-replay the first portion of the protocol (i.e., the one related to Π') from the challenge session to many copies of the sender, each initialized with a different plaintext $m \neq m^*$, thus obtaining several MACs under the same key. To see this in the proof, check the description of Game 1, point 2.

If we instantiate the above construction with a 1-round (aka non-interactive) iCCA-secure encryption scheme, and one of the constructions of MACs from $\mathsf{IND-CCA}$ encryption proposed in [19] (that have the advantage of having a 'compact' secret key), we then obtain an elegant and efficient 2-round PKMA protocol based only on $\mathsf{IND-CCA}$ security. Moreover, by directly observing the MAC of [19] and the resulting protocol, we managed to further optimize this protocol: we notice that the ephemeral secret key dk' (which is part of the MAC key with r) is only used for verification, and there is no need to encrypt it inside c; instead, we use labels to bind ek' with c. The resulting optimized protocol is presented in Figure 3.

By instantiating the result of Theorem 2 (and our optimization in Figure 3) with known constructions of CCA-secure encryption from Factoring [32], DDH [15], or CDH [50], we obtain the following corollary.

Corollary 2. *If the Factoring (resp. DDH, CDH) assumption holds, there exists an efficient 2-round PKMA.*

3.3 Secure Round Extension of Message Transmission Protocols

We discuss a generic methodology to securely increase the number of rounds of message transmission protocols. Although at a first glance this construction might not look very interesting as it decreases efficiency, it will turn out to be particularly useful to achieve our stronger notions of weak (resp. strong) replay security that we discuss in Section 4.

For these notions we will indeed show that interaction is extremely important, and more precisely that at least 2 rounds of interaction are *necessary* to achieve replay security (and at least 3 rounds

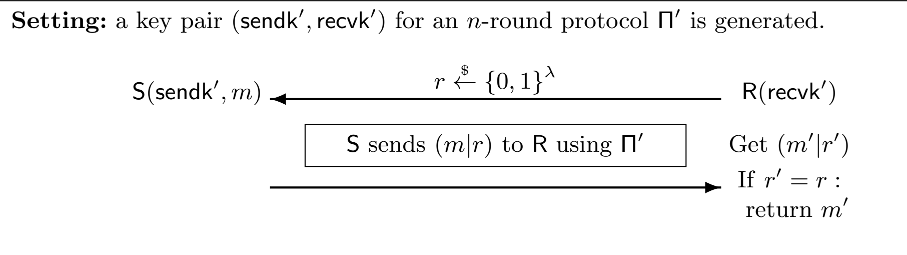

Setting: a key pair $(\mathsf{sendk}', \mathsf{recvk}')$ for an n-round protocol Π' is generated.
$\mathsf{S}(\mathsf{sendk}', m)$
$r \xleftarrow{\$} \{0,1\}^\lambda$
$\mathsf{R}(\mathsf{recvk}')$
S sends $(m|r)$ to R using Π'
Get $(m'|r')$
If $r' = r$:
return m'

Fig. 4. $(n+1)$-round protocol Π from n-round protocol Π' (S' speaks first).

are necessary for strong replay-secure encryption). The basic idea for constructing an $(n+1)$ message transmission protocol Π from an n-round Π' is sketched in Figure 4, and consists in letting the party who speaks first send a random nonce r, and then running the n-round protocol with plaintext $m|r$. Finally, the receiver will terminate correctly only if the n-round protocol returns a plaintext $m|r$, where r is the *same* nonce sent in the first message.

In the following theorem we prove that the above construction preserves iCCA and iCMA security.

Theorem 3. *For any $n \geq 1$, if Π' is an* iCCA *(resp.* iCMA*) secure n-round protocol, there exists an $(n+1)$-round protocol Π that is* iCCA *(resp.* iCMA*) secure.*

The proof of the Theorem appears in Appendix F.2 and F.3. Here we pause to show the following interesting corollary, where (a) is obtained by applying our observation that 1-round iCMA-secure PKMA is equivalent to strongly unforgeable signatures, and (b) follows from our Theorem 1.

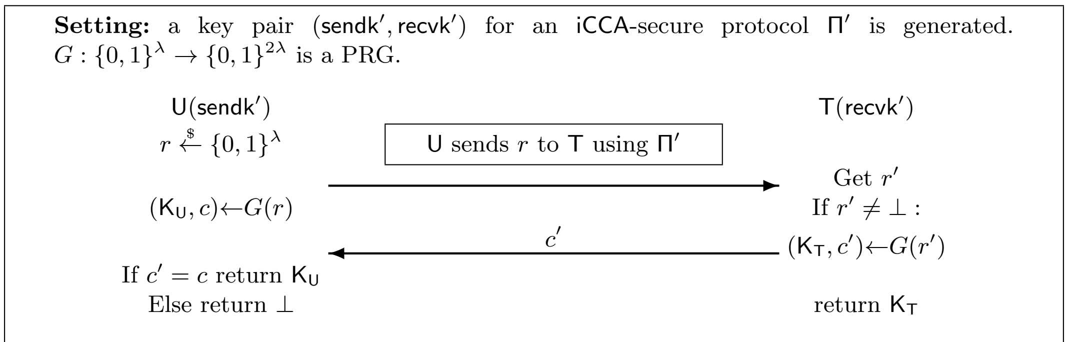
Corollary 3. *(a) For any $n \geq 1$, one-way functions are sufficient to build an n-round PKMA protocol that is* iCMA*-secure. (b) For any $n \geq 2$,* IND−CPA*-secure PKE is sufficient to build an n-round encryption protocol that is* iCCA*-secure.*

A formal description of our round-extension construction follows. If $\Pi' = (\mathsf{Setup}', \mathsf{S}', \mathsf{R}')$ is an n-round message transmission protocol, then we construct an $(n+1)$-round protocol $\Pi = (\mathsf{Setup}, \mathsf{S}, \mathsf{R})$ as follows. The key generation algorithm is the same, i.e., $\mathsf{Setup} = \mathsf{Setup}'$ and $\mathsf{sendk} = \mathsf{sendk}', \mathsf{recvk} = \mathsf{recvk}'$. To build algorithms S and R we distinguish two cases according to which party speaks first in Π': S' or R'.

1. R **speaks first.** In the new protocol Π', it will be S' who will speak first.
   $\mathsf{S}'(\mathsf{sendk}', m)$**:** first, choose $r \xleftarrow{\$} \{0,1\}^\lambda$ and send r to R'. To react to the later messages, run $\mathsf{S}(\mathsf{sendk}, m|r)$ (i.e., on the message obtained by concatenating m and r).
   $\mathsf{R}'(\mathsf{recvk}')$**:** wait for the message r from S'. Next, store r and run $\mathsf{R}(\mathsf{recvk})$. At the end of a session, let $m = (m'|r')$ be the the message returned by R. If $r' = r$, then return m', otherwise, output $\perp$.
2. S **speaks first.** In the new protocol Π', it will be R' who will speak first.
   $\mathsf{S}'(\mathsf{sendk}', m)$**:** wait for the message r from R' and then run $\mathsf{S}(\mathsf{sendk}, m|r)$.
   $\mathsf{R}'(\mathsf{recvk}')$**:** first, choose a random $r \xleftarrow{\$} \{0,1\}^\lambda$ and send r to S'. Next, run $\mathsf{R}(\mathsf{recvk})$, and let $m = (m'|r')$ be the the message returned by R at the end of its execution. If $r' = r$, then return m', otherwise output $\perp$.

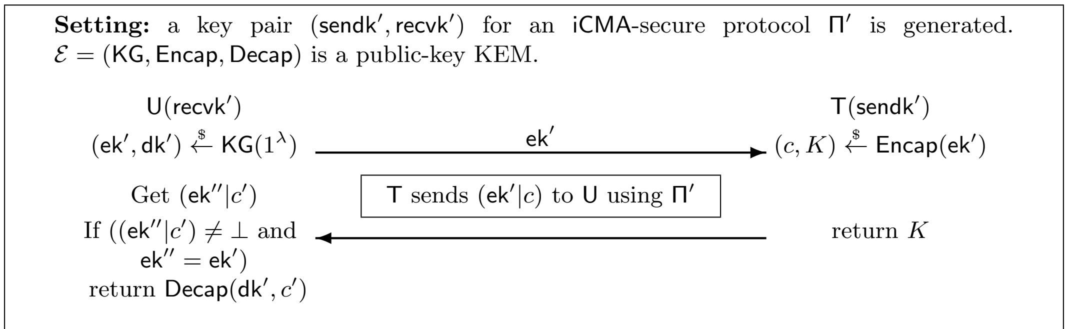
3.4 1-KE Protocols based on iCCA and iCMA Security

In this section we show two simple realizations of anonymous key-exchange. The first one (described in Figure 5) uses an iCCA-secure protocol Π' and a pseudorandom generator (see Appendix A.1 for the PRG definition).[19] Our second construction of 1-KE (described in Figure 6) uses an IND−CPA-secure key encapsulation mechanism (see Appendix A for the definition of KEM) and an iCMA-secure protocol Π'.


Setting: a key pair $(\mathsf{sendk}', \mathsf{recvk}')$ for an iCCA-secure protocol Π' is generated. $G : \{0,1\}^\lambda \rightarrow \{0,1\}^{2\lambda}$ is a PRG.

$\mathsf{U}(\mathsf{sendk}')$
$\mathsf{T}(\mathsf{recvk}')$
$r \overset{\$}{\leftarrow} \{0,1\}^\lambda$
U sends r to T using Π'
Get r'
$(\mathsf{K_U}, c) \leftarrow G(r)$
If $r' \neq \perp$:
c'
$(\mathsf{K_T}, c') \leftarrow G(r')$
If $c' = c$ return $\mathsf{K_U}$
Else return $\perp$
return $\mathsf{K_T}$

Fig. 5. 1-KE from iCCA-secure encryption.


Setting: a key pair $(\mathsf{sendk}', \mathsf{recvk}')$ for an iCMA-secure protocol Π' is generated. $\mathcal{E} = (\mathsf{KG}, \mathsf{Encap}, \mathsf{Decap})$ is a public-key KEM.

$\mathsf{U}(\mathsf{recvk}')$
$\mathsf{T}(\mathsf{sendk}')$
$(\mathsf{ek}', \mathsf{dk}') \overset{\$}{\leftarrow} \mathsf{KG}(1^\lambda)$
ek'
$(c, K) \overset{\$}{\leftarrow} \mathsf{Encap}(\mathsf{ek}')$
Get $(\mathsf{ek}''|c')$
T sends $(\mathsf{ek}'|c)$ to U using Π'
If $((\mathsf{ek}''|c') \neq \perp$ and $\mathsf{ek}'' = \mathsf{ek}')$
return K
return $\mathsf{Decap}(\mathsf{dk}', c')$

Fig. 6. 1-KE from iCMA-secure PKMA and $\mathsf{IND-CPA}$-secure KEM.

The security of these protocols is proven via the following theorems:

Theorem 4. *If Π' is* iCCA*-secure, and G is a pseudo-random generator, then the protocol Π in Figure 5 is a secure 1-KE.*

Proof. To prove the security of Π we define the following hybrid games:

Game 0: this is the real $\mathbf{Exp}^{\mathsf{1-KE-Sec}}_{\Pi,\mathcal{A}}(\lambda)$ experiment.

Game 1: this is the same as Game 0 except that in the challenge session U picks a random string $r^* \overset{\$}{\leftarrow} \{0,1\}^\lambda$, but runs the encryption protocol Π' sending message 0. Yet, U still computes $(c^*, \mathsf{K_T}^*) \leftarrow G(r^*)$ and uses these values as in Game 0 (i.e., to check if $c^* = c'$ and, if so, to define $K_0 = \mathsf{K_T}^*$). Moreover, if the oracle T is queried to obtain the last message (i.e., c') on a session whose first part is equal to the run of the encryption protocol in the challenge session, then the oracle returns c^*.
Via a simple reduction to the iCCA-security of Π', it is possible to show that there exists $\mathcal{B}$ such that: $|\Pr[G_0] - \Pr[G_1]| \leq 2 \cdot \mathbf{Adv}^{\mathsf{iCCA}}_{\Pi',\mathcal{B}}(\lambda)$.

Game 2: this is the same as Game 1 except that $c^*, \mathsf{K_T}^* \overset{\$}{\leftarrow} \{0,1\}^\lambda$ are generated in a truly random way, instead of computing them as $(c^*, \mathsf{K_T}^*) \leftarrow G(r^*)$.
It is not hard to see that under the assumption that G is a PRG Game 2 is computationally indistinguishable from Game 1.

Game 3: Let Forge be the event that in Game 2 the challenge session completes with $K_0 \neq \perp$ while $\mathcal{A}$ is not ping-pong. Then Game 3 proceeds exactly as Game 2, except that if Forge occurs, then the game outputs 0 (instead of 1, as is done in $\mathbf{Exp}^{\mathsf{1-KE-Sec}}_{\Pi,\mathcal{A}}(\lambda)$ and in Game 2). Hence, Game 3 and Game 2 are identically distributed unless Forge occurs, i.e., $|\Pr[G_2] - \Pr[G_3]| \leq \Pr[\mathsf{Forge}]$.
We observe that the event Forge essentially occurs if $\mathcal{A}$ sends the correct value c^* to the challenger. However, in Game 2 c^* is chosen uniformly at random in $\{0,1\}^\lambda$. Hence, if $\mathcal{A}$ is not ping-pong, we can bound $\Pr[\mathsf{Forge}] \leq 1/2^\lambda$.

[19] The PRG is actually used only for efficiency reasons. Otherwise, U can directly send random $\mathsf{K_U}, c$ using Π'.

To conclude the proof, if we analyze the probability that Game 3 outputs 1 (hence, Forge does not occur), then $\mathcal{A}$'s view of Game 3 in the second part (i.e., when $\mathcal{A}$ is given K_b) is exactly the same no matter which is the bit b (both session keys K_0 and K_1 are indeed randomly chosen or they are both $\perp$). Hence, $\mathcal{A}$ has probability 1/2 of guessing the right $b' = b$. □

Theorem 5. *If Π' is iCMA-secure, and $\mathcal{E}$ is an $\mathsf{IND{-}CPA}$-secure KEM, then the protocol Π in Figure 6 is a secure 1-KE.*

Proof. To prove the security of Π we define the following simple hybrid games:

Game 0: This is the same as experiment $\mathbf{Exp}^{\mathsf{1-KE-Sec}}_{\Pi,\mathcal{A}}(\lambda)$.

Game 1: Consider experiment Game 0, and let $T^* = \langle(\mathsf{ek}', t_1^*), T^{*'}\rangle$ be the transcript of the challenge session, where $T^{*'}$ is the portion of transcript which corresponds to the run of the protocol Π' (in the challenge session). Similarly, consider the transcripts T_i of all the oracle sessions and write $T_i = \langle(\mathsf{ek}'_i, t_1^i), T'_i\rangle$. Let Forge be the event that in the challenge session the protocol Π' completes correctly but $\mathcal{A}$ is not ping-pong.
Game 1 proceeds as Game 0 except that, if Forge occurs, then Game 1 outputs 0 (instead of 1).
Clearly, Game 1 is identical to Game 0 unless Forge occurs, i.e., $|\Pr[G_0] - \Pr[G_1]| \leq \Pr[\mathsf{Forge}]$. Under the assumption that Π' is iCMA-secure, one can easily prove that $\Pr[\mathsf{Forge}]$ is negligible.

So, we are left with bounding $|\Pr[G_1] - 1/2|$. Let us split the event G_1 around the event $K_0 = \perp$. If $K_0 = \perp$, it is easy to see that $\Pr[G_1]$ is at most 1/2. On the other hand, if $K_0 \neq \perp$ then recall that Game 1 can output 1 only when $\mathcal{A}$ is ping-pong (i.e., Forge does not occur). Then we argue that under the assumption that the scheme $\mathcal{E}$ is $\mathsf{IND{-}CPA}$-secure we have that $p_1 = |\Pr[G_1 \wedge K_0 \neq \perp] - 1/2|$ is negligible. The reduction is straightforward. We provide it below for completeness.

Assume there exists $\mathcal{A}$ such that $p_1 \geq \epsilon$ is non-negligible, then we construct an adversary $\mathcal{B}$ which has non-negligible advantage ϵ/Q against the $\mathsf{IND{-}CPA}$ security of $\mathcal{E}$ (where Q is an upper bound on the number of oracle sessions opened by $\mathcal{A}$). $\mathcal{B}$ receives the public key ek^* and works as follows. It picks two random strings $K_0^*, K_1^* \xleftarrow{\$} \{0,1\}^\lambda$, submits them to its challenger and obtains a ciphertext c^*. Moreover, it initializes a counter $j = 0$ and picks a random integer $\mu \xleftarrow{\$} \{1, \ldots, Q\}$, which represents a guess on which of the Q oracle sessions will be a ping-pong of the challenge session. For simplicity, we restrict such a choice only to oracle sessions such that the first message is ek^* (as this is necessary for $\mathcal{A}$ to be ping-pong). Next, $\mathcal{B}$ generates a pair of keys $(\mathsf{sendk}', \mathsf{recvk}') \xleftarrow{\$} \mathsf{Setup}'(1^\lambda)$ for Π' and runs $\mathcal{A}(\mathsf{recvk}')$. $\mathcal{B}$ sends ek^* as the first message in the challenge session and uses the private key sendk' to simulate the authentication in the answers to all oracle queries to T. To generate the ciphertext $\mathcal{B}$ proceeds as follows. If the adversary sends a public key $\mathsf{ek}_i \neq \mathsf{ek}^*$, $\mathcal{B}$ simply chooses a random $K_i \xleftarrow{\$} \{0,1\}^\lambda$, encrypts $c_i \xleftarrow{\$} \mathsf{Enc}(\mathsf{ek}_i, K_i)$, and runs Π' on (ek_i, c_i). If $\mathsf{ek}_i = \mathsf{ek}^*$, $\mathcal{B}$ first increments j. If $j = \mu$, then $\mathcal{B}$ proceeds by using the challenge ciphertext c^*. Otherwise, it chooses a random K_i and proceeds as before. Now, assume that $\mathcal{A}$ completes the challenge session, and recall that since Forge does not occur $\mathcal{A}$ is ping-pong. If $\mathcal{A}$ completes by sending c^* (i.e., μ was the right guess), then $\mathcal{B}$ returns K_0. Otherwise, if $\mathcal{A}$ does not send c^* or $\mathcal{A}$ asks to reveal the session key on the μ-th oracle session, then $\mathcal{B}$ aborts and outputs a random bit (this is essentially the case that μ was not the right guess). Finally, if there is no abort $\mathcal{B}$ returns the same bit of $\mathcal{A}$.

Notice that as long as $\mathcal{B}$ does not abort, its simulation is perfectly distributed. Thus the choice of μ is perfectly hidden, i.e., $\mathcal{B}$ does not abort with probability $1/Q$. To conclude the proof, observe that if c^* encrypts K_0, then $\mathcal{B}$ is simulating $\mathbf{Exp}^{\mathsf{1-KE-Sec}}_{\Pi,\mathcal{A}}(\lambda)$ with bit $b = 0$, whereas if c^* encrypts K_1, then $\mathcal{B}$ is perfectly simulating the distribution of $\mathbf{Exp}^{\mathsf{1-KE-Sec}}_{\Pi,\mathcal{A}}(\lambda)$ with $b = 1$. Hence, we have that $\mathcal{B}$ has advantage at least ϵ/Q against the $\mathsf{IND{-}CPA}$ security of $\mathcal{E}$. □

Concrete Instantiations. By instantiating the protocols in Figures 5 and 6 with 1- or 2-round iCCA- and iCMA-secure schemes, we obtain four efficient instantiations of 1-KE. Below we briefly mention the four resulting instantiations. Later in Section 4.4 we will analyze them with a special focus on the advanced properties of forward security and deniability.

1. Protocol of Figure 5 where the iCCA protocol Π' is a non-interactive $\mathsf{IND-CCA}$ scheme: we obtain a *2-round* 1-KE based on $\mathsf{IND-CCA}$.
2. Protocol of Figure 5 where the iCCA protocol Π' is our 2-round protocol of Section 3.1 based on $\mathsf{IND-CPA}$ security: we obtain a *3-round* 1-KE based on $\mathsf{IND-CPA}$ security.
3. Protocol of Figure 6 where the iCMA protocol Π' is a digital signature: we obtain a *2-round* 1-KE based on $\mathsf{IND-CPA}$ security. It is worth noting that when implementing the KEM with standard DH key-exchange ($\mathsf{ek}' = g^x, c = g^y, K = g^{xy}$) we essentially recover protocol A-DHKE-1 in [48]. A very similar protocol is also recovered in the recent, independent work of Maurer et al. [38].
4. Protocol of Figure 6 where the iCMA protocol Π' is the 2-round PKMA based on $\mathsf{IND-CCA}$ and MACs that we call Π_{mac}: we obtain a *2-round* 1-KE (as we can piggy-back the first round of Π_{mac} on the first round of the 1-KE).

Confirmed Encryption and Confidential Authentication. We would like to point out that it is possible to reinterpret our two constructions in a more elegant way by using intermediate notions of (necessarily interactive) encryption and authentication that we call *confirmed encryption* and *confidential authentication*. In brief, confirmed encryption is an extension of our interactive encryption notion in which the sender gets a confirmation that the receiver obtained the encrypted message, and thus accepts/rejects accordingly. Confidential authentication, instead, adds a privacy property to PKMA protocols in such a way that the receiver gets convinced about the authenticity of the transmitted message, and that no information about the message is leaked to adversaries controlling the communication channel. In Section 4.5, we provide a more detailed description of these two notions, and we show how to interpret the 1-KE protocols of Figures 5 and 6 in a very elegant way as "confirmed encryption of random K" and "confidential authentication of random K" respectively.

4 Advanced Security Properties from the Power of Interaction

We discuss three advanced security properties of message transmission protocols, each *requiring interaction*: replay security, forward security, and deniability. While the last two properties have been already considered in previous work in the context of encryption, key exchange, and message authentication, the first one, replay security, is new and aims at obtaining more intuitive security definitions of message transmission. Interestingly, we will show that replay security can be achieved only by interactive protocols, and that with enough interaction our notions of iCCA and iCMA security already provide replay-secure protocols. Finally, we will also define and construct stronger forms of necessarily interactive PKE/PKMA schemes, called *confirmed encryption* and *confidential authentication*.

4.1 Replay Security

Both our definitions of iCCA and iCMA security syntactically disallow man-in-the-middle (MiM) attacks against the challenge session. Even though this restriction allows us to obtain meaningful (i.e., non-trivial) security definitions, MiM adversaries who replay messages are, in fact, unavoidable. It is therefore left to the application to make sure that such attacks be infeasible. The question however is: how to do it in practice? Checking matching transcripts seems, in fact, hard and likely to be infeasible if one wants to take security measures before a sensitive session is completed.

In this section we propose a solution to this issue by introducing orthogonal, but arguably more intuitive notions that we call *replay security* for PKMA/PKE. For PKMA, the basic idea is that a honest verifier is assured that the "current" message is actually being authenticated by the secret key owner "now", as opposed to some time in the past. For PKE, a honest encryptor is similarly assured that the "current" message can only be decrypted by the secret key owner "now", as opposed to some time in the future. At a more technical level, this means that the only way for an adversary to break the security of message transmission protocols is to have access to the honest party (e.g., the legitimate decryptor or authenticator) *during* the challenge session. Therefore, as long as the honest party is careful enough during some particularly sensitive transmission, the security property of this communication is preserved, even if the party "loses its guard" *before* or *after* (but not during) the important transmission.

In what follows we formalize this notion of replay security, and then we show that traditional non-interactive protocols (e.g., CCA encryption and signatures) *cannot* be replay-secure. Intuitively, the reason is that a ciphertext or a digital signature can always be "replayed" after its transmission (this also explains our choice for the name of this notion).

Time Intervals and Concurrent Sessions. To formalize our definitions and argue more easily about concurrent sessions, we refine our notion of time and we introduce an intuitive terminology. If t and t' are time instants such that $t < t'$, then we denote with $[t, t']$ the *time interval* between t and t', i.e., the sequence $\langle t, t+1, t+2, \ldots, t' \rangle$. For every protocol transcript $T = \langle (t_1, M_1), \ldots, (t_n, M_n) \rangle$ (i.e., for every session) of an n-round protocol there exists a corresponding time interval $[t_1, t_n]$ in which the session starts and ends. Let $[t_1^*, t_n^*]$ and $[t_1, t_n]$ be the time intervals of two protocol sessions. We say that $[t_1^*, t_n^*]$ and $[t_1, t_n]$ *overlap* if $[t_1, t_n] \cap [t_1^*, t_n^*] \neq \emptyset$. Moreover, we say that $\mathcal{A}$ is an *overlapping* adversary if it generates an oracle session $[t_1, t_n]$ that overlaps with the challenge session $[t_1^*, t_n^*]$.

In the following lemma we show that in any protocol with at least two rounds, any ping-pong adversary is also overlapping. We will use this general statement to prove that any 2-round secure message-transmission protocol is also replay-secure.

Lemma 2. *Let $n \geq 2$ and Π be an n-round message transmission protocol. If $\mathcal{A}$ is a ping-pong adversary against Π, then $\mathcal{A}$ is overlapping.*

Proof. Assume by contradiction that $\mathcal{A}$ is not overlapping, then we show that $\mathcal{A}$ is not ping-pong. Let T^* and $[t_1^*, t_n^*]$ be the transcript and time interval of the challenge session. Since $\mathcal{A}$ is not overlapping, no oracle session T overlaps with T^*, i.e., for every oracle session with time interval $[t_1, t_n]$ it holds $[t_1^*, t_n^*] \cap [t_1, t_n] = \emptyset$. However, since $n \geq 2$, it is easy to see that if $[t_1^*, t_n^*] \cap [t_1, t_n] = \emptyset$ then the timestamps of these two sessions are certainly not alternating, and thus $T \not\equiv T^*$. $\square$

Replay-Secure Public-Key Message Authentication. Informally speaking, a PKMA protocol is replay-secure if all oracle sessions initialized with the challenge plaintext m^* do not overlap with the challenge session. This essentially means that the adversary loses access to the legitimate signer (on message m^*) *before* starting to forge m^*.

For a more formal definition, consider the experiment $\mathbf{Exp}_{\Pi,\mathcal{A}}^{\mathsf{iCMA}}$ (defined in Section 2.2) and let $[t_1^*, t_n^*]$ be the time interval of the challenge session, and $\{[t_1^i, t_n^i]\}_{i=1,\ldots,Q}$ be the time intervals of all the oracle sessions established by $\mathcal{A}$. Moreover, let m^i be the plaintext used to initialize the sender oracle $\mathsf{S}(\mathsf{sendk}, m^i)$ in the i-th oracle session. Then we call $\mathcal{A}$ a *replay adversary* if there exists $i \in \{1, \ldots, Q\}$ such that $m^i = m^*$ and $[t_1^i, t_n^i] \cap [t_1^*, t_n^*] \neq \emptyset$. Replay security for PKMA is defined as as follows:

Definition 9 (Replay Secure PKMA). *Let $\mathbf{Exp}_{\Pi,\mathcal{A}}^{\mathsf{RSMA}}$ be the same experiment as $\mathbf{Exp}_{\Pi,\mathcal{A}}^{\mathsf{iCMA}}$, except that $\mathcal{A}$ is required not to be a replay adversary (instead of not being ping-pong). Then we say that an interactive protocol Π is a replay secure PKMA (*RSMA *for short) if for any PPT $\mathcal{A}$, its advantage $\mathbf{Adv}_{\Pi,\mathcal{A}}^{\mathsf{RSMA}}(\lambda) = \left|\Pr[\mathbf{Exp}_{\Pi,\mathcal{A}}^{\mathsf{RSMA}}(\lambda) = 1] - \frac{1}{2}\right|$ is negligible.*

First, in the following theorem we show that 1-round PKMA protocols cannot be replay-secure. Its proof is rather simple and follows from the fact that in 1-round protocols there is clearly no overlap between different sessions. Hence, the dummy adversary who replays a signature received from the legitimate signer is a valid adversary in the RSMA experiment.

Theorem 6. *Let $\mathsf{\Pi}$ be any 1-round* iCMA*-secure PKMA. Then $\mathsf{\Pi}$ is not* RSMA*-secure.*

It is worth noting that one can obtain 1-round replay-secure solutions in different models, e.g., by introducing global time periods and requiring the signer to always sign the message with the current timestamp. However, such a solution falls outside the pure non-interactive model considered by our work.

Therefore, while 1-round replay-secure PKMA cannot be achieved, in the following theorem we show that with at least *two* rounds of interaction any iCMA-secure protocol is a replay-secure PKMA.

Theorem 7. *For $n \geq 2$, any n-round* iCMA*-secure protocol $\mathsf{\Pi}$ is* RSMA*-secure.*

Proof. The proof follows by observing that if $\mathcal{A}$ is not a replay adversary in $\mathbf{Exp}^{\mathsf{RSMA}}_{\mathsf{\Pi},\mathcal{A}}$ (for $\mathsf{\Pi}$ with at least 2 rounds), then $\mathcal{A}$ is also not ping-pong in $\mathbf{Exp}^{\mathsf{iCMA}}_{\mathsf{\Pi},\mathcal{A}}$. Namely, we show that for a non-replay $\mathcal{A}$ we have that for all $i = 1, \ldots, Q$, $T_i \not\equiv T^*$. For every i, there are only two possible cases: $m^i = m^*$ or $m^i \neq m^*$. If $m^i = m^*$, then $T_i \not\equiv T^*$ follows by Lemma 2. In the case of sessions i where $m^i \neq m^*$, assume by contradiction that $T_i \equiv T^*$. Then by definition of match the sessions must share the same protocol messages in the same order (i.e., $M^i_j = M^*_j$) which, by correctness, implies that $m^i = m^*$, which is a contradiction. □

From Theorem 7 we obtain two interesting results that we summarize in the following corollary:

Corollary 4. *(1) There exists a simple 2-round replay-secure PKMA protocol based on any strongly unforgeable signature scheme (and thus on one-way functions); (2) there exists a simple 2-round replay-secure PKMA protocol based on any* IND−CCA*-secure PKE.*

The construction (1) follows by combining Theorem 7 with our round-extension result (Theorem 3) applied to any strongly unforgeable signature (aka 1-round iCMA-secure PKMA). The construction (2) from IND−CCA-secure PKE is instead obtained by applying Theorem 7 to our 2-round construction of Theorem 2 (instantiated with a non-interactive IND−CCA-secure PKE scheme).

Remark 2 (Relation to Concurrent-Secure Identification Schemes). Notice that in the special case when the message space has cardinality 1, our notion of replay-secure PKMA essentially corresponds to the strongest security notion for identification schemes, called impersonation security under concurrent attacks [6]. In this case (i.e., message space of cardinality 1), a PKMA can be indeed seen as an identification scheme. Moreover, by considering authentication with an empty message space, the 2-round PKMA protocols mentioned in Corollary 4 recover well known 2-round, signature-based and encryption-based, identification schemes (see, e.g., [3], and notice that outputting the secret key is indeed a secure MAC for an empty message space).

Replay-Secure Public-Key Encryption. Informally speaking, a PKE protocol is *replay-secure* if there is no overlap between the challenge session and all oracle sessions in which the plaintext revealed by the receiver is one of the two challenge plaintexts. In essence, this means that *during* the challenge session the adversary loses access to the legitimate decryptor , but only on the challenge plaintexts m_0 or m_1. More formally, consider the experiment $\mathbf{Exp}^{\mathsf{iCCA}}_{\mathsf{\Pi},\mathcal{A}}$ (defined in Section 2.2) and let $[t^*_1, t^*_n]$ be the time interval of the challenge session, and $\{[t^i_1, t^i_n]\}_{i=1,\ldots,Q}$ be the time intervals of all the oracle sessions established by $\mathcal{A}$. Moreover, let m^i be the plaintext revealed by the receiver in the i-th oracle session (wlog we only consider completed sessions). Then, we call $\mathcal{A}$ a *replay adversary* if there exists $i \in \{1, \ldots, Q\}$ such that $m^i = m_0$ or $m^i = m_1$, and $[t^i_1, t^i_n]$ overlaps with $[t^*_1, t^*_n]$. Replay security for PKE is defined as follows:

Definition 10 (Replay-Secure Encryption). *Let $\mathbf{Exp}^{\mathsf{RSE}}_{\mathsf{\Pi},\mathcal{A}}$ be the same as experiment $\mathbf{Exp}^{\mathsf{iCCA}}_{\mathsf{\Pi},\mathcal{A}}$, except that $\mathcal{A}$ is required not to be a replay adversary (instead of denying it to be ping-pong). Then we say that an interactive protocol $\mathsf{\Pi}$ is a* replay-secure encryption *(*RSE*) if for any PPT $\mathcal{A}$, its advantage $\mathbf{Adv}^{\mathsf{RSE}}_{\mathsf{\Pi},\mathcal{A}}(\lambda) = \left|\Pr[\mathbf{Exp}^{\mathsf{RSE}}_{\mathsf{\Pi},\mathcal{A}}(\lambda) = 1] - \frac{1}{2}\right|$ is negligible.*

It is worth mentioning that the notion of replay-secure PKE is similar to the notion of Replayable CCA-secure encryption (RCCA) introduced by Canetti, Krawczyk and Nielsen [12]. In RCCA security the adversary is allowed to submit any ciphertext c to the decryption oracle, except that if c decrypts to m_0 or m_1 the adversary gets a special string `test` as response.

Similarly to replay-secure PKMA, in the following theorems we show that replay-secure PKE requires at least 2-rounds of interaction to be achieved.

Theorem 8. *Let Π be any 1-round* iCCA*-secure PKE. Then Π is not* RSE*-secure.*

The proof is obtained by considering the adversary who replays the challenge ciphertext to the receiver.

Theorem 9. *For $n \geq 2$, any n-round* iCCA*-secure protocol Π is* RSE*-secure.*

Proof. The proof follows by observing that if $\mathcal{A}$ is not a replay adversary in $\mathbf{Exp}^{\mathsf{RSE}}_{\Pi,\mathcal{A}}$ where Π has at least 2 rounds, then $\mathcal{A}$ is also not ping-pong in $\mathbf{Exp}^{\mathsf{iCCA}}_{\Pi,\mathcal{A}}$. Namely, we show that for a non-replay adversary $\mathcal{A}$ we have that for all $i = 1, \dots, Q$, it holds $T_i \not\equiv T^*$. For every i, there are only two possible cases: (1) $m^i = m_0$ or $m^i = m_1$, and (2) $m^i \neq m_0, m_1$. In the first case, note that $T_i \not\equiv T^*$ follows by Lemma 2, i.e., if the $\mathcal{A}$ is not replay, then $\mathcal{A}$ is not ping-pong w.r.t. these sessions. In the case of sessions i where $m^i \neq m_0, m_1$, assume by contradiction that $T_i \equiv T^*$. Then by definition of match, these sessions must share the same protocol messages in the very same order (i.e., $M^i_j = M^*_j$). However, by correctness, this implies that $m^i = m_b$ where b is the secret bit chosen in the experiment. But this is a contradiction as we assumed $m^i \neq m_0, m_1$. □

From Theorem 9 we obtain a collection of nice results that we summarize in the following Corollary:

Corollary 5. *(1)* IND−CPA*-secure PKE is sufficient to build* RSE*-secure PKE; (2) there exists an efficient 2-round* RSE*-secure PKE protocol based on 1-bounded-*IND−CCA*-secure PKE and signature schemes (see Theorem 1); (3) there exists an efficient 2-round* RSE*-secure PKE protocol based on any* IND−CCA*-secure PKE (via the round-extension transformation of Theorem 3).*

Strong Replay-Secure Encryption. We notice that our definition of replay security (for both PKMA and PKE) does not allow the adversary to suspend "critical" sessions (e.g., sessions authenticating m^*, or sessions which decrypt to m_0 or m_1) during the challenge session, and to later resume these sessions once the challenge session is over. While allowing such suspended sessions would not make sense for PKMA (indeed observe that the outcome of the security experiment is determined upon the end of the challenge session), it might be a reasonable strengthening for replay-secure PKE. Here we define strong-replay-secure PKE, and we show that two rounds of interaction are insufficient to achieve strong replay-security with suspended sessions (see Theorem 10 below), but *three* rounds are enough and indeed any (at least) 3-round iCCA-secure PKE is strong replay-secure (cf. Theorem 11).

Towards defining strong replay security more formally, let us say that $[t_1, t_n]$ is *off* during $[t^*_1, t^*_n]$ if for all $j = 1, \dots, n$ we have that $t_j \notin [t^*_1, t^*_n]$. We call $\mathcal{A}$ a *strong replay adversary* if there exists $i \in \{1, \dots, Q\}$ such that $m^i = m_0$ or $m^i = m_1$, and $[t^i_1, t^i_n]$ is *not off* during $[t^*_1, t^*_n]$.

Definition 11 (Strong Replay-Secure Encryption). *Let $\mathbf{Exp}^{\mathsf{sRSE}}_{\Pi,\mathcal{A}}$ be the same experiment as $\mathbf{Exp}^{\mathsf{iCCA}}_{\Pi,\mathcal{A}}$, except that $\mathcal{A}$ is required not to be a strong replay adversary (instead of denying it to be ping-pong). Then we say that an interactive protocol Π is a* strong replay secure encryption *(*sRSE*) if for any PPT $\mathcal{A}$, its advantage $\mathbf{Adv}^{\mathsf{sRSE}}_{\Pi,\mathcal{A}}(\lambda) = \left|\Pr[\mathbf{Exp}^{\mathsf{sRSE}}_{\Pi,\mathcal{A}}(\lambda) = 1] - \frac{1}{2}\right|$ is negligible.*

Theorem 10. *Let Π be any 2-round* iCCA*-secure PKE. Then Π is not* sRSE*-secure.*

Proof. To prove the theorem we show that there exists an adversary $\mathcal{A}$ who is not a strong replay adversary in $\mathbf{Exp}^{\mathsf{sRSE}}_{\Pi,\mathcal{A}}$ but is a ping-pong adversary. Then, being ping-pong, $\mathcal{A}$ can trivially obtain a decryption of the challenge plaintext.

First, observe that in any 2-round protocol we have the first message from R to S. So, consider the following adversary $\mathcal{A}$ that plays the role of the receiver in the challenge session:

- $\mathcal{A}$ queries R on a new session and obtains M_1 with timestamp $t_1 = t$.
- $\mathcal{A}$ sends M_1 to the honest sender S as the first protocol message in the challenge session (here M_1 gets timestamp $t_1^* = t + 1$). $\mathcal{A}$ receives back M_2 from S, where M_2 gets timestamp $t_2^* = t + 2$.
- $\mathcal{A}$ forwards M_2 to R in the session previously opened in step 1 (this message gets timestamp $t_2 = t + 3$).

While $\mathcal{A}$ is clearly ping-pong according to Definition 4, $\mathcal{A}$ is still not a strong replay adversary in $\mathbf{Exp}^{\mathsf{sRSE}}_{\Pi,\mathcal{A}}$ since $t_1, t_2 \notin [t+1, t+2]$. This is exactly a case in which the adversary suspended a session. $\square$

Theorem 11. *For $n \geq 3$, any n-round* iCCA*-secure protocol* Π *is* sRSE*-secure.*

The proof is basically the same as that of Theorem 9, except that here we use the following lemma to show that for 3-round protocols any ping-pong adversary generates an oracle session (decrypting to one of the challenge plaintexts) which is not off during the challenge session.

Lemma 3. *Let $n \geq 3$, and let T, $[t_1, t_n]$ and T^*, $[t_1^*, t_n^*]$ be the transcripts and time intervals of two sessions of an n-round protocol. If $[t_1, t_n]$ is off during $[t_1^*, t_n^*]$, then T does not match with T^*.*

Proof. Recall that $[t_1, t_n]$ is off during $[t_1^*, t_n^*]$ if for all $j = 1, \ldots, n$ we have that $t_j \notin [t_1^*, t_n^*]$. This means that either one of the following cases occurs: (1) $[t_1^*, t_n^*] \cap [t_1, t_n] = \emptyset$, (2) $[t_1^*, t_n^*] \cap [t_1, t_n] \neq \emptyset$. Case (1) is identical to that of Lemma 2. In case (2), we have that the two sessions overlap, and we also know that $t_j \notin [t_1^*, t_n^*], \forall j = 1, \ldots, n$, that is $t_1 < t_1^*$ and $t_3 > t_3^*$. Since there are at least 3 rounds, there exists at least a distinct timestamp t_2 such that $t_1 < t_2 < t_n$ and such that t_2 satisfies either one of the following conditions: $t_1 < t_2 < t_1^*$, or $t_n > t_2 > t_n^*$. However, one can again check that in neither one of these cases the timestamps are correctly alternating. Therefore, $T \not\equiv T^*$. $\square$

4.2 Forward Security

Intuitively, forward security guarantees that any leak of secret information at some time t should not affect the security of protocol runs that occurred in the past, i.e., at any time $t' < t$. Forward security is a desirable property that is not known to be achieved by standard non-interactive public key encryption. Here we formalize suitable definitions of forward security for interactive encryption protocols, and we show that our constructions satisfy this property.

We consider both a weak and a strong version of forward security. To define *weak forward security*, we introduce an oracle $\mathsf{Corrupt}$, which outputs the secret key recvk of the receiver R. Then we define the experiment $\mathbf{Exp}^{\mathsf{wFS}}_{\Pi,\mathcal{A}}(\lambda)$ to be the same as $\mathbf{Exp}^{\mathsf{iCCA}}_{\Pi,\mathcal{A}}(\lambda)$ except that $\mathcal{A}$ is additionally given access to the oracle $\mathsf{Corrupt}$ that can be queried only *after* the challenge session is completed.

Definition 12 (Weak Forward Security). *We define the advantage of an adversary $\mathcal{A}$ in breaking weak forward security* (wFS) *of protocol* Π *as* $\mathbf{Adv}^{\mathsf{wFS}}_{\Pi,\mathcal{A}}(\lambda) = \left|\Pr[\mathbf{Exp}^{\mathsf{wFS}}_{\Pi,\mathcal{A}}(\lambda) = 1] - \frac{1}{2}\right|$*, and we say that* Π *is weak forward secure* (wFS) *if for any PPT $\mathcal{A}$,* $\mathbf{Adv}^{\mathsf{wFS}}_{\mathsf{iPKE},\mathcal{A}}(\lambda)$ *is negligible.*

To define *strong forward security* we introduce another oracle, $\mathsf{StateCorrupt}$, which outputs the receiver's secret key recvk as well as R's private state, consisting of random coins and private information generated during the run of the protocol. In order to make the game non-trivial for the adversary, we restrict the revealed state only to sessions that are at the time of the query not completed, and whose transcript T does not *partially match* with the transcript T^* of the challenge session. We say that a transcript T *partially matches* with T^* if T matches T^* in the old sense up to the first n' messages, where n' is the length of T, i.e., if T^* has length n and T has length $n' \leq n$, we consider the first portion $\tilde{T}^*$ of length n' of the transcript T^*, and we apply the previous definition of match, i.e., $T \equiv \tilde{T}^*$. This notion of partial match is introduced to formalize the fact that the adversary should obtain not even a small portion of the private state concerning the challenge session. Although this may appear a restriction, we make the following two observations. First, it follows our intuition for the security of interactive encryption in which sensitive transmissions should be particularly secured so as not to leak private state information. Second, it is reasonable to think that once the decryptor loses its control and reveals *all* its private state to the adversary (even the state of uncompleted sessions), then there should be no specific security guarantees for every session which is significantly related with the revealed private state.

Formally, we define the experiment $\mathbf{Exp}^{\mathsf{sFS}}_{\Pi,\mathcal{A}}(\lambda)$ to be the same as $\mathbf{Exp}^{\mathsf{iCCA}}_{\Pi,\mathcal{A}}(\lambda)$ except that $\mathcal{A}$ is additionally given access to the oracle $\mathsf{StateCorrupt}$, that can be queried only *after* the challenge session is completed.

Definition 13 (Strong Forward Security). *We define the advantage of $\mathcal{A}$ in breaking the strong forward security* (sFS) *of Π as $\mathbf{Adv}^{\mathsf{sFS}}_{\Pi,\mathcal{A}}(\lambda) = \left|\Pr[\mathbf{Exp}^{\mathsf{sFS}}_{\Pi,\mathcal{A}}(\lambda) = 1] - \frac{1}{2}\right|$, and we say that Π is strong forward secure* (sFS) *if for any PPT $\mathcal{A}$, $\mathbf{Adv}^{\mathsf{sFS}}_{\mathsf{iPKE},\mathcal{A}}(\lambda)$ is negligible.*

Building Forward Secure (Interactive) PKE. Below we show that our construction of interactive encryption from $\mathsf{IND-CPA}$-secure encryption proposed in Section 3.1 achieves strong forward security.

Intuitively, this follows from observing that the long-term private key of R is the signing key sk whereas its private state consists of one-time decryption keys dk. In more detail, we show that our proof of iCCA security (Theorem 1) can be adapted to the case of strong forward security as follows. First, we can define the same event Forge, and observe that the proof of Claim 1 to bound $\Pr[\mathsf{Forge}]$ remains the same. Indeed, in this proof $\mathsf{StateCorrupt}$ queries do not have to be simulated as the entire simulation will stop *before* the challenge session is completed. Second, the remaining part of the proof of Theorem 1 can be easily changed as follows to enable the simulator $\mathcal{B}$ answer $\mathsf{StateCorrupt}$ queries. $\mathcal{B}$ knows the secret key $\mathsf{recvk} = \mathsf{sendk}'$ (it is the secret key generated by $\mathcal{B}$ itself), which can thus be returned in output. Then, by definition of strong forward security the only sessions whose state is to be revealed are those sessions that do not partially match with the challenge session: essentially those sessions for which $\mathcal{B}$ already knows the one-time decryption key dk.

Remark 3. While we showed that our construction of Section 3.1 satisfies strong forward security, we observe that a construction obtained by applying our round-extension Theorem 3 instead cannot be proven strong forward secure because of the following attack (here we consider the case when S speaks first).

- The adversary starts the challenge session with to get the first message r^* from the challenger.
- Next, it chooses some $r \neq r^*$ and queries R by sending r to start a new session.
- It receives back ek from R (at the end of Π'), and forwards ek to the challenger as the second message.
- Let c^* be the message received by the challenger. Now, $\mathcal{A}$ queries $\mathsf{StateCorrupt}$ which will return the secret key dk corresponding to ek. Notice that such a query is legal as this session does not partially match with the challenge one as $r \neq r^*$.
- Finally, $\mathcal{A}$ uses dk to decrypt the ciphertext c^*.

The above issue stems from the fact that our round extension transformation does not preserve forward secrecy. However, nothing is lost as the issue can be fixed, thus achieving strong forward security even for iCCA protocols obtained via round-extension. To do this, we can modify our generic transformation as follows: when R receives the random r from S, R signs *every* protocol message together with r, and includes such signatures along with the messages. Clearly, the above attack no longer applies as $\mathcal{A}$ will not be able to create valid protocol messages for an $r \neq r^*$. A formal proof easily follows the intuition above, and is omitted.

4.3 Deniability

Informally speaking, a PKMA protocol is deniable if the authenticator S can authenticate a message m to the receiver R in such a way that R *cannot* use the transcript of their conversation as evidence to later convince third parties about the fact that S took part in the protocol and authenticated m.

The area of deniable authentication has attracted a lot of attention [23,22,34,17] and has several variants depending on the exact attack scenario. While a detailed exploration of this area is beyond the scope of our work, to illustrate the potential of interactivity we show that our PKMA protocols based on iCCA encryption already satisfies the weakest form of (perfect) passive deniability, which we define below.

Definition 14 (Passive Deniability). *A PKMA protocol* $\Pi = (\mathsf{Setup}, \mathsf{S}, \mathsf{R})$ *is* passive deniable *if there exists a PPT simulator* Sim *such that for all honestly generated keys* $(\mathsf{recvk}, \mathsf{sendk}) \xleftarrow{\$} \mathsf{Setup}(1^\lambda)$*, and for any message* $m \in \mathcal{M}$*,* $\mathsf{Sim}(\mathsf{recvk}, m)$ *generates a transcript* π_S *that is (computationally, statistically, or perfectly) indistinguishable from a transcript* π *of a real execution* $\langle \mathsf{S}(\mathsf{sendk}, m), \mathsf{R}(\mathsf{recvk})\rangle$ *of the protocol.*

Forward Deniability. As noted by Di Raimondo and Gennaro [17], the standard definition of deniability provides guarantees only to the sender. However, if the sender later changes its mind, it might be able to exhibit some witness (e.g., its private key) that proves its participation in the protocol. To rule out even this possibility, Di Raimondo and Gennaro introduced the notion of *forward deniability*, and showed that every protocol which is deniable in a statistical or perfect sense is also forward deniable.

Remark 4 (Strong Deniability). It is possible to also consider a stronger form of deniability [23], in which the receiver might be dishonest. The resulting definition is essentially very similar to the one for dishonest verifier zero-knowledge. We refer to [23] for more details.

Building Forward Deniable PKMA. Let Π_{mac} and Π_{lab} be our PKMA protocols in Section 3.2 and Appendix E based on iCCA encryption and MACs and on labeled iCCA encryption, respectively, when instantiated with non-interactive, i.e., 1-round, (labeled) iCCA protocols. Then we can state the following theorem:

Theorem 12. *The protocols* Π_{mac} *and* Π_{lab} *are passive forward deniable.*

For protocol Π_{lab}, the proof simply follows by observing that the following PPT simulator Sim satisfies Definition 14. $\mathsf{Sim}(\mathsf{recvk}, m)$ chooses a random $r \xleftarrow{\$} \tilde{\mathcal{M}}$ (where $\tilde{\mathcal{M}}$ is the message space of the labeled iCCA encryption protocol), and outputs $\pi_S = \langle m, \mathsf{S}_m(\mathsf{recvk}, r), r\rangle$. It is easy to observe that π_S is exactly distributed as the transcript of a real execution of the protocol. For the protocol Π_{mac} the proof is essentially the same except that the simulated transcript is $\pi_S = \langle \mathsf{S}(\mathsf{recvk}, r), \mathsf{Tag}_r(m)\rangle$. Finally, note that since the transcripts are perfectly indistinguishable the protocols are also forward deniable.

As already noticed in previous work [23,18], both protocols Π_{mac} and Π_{lab} can be modified by adding a challenge-response subprotocol in order to make them strong deniable for sequential executions.

4.4 Forward Security and Deniability for Anonymous Key-Exchange

We consider the advanced properties of *forward security* and *deniability* also for anonymous key-exchange.

In a nutshell, forward security guarantees that once a session is completed, the session key remains secure even if the adversary learns the long-term secret keys (in the case of 1-KE, only the authenticated party T has a long-term secret key). The formal definition of forward security is very similar to the one given in Section 4.2 for iCCA encryption protocols. For weak forward security, we run $\mathcal{A}$ in experiment $\mathbf{Exp}_{\Pi,\mathcal{A}}^{\mathsf{1-KE-Sec}}(\lambda)$ with the additional $\mathsf{Corrupt}$ oracle, which can be queried only after the challenge session is completed. For strong forward security, we run $\mathcal{A}$ in experiment $\mathbf{Exp}_{\Pi,\mathcal{A}}^{\mathsf{1-KE-Sec}}(\lambda)$ with the oracle $\mathsf{StateCorrupt}$ which can be queried only after the challenge session is over. $\mathsf{StateCorrupt}$ is defined as in the iCCA case, i.e., it reveals the state of all the opened sessions that do not partially match with the challenge session.

We consider deniability with respect to the keyed party T. Informally, this property says that the unkeyed party U cannot use the transcript of their conversation to convince third parties that T took part in the protocol. The definition is essentially the same as that for PKMA protocols, and we can have both a weak (aka passive) and a strong version of deniability.

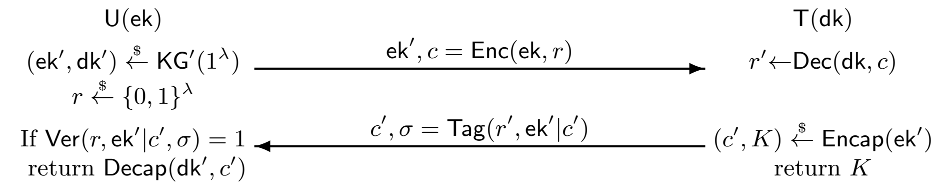
Realizations of Forward-Secure/Deniable 1-KE. Here we analyze the four instantiations of 1-KE protocols mentioned in Section 3.4, with a special focus on the properties of forward security vs. deniability:

1. Protocol of Figure 5 where the iCCA protocol Π' is a non-interactive $\mathsf{IND-CCA}$ scheme: we obtain a *2-round* 1-KE based on $\mathsf{IND-CCA}$ that *is (forward) passive deniable* (a perfectly indistinguishable transcript for a honest U is easily simulatable), but it is *not forward secure* (recovering the long-term key recvk' trivially allows to recover r).
2. Protocol of Figure 5 where the iCCA protocol Π' is our 2-round protocol of Section 3.1 based on $\mathsf{IND-CPA}$ security: we obtain a *3-round* 1-KE based on $\mathsf{IND-CPA}$ security that is *not deniable* (as T signs the first message with a digital signature) but it *is forward secure* (since so is the 2-round iCCA protocol).
3. Protocol of Figure 6 where the iCMA protocol Π' is a digital signature: we obtain a *2-round* 1-KE based on $\mathsf{IND-CPA}$ security that is clearly *not deniable* (as T signs c) but it can be shown *forward-secure* (as dk' is a short-term key which is deleted once the session is over). It is worth noting that when implementing the KEM with standard DH key-exchange ($\mathsf{ek}' = g^x, c = g^y, K = g^{xy}$) we essentially recover protocol A-DHKE-1 in [48]. A very similar protocol based on $\mathsf{IND-CPA}$ KEM is also recovered in the recent, independent work of Maurer et al. [38].
4. Protocol of Figure 6 where the iCMA protocol Π' is the 2-round PKMA based on $\mathsf{IND-CCA}$ and MACs that we call Π_{mac}: we obtain a *2-round* 1-KE (as we can piggy-back the first round of Π_{mac} on the first round of the 1-KE). Somewhat interestingly, this instantiation achieves *the best possible properties for a 2-round protocol*: it enjoys *both* passive forward deniability (as Π_{mac} is passive forward-deniable) and forward security (since dk' is short-term, as in the previous case). The resulting protocol is depicted in Figure 7, and we notice that this essentially recovers the unilateral version of SKEME [35]. Moreover, by using the MAC of [19] and by applying similar optimizations as in Figure 3, we obtain a 1-KE protocol based only on CCA security (depicted in Figure 8).

4.5 Confirmed Encryption and Confidential Authentication

In this section we introduce two advanced notions of (interactive) PKE and PKMA that we call *confirmed encryption* and *confidential authentication* (ConfPKE and ConfPKMA, for short). The basic idea is to extend encryption in such a way that the sender receives confirmation that the receiver obtained the transmitted message, and to extend authentication so that the transmitted messages remain private. At a high level, these two notions have similarities as they both aim to capture at the same time confidentiality and some notion of integrity. The main difference is which of the two parties obtains

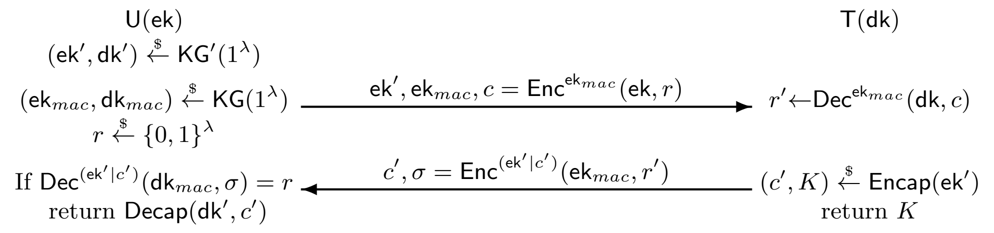
Setting: a key pair $(\mathsf{ek},\mathsf{dk})$ for an $\mathsf{IND-CCA}$-secure PKE $\mathcal{E} = (\mathsf{KG},\mathsf{Enc},\mathsf{Dec})$ is generated. $\mathcal{E}' = (\mathsf{KG}',\mathsf{Encap},\mathsf{Decap})$ is an $\mathsf{IND-CPA}$-secure KEM.



$\mathsf{U}(\mathsf{ek})$
$\mathsf{T}(\mathsf{dk})$
$(\mathsf{ek}',\mathsf{dk}') \xleftarrow{\$} \mathsf{KG}'(1^\lambda)$
$r \xleftarrow{\$} \{0,1\}^\lambda$
$\mathsf{ek}', c = \mathsf{Enc}(\mathsf{ek}, r)$
$r' \leftarrow \mathsf{Dec}(\mathsf{dk}, c)$
$c', \sigma = \mathsf{Tag}(r', \mathsf{ek}'|c')$
$(c', K) \xleftarrow{\$} \mathsf{Encap}(\mathsf{ek}')$
return K
If $\mathsf{Ver}(r, \mathsf{ek}'|c', \sigma) = 1$
return $\mathsf{Decap}(\mathsf{dk}', c')$

Fig. 7. A 2-round forward-deniable and forward-secure 1-KE.

Setting: a key pair $(\mathsf{ek},\mathsf{dk})$ for a labeled $\mathsf{IND-CCA}$-secure PKE $\mathcal{E} = (\mathsf{KG},\mathsf{Enc},\mathsf{Dec})$ is generated. $\mathcal{E}' = (\mathsf{KG}',\mathsf{Encap},\mathsf{Decap})$ is an $\mathsf{IND-CPA}$-secure KEM.



$\mathsf{U}(\mathsf{ek})$
$\mathsf{T}(\mathsf{dk})$
$(\mathsf{ek}',\mathsf{dk}') \xleftarrow{\$} \mathsf{KG}'(1^\lambda)$
$(\mathsf{ek}_{mac},\mathsf{dk}_{mac}) \xleftarrow{\$} \mathsf{KG}(1^\lambda)$
$r \xleftarrow{\$} \{0,1\}^\lambda$
$\mathsf{ek}', \mathsf{ek}_{mac}, c = \mathsf{Enc}^{\mathsf{ek}_{mac}}(\mathsf{ek}, r)$
$r' \leftarrow \mathsf{Dec}^{\mathsf{ek}_{mac}}(\mathsf{dk}, c)$
$c', \sigma = \mathsf{Enc}^{(\mathsf{ek}'|c')}(\mathsf{ek}_{mac}, r')$
$(c', K) \xleftarrow{\$} \mathsf{Encap}(\mathsf{ek}')$
return K
If $\mathsf{Dec}^{(\mathsf{ek}'|c')}(\mathsf{dk}_{mac}, \sigma) = r$
return $\mathsf{Decap}(\mathsf{dk}', c')$

Fig. 8. A 2-round forward-deniable and forward-secure 1-KE based on CCA encryption.

such integrity guarantee. This is essentially due to the fact that in the two notions the role of the keyed parties (i.e., who has a public/private key) is swapped.

Confirmed Encryption. To define ConfPKE, consider a message transmission protocol Π defined as in Section 2.1, with the only change that the sender also returns a local output – a plaintext $m \in \mathcal{M}$ or an error $\bot$ – according to whether it receives evidence that the receiver obtained the transmitted plaintext m. As in 1-KE, observe that such a change implies that wlog the receiver always speaks last. Correctness of ConfPKE is thus obtained by extending the one of message transmission protocols so that both sender and receiver output the same message, i.e., $\langle \mathsf{S}(\mathsf{sendk}, m), \mathsf{R}(\mathsf{recvk})\rangle = (m, m)$ holds for all honestly generated keys and all plaintexts $m \in \mathcal{M}$.

For security, we want essentially two properties: confidentiality (no information about the transmitted plaintexts is leaked) and confirmation (the sender is correctly assured that the receiver obtained the transmitted plaintext). To formalize this notion we define experiment $\mathbf{Exp}^{\mathsf{ConfEnc}}_{\Pi,\mathcal{A}}(\lambda)$ in Figure 9. Briefly, it works as follows: given oracle access to the keyed party $\mathsf{R}(\mathsf{recvk})$, $\mathcal{A}$ first chooses two plaintexts m_0, m_1, and then runs a challenge session with $\mathsf{S}(\mathsf{sendk}, m_b)$. Since here R speaks last, as in 1-KE we extend the ping-pong definition, and we say that $\mathcal{A}$ is "full-ping-pong" if $\mathcal{A}$ is ping-pong and, in the ping-pong session, $\mathcal{A}$ makes a last query to R that returns m'. $\mathcal{A}$ wins the game in two cases: (line 4) it breaks confirmation by letting S accept for some plaintext and without trivially forwarding messages, or (line 5) it breaks confidentiality by correctly guessing b and without being full-ping-pong. Therefore, we say that Π is a secure ConfPKE scheme if for every PPT $\mathcal{A}$, its advantage $\mathbf{Adv}^{\mathsf{ConfEnc}}_{\Pi,\mathcal{A}}(\lambda) = \left|\Pr[\mathbf{Exp}^{\mathsf{ConfEnc}}_{\Pi,\mathcal{A}}(\lambda) = 1] - \frac{1}{2}\right|$ is negligible.

Confidential Authentication. For confidential authentication, we consider a standard message transmission protocol Π (without any syntactic change), and we say that Π is a secure ConfPKMA if for any PPT $\mathcal{A}$ its advantage $\mathbf{Adv}^{\mathsf{ConfAuth}}_{\Pi,\mathcal{A}}(\lambda) = \left|\Pr[\mathbf{Exp}^{\mathsf{ConfAuth}}_{\Pi,\mathcal{A}}(\lambda) = 1] - \frac{1}{2}\right|$ is negligible. The experiment $\mathbf{Exp}^{\mathsf{ConfAuth}}_{\Pi,\mathcal{A}}(\lambda)$ is described in Figure 9 and is similar in the spirit to the one of ConfPKE, except that the keyed parties are swapped. So, given oracle access to $\mathsf{S}(\mathsf{sendk}, \cdot)$, $\mathcal{A}$ first chooses two plaintexts m_0, m_1 and then runs a challenge session with the receiver. In this session, however, $\mathcal{A}$ is also given oracle access to a single specific sender's copy $\mathsf{S}_1(\mathsf{sendk}, m_b)$ transmitting m_b. $\mathcal{A}$ wins the game in two cases:

```
Experiment $\mathbf{Exp}^{\mathsf{ConfEnc}}_{\Pi,\mathcal{A}}(\lambda)$
1. $b \xleftarrow{\$} \{0,1\}$; $(\mathsf{sendk},\mathsf{recvk}) \xleftarrow{\$} \mathsf{Setup}(1^\lambda)$
2. $(m_0, m_1) \leftarrow \mathcal{A}^{\mathsf{R}(\mathsf{recvk})}(\mathsf{sendk})$
3. $(m', b') \leftarrow \langle \mathsf{S}(\mathsf{sendk}, m_b), \mathcal{A}^{\mathsf{R}(\mathsf{recvk})}(\mathsf{sendk}) \rangle$
4. If $m' \neq \perp$ and $\mathcal{A}$ is not "ping-pong",
   then output 1
5. Else if $b' = b$ and $\mathcal{A}$ is not "full-ping-pong",
   then output 1
6. Else output 0.
```

```
Experiment $\mathbf{Exp}^{\mathsf{ConfAuth}}_{\Pi,\mathcal{A}}(\lambda)$
1. $b \xleftarrow{\$} \{0,1\}$; $(\mathsf{sendk},\mathsf{recvk}) \xleftarrow{\$} \mathsf{Setup}(1^\lambda)$
2. $(m_0, m_1) \leftarrow \mathcal{A}^{\mathsf{S}(\mathsf{sendk},\cdot)}(\mathsf{recvk})$
3. $(b', m') \leftarrow \langle \mathcal{A}^{\mathsf{S}_1(\mathsf{sendk}, m_b), \mathsf{S}(\mathsf{sendk},\cdot)}(\mathsf{recvk}), \mathsf{R}(\mathsf{recvk}) \rangle$
4. If $m' \neq \perp$ and $\mathcal{A}$ is not "ping-pong",
   then output 1
5. Else if $b' = b$ and $\mathcal{A}$ is not "ping-pong" w.r.t. $\mathsf{S}(\mathsf{sendk}, m_b)$,
   then output 1
6. Else output 0.
```

Fig. 9. Security experiments of ConfPKE and ConfPKMA.

(line 4) it breaks confirmation by letting S accept for some plaintext and without trivially forwarding messages, or (line 5) it breaks confidentiality by correctly guessing b and without being ping-pong (but notice that in this case we only care about ping-pong w.r.t. to the session transmitting m_b). We remark that although our $\mathsf{ConfAuth}$ security definition considers a single challenge (aka "left-or-right") oracle $\mathsf{S}_1(\mathsf{sendk}, m_b)$, it can be extended to the multi-challenge setting via a standard hybrid argument since here the adversary has also access to (multiple instances of) the sender oracle $\mathsf{S}(\mathsf{sendk}, \cdot)$.

Application to Anonymous Key-Exchange. We use the notions of ConfPKE and ConfPKMA to obtain a further smooth and clean transition from $\mathsf{iCCA}/\mathsf{iCMA}$ security to anonymous key-exchange. In the following lemmas, we show that by doing either "confirmed encryption of random K" or "confidential authentication of random K" we obtain secure 1-KE protocols, that are essentially a re-interpretation in a very elegant way of the two 1-KE protocols in Figures 5 and 6.

Lemma 4. *Let Π be a message transmission protocol, and let Π_1 be the 1-KE protocol in which U chooses a random K and sends K to T by running $\mathsf{S}(\mathsf{sendk}, K)$ (and, of course, T runs $\mathsf{R}(\mathsf{recvk})$). If Π is a secure ConfPKE, then Π_1 is a secure 1-KE.*

Lemma 5. *Let Π be a message transmission protocol, and let Π_2 be the 1-KE protocol in which: T chooses random K_1, K_2 and sends (K_1, K_2) to U by running $\mathsf{S}(\mathsf{sendk}, K_1|K_2)$; U (running $\mathsf{R}(\mathsf{recvk})$) gets K_1, K_2 and sends K_2 back to T. If Π is a secure ConfPKMA, then Π_2 is a secure 1-KE.*

We provide a proof sketch for Lemma 4. The proof of Lemma 5 is very similar.

Assume that $\mathcal{A}$ can break the security of the 1-KE protocol Π_1, we build an adversary $\mathcal{B}$ which breaks the security of the $ConfPKE$ scheme. Essentially, $\mathcal{B}$ runs $\mathcal{A}$ by forwarding all $\mathcal{A}$'s queries to its oracles. In particular, $\mathcal{B}$ simulates the challenge session by choosing two random keys K_0, K_1 so that $\mathcal{B}$'s challenger will run the challenge session with one of these two keys. Once $\mathcal{A}$ sends the last protocol message in the challenge session, $\mathcal{B}$ checks if $\mathcal{A}$ (and thus also $\mathcal{B}$ itself) was ping-pong and proceeds as follows: (i) if $\mathcal{A}$ was not ping-pong, then $\mathcal{B}$ runs $b' \leftarrow \mathcal{A}(\perp)$ and outputs the same b'. (ii) Otherwise, if $\mathcal{A}$ was ping-pong (in this case the sender must accept by correctness), then $\mathcal{B}$ runs $b' \leftarrow \mathcal{A}(K_0)$ and returns b'. To see why $\mathcal{B}$'s simulation is correct, observe that $\mathcal{A}$ has to obey essentially the same rules in the security experiments of $\mathsf{ConfEnc}$ and the one of $1\text{−}\mathsf{KE}\text{−}\mathsf{Sec}$. The definition of ping-pong is the same and the only difference between the security experiments is that in $1\text{−}\mathsf{KE}\text{−}\mathsf{Sec}$, if U rejects, then $\mathcal{A}$ can win only with probability 1/2 (recall that in this case it has to distinguish between two keys $K_0 = K_1 = \perp$). In contrast, in $\mathsf{ConfEnc}$, even if the sender rejects, the adversary may have the chance to win the game with probability non-negligibly higher than 1/2. However, it is not hard to see that this asymmetry between the security definitions is not relevant while proving that $ConfPKE$ implies 1-KE. Intuitively, if a non-ping-pong $\mathcal{A}$ makes U accept, the same holds for $\mathcal{B}$ w.r.t. the sender. Otherwise, if a ping-pong $\mathcal{A}$ has non-negligible advantage in distinguishing a real-or-random session key, then $\mathcal{B}$ will also have non-negligible advantage in distinguishing which of the two messages were sent in the challenge session.

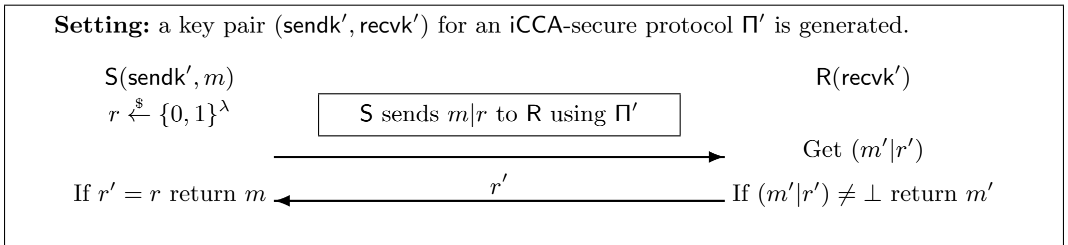

Setting: a key pair $(\mathsf{sendk}', \mathsf{recvk}')$ for an iCCA-secure protocol Π' is generated.
$\mathsf{S}(\mathsf{sendk}', m)$
$\mathsf{R}(\mathsf{recvk}')$
$r \xleftarrow{\$} \{0,1\}^\lambda$
S sends $m|r$ to R using Π'
Get $(m'|r')$
If $r' = r$ return m
r'
If $(m'|r') \neq \perp$ return m'

Fig. 10. ConfPKE from iCCA-secure encryption.

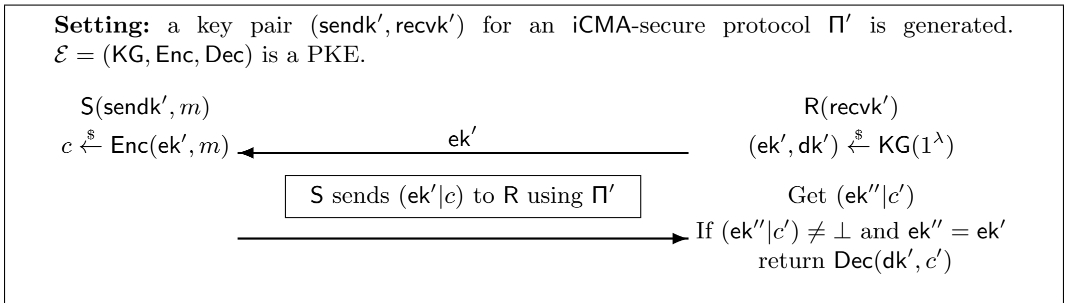

Setting: a key pair $(\mathsf{sendk}', \mathsf{recvk}')$ for an iCMA-secure protocol Π' is generated. $\mathcal{E} = (\mathsf{KG}, \mathsf{Enc}, \mathsf{Dec})$ is a PKE.
$\mathsf{S}(\mathsf{sendk}', m)$
$\mathsf{R}(\mathsf{recvk}')$
$c \xleftarrow{\$} \mathsf{Enc}(\mathsf{ek}', m)$
ek'
$(\mathsf{ek}', \mathsf{dk}') \xleftarrow{\$} \mathsf{KG}(1^\lambda)$
S sends $(\mathsf{ek}'|c)$ to R using Π'
Get $(\mathsf{ek}''|c')$
If $(\mathsf{ek}''|c') \neq \perp$ and $\mathsf{ek}'' = \mathsf{ek}'$
return $\mathsf{Dec}(\mathsf{dk}', c')$

Fig. 11. ConfPKMA from iCMA-secure PKMA and IND−CPA-secure PKE.

Instantiations. Finally, we focus on realizing confirmed encryption and confidential authentication. In particular, we show how to build a ConfPKE scheme based on an iCCA-secure protocol (see Figure 10), and a ConfPKMA scheme based on an iCMA-secure protocol and a (non-interactive) IND−CPA-secure PKE scheme (see Figure 11). The proofs of security of these two protocols are very similar to the ones of Theorems 4 and 5, and are omitted. Finally, to see how the intermediate notions of ConfPKE and ConfPKMA offer a smooth transition from iCCA/iCMA security towards 1-KE, it is interesting to observe that our two constructions of 1-KE in Figures 5 and 6 can be seen as the result of applying (with some optimizations) Lemmas 4 and 5 to the protocol of Figures 10 and 11 respectively. Precisely, we consider the following optimizations: in the protocol of Figure 5 only a single random value r is sent and a PRG is used to expand r in two pseudo-random strings, while the protocol of Figure 6 uses a KEM instead of a PKE.

Acknowledgements. The authors would like to thank Adam O'Neill, Victor Shoup and Stefano Tessaro for valuable discussions on this work.

References

1. M. Bellare, R. Canetti, and H. Krawczyk. A modular approach to the design and analysis of authentication and key exchange protocols (extended abstract). In *30th Annual ACM Symposium on Theory of Computing*, pages 419–428. ACM Press, May 1998.
2. M. Bellare, A. Desai, E. Jokipii, and P. Rogaway. A concrete security treatment of symmetric encryption. In *38th Annual Symposium on Foundations of Computer Science*, pages 394–403. IEEE Computer Society Press, Oct. 1997.
3. M. Bellare, M. Fischlin, S. Goldwasser, and S. Micali. Identification protocols secure against reset attacks. In B. Pfitzmann, editor, *Advances in Cryptology – EUROCRYPT 2001*, volume 2045 of *Lecture Notes in Computer Science*, pages 495–511. Springer, May 2001.
4. M. Bellare and S. Micali. How to sign given any trapdoor function (extended abstract). In *20th Annual ACM Symposium on Theory of Computing*, pages 32–42. ACM Press, May 1988.
5. M. Bellare and S. Micali. How to sign given any trapdoor function. *Journal of the ACM*, 39(1):214–233, 1992.
6. M. Bellare and A. Palacio. GQ and Schnorr identification schemes: Proofs of security against impersonation under active and concurrent attacks. In M. Yung, editor, *Advances in Cryptology – CRYPTO 2002*, volume 2442 of *Lecture Notes in Computer Science*, pages 162–177. Springer, Aug. 2002.
7. M. Bellare and P. Rogaway. Entity authentication and key distribution. In D. R. Stinson, editor, *Advances in Cryptology – CRYPTO'93*, volume 773 of *Lecture Notes in Computer Science*, pages 232–249. Springer, Aug. 1993.

8. M. Bellare and P. Rogaway. Random oracles are practical: A paradigm for designing efficient protocols. In V. Ashby, editor, *ACM CCS 93: 1st Conference on Computer and Communications Security*, pages 62–73. ACM Press, Nov. 1993.
9. J. Camenisch and V. Shoup. Practical verifiable encryption and decryption of discrete logarithms. In D. Boneh, editor, *Advances in Cryptology – CRYPTO 2003*, volume 2729 of *Lecture Notes in Computer Science*, pages 126–144. Springer, Aug. 2003.
10. R. Canetti, S. Halevi, and J. Katz. A forward-secure public-key encryption scheme. In E. Biham, editor, *Advances in Cryptology – EUROCRYPT 2003*, volume 2656 of *Lecture Notes in Computer Science*, pages 255–271. Springer, May 2003.
11. R. Canetti and H. Krawczyk. Analysis of key-exchange protocols and their use for building secure channels. In B. Pfitzmann, editor, *Advances in Cryptology – EUROCRYPT 2001*, volume 2045 of *Lecture Notes in Computer Science*, pages 453–474. Springer, May 2001.
12. R. Canetti, H. Krawczyk, and J. B. Nielsen. Relaxing chosen-ciphertext security. In D. Boneh, editor, *Advances in Cryptology – CRYPTO 2003*, volume 2729 of *Lecture Notes in Computer Science*, pages 565–582. Springer, Aug. 2003.
13. S. G. Choi, D. Dachman-Soled, T. Malkin, and H. Wee. Black-box construction of a non-malleable encryption scheme from any semantically secure one. In R. Canetti, editor, *TCC 2008: 5th Theory of Cryptography Conference*, volume 4948 of *Lecture Notes in Computer Science*, pages 427–444. Springer, Mar. 2008.
14. R. Cramer, G. Hanaoka, D. Hofheinz, H. Imai, E. Kiltz, R. Pass, A. Shelat, and V. Vaikuntanathan. Bounded CCA2-secure encryption. In K. Kurosawa, editor, *Advances in Cryptology – ASIACRYPT 2007*, volume 4833 of *Lecture Notes in Computer Science*, pages 502–518. Springer, Dec. 2007.
15. R. Cramer and V. Shoup. A practical public key cryptosystem provably secure against adaptive chosen ciphertext attack. In H. Krawczyk, editor, *Advances in Cryptology – CRYPTO'98*, volume 1462 of *Lecture Notes in Computer Science*, pages 13–25. Springer, Aug. 1998.
16. R. Cramer and V. Shoup. Signature schemes based on the strong RSA assumption. In *ACM CCS 99: 6th Conference on Computer and Communications Security*, pages 46–51. ACM Press, Nov. 1999.
17. M. Di Raimondo and R. Gennaro. New approaches for deniable authentication. In V. Atluri, C. Meadows, and A. Juels, editors, *ACM CCS 05: 12th Conference on Computer and Communications Security*, pages 112–121. ACM Press, Nov. 2005.
18. M. Di Raimondo, R. Gennaro, and H. Krawczyk. Deniable authentication and key exchange. In A. Juels, R. N. Wright, and S. Vimercati, editors, *ACM CCS 06: 13th Conference on Computer and Communications Security*, pages 400–409. ACM Press, Oct. / Nov. 2006.
19. Y. Dodis, E. Kiltz, K. Pietrzak, and D. Wichs. Message authentication, revisited. In D. Pointcheval and T. Johansson, editors, *Advances in Cryptology – EUROCRYPT 2012*, volume 7237 of *Lecture Notes in Computer Science*, pages 355–374. Springer, Apr. 2012.
20. D. Dolev, C. Dwork, and M. Naor. Non-malleable cryptography (extended abstract). In *23rd Annual ACM Symposium on Theory of Computing*, pages 542–552. ACM Press, May 1991.
21. D. Dolev, C. Dwork, and M. Naor. Nonmalleable cryptography. *SIAM Journal on Computing*, 30(2):391–437, 2000.
22. C. Dwork and M. Naor. Zaps and their applications. In *41st Annual Symposium on Foundations of Computer Science*, pages 283–293. IEEE Computer Society Press, Nov. 2000.
23. C. Dwork, M. Naor, and A. Sahai. Concurrent zero-knowledge. In *30th Annual ACM Symposium on Theory of Computing*, pages 409–418. ACM Press, May 1998.
24. C. Dwork and A. Sahai. Concurrent zero-knowledge: Reducing the need for timing constraints. In H. Krawczyk, editor, *Advances in Cryptology – CRYPTO'98*, volume 1462 of *Lecture Notes in Computer Science*, pages 442–457. Springer, Aug. 1998.
25. A. Fiat and A. Shamir. How to prove yourself: Practical solutions to identification and signature problems. In A. M. Odlyzko, editor, *Advances in Cryptology – CRYPTO'86*, volume 263 of *Lecture Notes in Computer Science*, pages 186–194. Springer, Aug. 1986.
26. D. Fiore, R. Gennaro, and N. P. Smart. Constructing certificateless encryption and ID-based encryption from ID-based key agreement. In M. Joye, A. Miyaji, and A. Otsuka, editors, *PAIRING 2010: 4th International Conference on Pairing-based Cryptography*, volume 6487 of *Lecture Notes in Computer Science*, pages 167–186. Springer, Dec. 2010.
27. R. Gennaro, S. Halevi, and T. Rabin. Secure hash-and-sign signatures without the random oracle. In J. Stern, editor, *Advances in Cryptology – EUROCRYPT'99*, volume 1592 of *Lecture Notes in Computer Science*, pages 123–139. Springer, May 1999.
28. Y. Gertner, T. Malkin, and S. Myers. Towards a separation of semantic and CCA security for public key encryption. In S. P. Vadhan, editor, *TCC 2007: 4th Theory of Cryptography Conference*, volume 4392 of *Lecture Notes in Computer Science*, pages 434–455. Springer, Feb. 2007.
29. I. Goldberg, D. Stebila, and B. Ustaoglu. Anonymity and one-way authentication in key exchange protocols. *Designs, Codes and Cryptography*, 67(2):245–269, 2013.
30. S. Goldwasser, S. Micali, and R. L. Rivest. A "paradoxical" solution to the signature problem (abstract) (impromptu talk). In G. R. Blakley and D. Chaum, editors, *Advances in Cryptology – CRYPTO'84*, volume 196 of *Lecture Notes in Computer Science*, page 467. Springer, Aug. 1984.

31. L. C. Guillou and J.-J. Quisquater. A "paradoxical" indentity-based signature scheme resulting from zero-knowledge. In S. Goldwasser, editor, *Advances in Cryptology – CRYPTO'88*, volume 403 of *Lecture Notes in Computer Science*, pages 216–231. Springer, Aug. 1988.
32. D. Hofheinz and E. Kiltz. Practical chosen ciphertext secure encryption from factoring. In A. Joux, editor, *Advances in Cryptology – EUROCRYPT 2009*, volume 5479 of *Lecture Notes in Computer Science*, pages 313–332. Springer, Apr. 2009.
33. T. Jager, F. Kohlar, S. Schäge, and J. Schwenk. On the security of TLS-DHE in the standard model. In R. Safavi-Naini and R. Canetti, editors, *Advances in Cryptology – CRYPTO 2012*, volume 7417 of *Lecture Notes in Computer Science*, pages 273–293. Springer, Aug. 2012.
34. J. Katz. Efficient and non-malleable proofs of plaintext knowledge and applications. In E. Biham, editor, *Advances in Cryptology – EUROCRYPT 2003*, volume 2656 of *Lecture Notes in Computer Science*, pages 211–228. Springer, May 2003.
35. H. Krawczyk. Skeme: a versatile secure key exchange mechanism for internet. In *Network and Distributed System Security, 1996., Proceedings of the Symposium on*, pages 114 –127, feb 1996.
36. H. Krawczyk, K. G. Paterson, and H. Wee. On the security of the TLS protocol: A systematic analysis. In R. Canetti and J. A. Garay, editors, *Advances in Cryptology – CRYPTO 2013, Part I*, volume 8042 of *Lecture Notes in Computer Science*, pages 429–448. Springer, Aug. 2013.
37. Y. Lindell. A simpler construction of cca2-secure public-key encryption under general assumptions. In E. Biham, editor, *Advances in Cryptology – EUROCRYPT 2003*, volume 2656 of *Lecture Notes in Computer Science*, pages 241–254. Springer, May 2003.
38. U. Maurer, B. Tackmann, and S. Coretti. Key exchange with unilateral authentication: Composable security definition and modular protocol design. Cryptology ePrint Archive, Report 2013/555, 2013. `http://eprint.iacr.org/`.
39. S. Myers and A. Shelat. Bit encryption is complete. In *50th Annual Symposium on Foundations of Computer Science*, pages 607–616. IEEE Computer Society Press, Oct. 2009.
40. M. Naor and O. Reingold. Number-theoretic constructions of efficient pseudo-random functions. In *38th Annual Symposium on Foundations of Computer Science*, pages 458–467. IEEE Computer Society Press, Oct. 1997.
41. M. Naor, O. Reingold, and A. Rosen. Pseudo-random functions and factoring (extended abstract). In *32nd Annual ACM Symposium on Theory of Computing*, pages 11–20. ACM Press, May 2000.
42. M. Naor and M. Yung. Universal one-way hash functions and their cryptographic applications. In *21st Annual ACM Symposium on Theory of Computing*, pages 33–43. ACM Press, May 1989.
43. M. Naor and M. Yung. Public-key cryptosystems provably secure against chosen ciphertext attacks. In *22nd Annual ACM Symposium on Theory of Computing*, pages 427–437. ACM Press, May 1990.
44. C. Rackoff and D. R. Simon. Non-interactive zero-knowledge proof of knowledge and chosen ciphertext attack. In J. Feigenbaum, editor, *Advances in Cryptology – CRYPTO'91*, volume 576 of *Lecture Notes in Computer Science*, pages 433–444. Springer, Aug. 1991.
45. J. Rompel. One-way functions are necessary and sufficient for secure signatures. In *22nd Annual ACM Symposium on Theory of Computing*, pages 387–394. ACM Press, May 1990.
46. A. Sahai. Non-malleable non-interactive zero knowledge and adaptive chosen-ciphertext security. In *40th Annual Symposium on Foundations of Computer Science*, pages 543–553. IEEE Computer Society Press, Oct. 1999.
47. C.-P. Schnorr. Efficient identification and signatures for smart cards. In G. Brassard, editor, *Advances in Cryptology – CRYPTO'89*, volume 435 of *Lecture Notes in Computer Science*, pages 239–252. Springer, Aug. 1989.
48. V. Shoup. On formal models for secure key exchange. Cryptology ePrint Archive, Report 1999/012, 1999. `http://eprint.iacr.org/`.
49. B. R. Waters. Efficient identity-based encryption without random oracles. In R. Cramer, editor, *Advances in Cryptology – EUROCRYPT 2005*, volume 3494 of *Lecture Notes in Computer Science*, pages 114–127. Springer, May 2005.
50. H. Wee. Efficient chosen-ciphertext security via extractable hash proofs. In T. Rabin, editor, *Advances in Cryptology – CRYPTO 2010*, volume 6223 of *Lecture Notes in Computer Science*, pages 314–332. Springer, Aug. 2010.

A Standard Cryptographic Primitives

We describe notation and recall some basic definitions that will be useful in our work. We denote with $\lambda \in \mathbb{N}$ a security parameter, and we say that a function $\epsilon(\lambda)$ is *negligible* if it vanishes faster than the inverse of any polynomial in λ. If X is a set, we denote with $x \overset{\$}{\leftarrow} X$ the process of selecting x uniformly at random in S. An algorithm $\mathcal{A}$ is called PPT if it is a probabilistic Turing machine whose running time is bounded by some polynomial in λ. If $\mathcal{A}$ is a PPT algorithm, then $y \overset{\$}{\leftarrow} \mathcal{A}(x)$ indicates the process of running $\mathcal{A}$ on input x and assigning its output to y.

A.1 Pseudorandom Generators

Let λ be the security parameter, and ℓ, L be polynomials in λ such that $\ell < L$. A function $G : \{0,1\}^\ell \to \{0,1\}^L$ is a pseudorandom generator (PRG) if for any PPT adversary $\mathcal{A}$ its advantage

$$\mathbf{Adv}_{\mathcal{A},G}(\lambda) = \left|\Pr[\mathcal{A}(y) = 1 : y = G(s), s \xleftarrow{\$} \{0,1\}^\ell] - \Pr[\mathcal{A}(y) = 1 : y \xleftarrow{\$} \{0,1\}^L]\right|$$

is at most negligible.

A.2 (Non-Interactive) CCA-secure Public-Key Encryption

A public key encryption scheme $\mathcal{E}$ is a tuple of algorithms $(\mathsf{KG}, \mathsf{Enc}, \mathsf{Dec})$ defined as follows:

$\mathsf{KG}(1^\lambda)$ on input the security parameter, the key generation returns a public key ek and a secret key dk.

$\mathsf{Enc}(\mathsf{ek}, m)$ on input the public key ek and a message m, it outputs a ciphertext c.

$\mathsf{Dec}(\mathsf{dk}, c)$ given the secret key dk and a ciphertext c, it outputs a message m or an error symbol $\perp$.

Consider the following experiment involving the scheme $\mathcal{E}$ and an adversary $\mathcal{A}$:

```
Experiment $\mathbf{Exp}^{\mathsf{IND-CCA}}_{\mathcal{E},\mathcal{A}}(\lambda)$
    $b \xleftarrow{\$} \{0,1\}$
    $(\mathsf{ek}, \mathsf{dk}) \xleftarrow{\$} \mathsf{KG}(1^\lambda)$
    $(m_0, m_1) \leftarrow \mathcal{A}^{\mathsf{Dec}(\mathsf{dk},\cdot)}(\mathsf{ek})$
    $c^* \xleftarrow{\$} \mathsf{Enc}(\mathsf{ek}, m_b)$
    $b' \leftarrow \mathcal{A}^{\mathsf{Dec}(\mathsf{dk},\cdot)}(c^*)$
    If $b' = b$ and $\mathcal{A}$ is "legal" output 1
    Else output 0.
```

In the above experiment, $\mathcal{A}$ is called "legal" if it does not query the decryption oracle $\mathsf{Dec}(\mathsf{dk}, \cdot)$ on the challenge ciphertext c^* (after $\mathcal{A}$ receives c^*).

The advantage of an adversary $\mathcal{A}$ in breaking the $\mathsf{IND-CCA}$ security of an encryption scheme $\mathcal{E}$ is

$$\mathbf{Adv}^{\mathsf{IND-CCA}}_{\mathcal{E},\mathcal{A}}(\lambda) = \left|\Pr[\mathbf{Exp}^{\mathsf{IND-CCA}}_{\mathcal{E},\mathcal{A}}(\lambda) = 1] - \frac{1}{2}\right|$$

Definition 15 **($\mathsf{IND-CCA}$ security).** *An encryption scheme $\mathcal{E}$ is $\mathsf{IND-CCA}$-secure if for any PPT $\mathcal{A}$, $\mathbf{Adv}^{\mathsf{IND-CCA}}_{\mathcal{E},\mathcal{A}}(\lambda)$ is negligible.*

A weaker notion of $\mathsf{IND-CCA}$ security that we consider in our work is *q-bounded* $\mathsf{IND-CCA}$ security [14]. This notion is defined as $\mathsf{IND-CCA}$ security except that the adversary is restricted to query the decryption oracle at most q times (where q is a pre-fixed bound).

A further weaker notion of security for public key encryption is semantic security, or indistinguishability against chosen-plaintext attacks ($\mathsf{IND-CPA}$). Its definition is the same as $\mathsf{IND-CCA}$ security except that the adversary does not get access to any decryption oracle.

Finally, we recall the notion of *key encapsulation mechanism* (KEM) which is closely related to public key encryption. A KEM is defined by three algorithms $(\mathsf{KG}, \mathsf{Encap}, \mathsf{Decap})$: the key generation KG is the same as in PKE; the probabilistic encapsulation algorithm Encap uses the public key ek to generate a ciphertext C and a key K; the decapsulation algorithm Decap takes as input the secret key dk and a ciphertext C and outputs a key K. For correctness, it is required that for all honestly generated pairs of keys $(\mathsf{ek}, \mathsf{dk})$, and $(C, K) \xleftarrow{\$} \mathsf{Encap}(\mathsf{ek})$ it holds $K = \mathsf{Decap}(\mathsf{dk}, C)$. The security definition of KEM is basically the same as that of PKE except that the goal of the adversary is to distinguish a honestly generated session key from a random one. It is worth noting that the standard Diffie-Hellman protocol (aka a simplified version of ElGamal) is a KEM: Let $\mathbb{G}$ be a cyclic group where DDH holds and $g \in \mathbb{G}$ be a generator: KG outputs $\mathsf{ek} = g^x$ and $\mathsf{dk} = x$ for a random x, Encap outputs $C = g^y$ $K = g^{xy}$ for a random y, and $\mathsf{Decap}(\mathsf{dk}, C)$ recovers the same $K = C^x = g^{xy}$.

A.3 Digital Signatures

A digital signature scheme consists of a triple of algorithms $\Sigma = (\Sigma.\mathsf{kg}, \mathsf{Sign}, \mathsf{Ver})$ working as follows:

$\Sigma.\mathsf{kg}(1^\lambda)$ the key generation takes as input a security parameter λ and returns a pair of keys $(\mathsf{sk}, \mathsf{vk})$.
$\mathsf{Sign}(\mathsf{sk}, m)$ on input a signing key sk and a message m, the signing algorithm produces a signature σ.
$\mathsf{Ver}(\mathsf{vk}, m, \sigma)$ given a triple vk, m, σ the verification algorithm tests if σ is a valid signature on m with respect to verification key vk.

For security we define the following experiment:

```
Experiment $\mathbf{Exp}^{\mathsf{uf\text{-}cma}}_{\mathcal{A},\Sigma}(\lambda)$
    $(\mathsf{sk}, \mathsf{vk}) \xleftarrow{\$} \Sigma.\mathsf{kg}(1^\lambda)$
    $(m^*, \sigma^*) \xleftarrow{\$} \mathcal{A}^{\mathsf{Sign}(\mathsf{sk},\cdot)}(\mathsf{vk})$
    If $\mathsf{Ver}(\mathsf{vk}, m^*, \sigma^*) = 1$ and $(m^*, \sigma^*)$ is "new" then output 1
    Else Output 0
```

We say that the forgery (m^*, σ^*) is "new" if it is different from all the pairs (m_i, σ_i) obtained from the signing oracle $\mathsf{Sign}(\mathsf{sk}, \cdot)$. We define the advantage of an adversary $\mathcal{A}$ in breaking the strong unforgeability against chosen-message attacks (suf-cma) of Σ as $\mathbf{Adv}^{\mathsf{suf\text{-}cma}}_{\mathcal{A},\Sigma}(\lambda) = \Pr[\mathbf{Exp}^{\mathsf{suf\text{-}cma}}_{\mathcal{A},\Sigma}(\lambda) = 1]$.

Definition 16 (suf-cma security). *A digital signature scheme Σ is* suf-cma*-secure if for any PPT $\mathcal{A}$, $\mathbf{Adv}^{\mathsf{suf\text{-}cma}}_{\mathcal{A},\Sigma}(\lambda)$ is negligible.*

A weaker notion of security is (simple) unforgeability against chosen-message attacks (uf-cma), which is defined as the strong version above, except that (m^*, σ^*) is considered "new" if only the message m^* (instead of the pair) is different from all messages m_i queried to the signing oracle.

A.4 Message Authentication Codes

A message authentication code consists of a triple of algorithms $\mathsf{MAC} = (\mathsf{Gen}, \mathsf{Tag}, \mathsf{Ver})$ working as follows:

$\mathsf{Gen}(1^\lambda)$**:** the key generation algorithm takes as input the security parameter λ and returns a key $k \in \mathcal{K}$.
$\mathsf{Tag}(k, m)$**:** on input a secret key $k \in \mathcal{K}$ and a message $m \in \mathcal{M}$, the authentication algorithm produces an authentication tag σ.
$\mathsf{Ver}(k, m, \sigma)$**:** given the secret key k, a message m and an authentication tag σ, the verification algorithm tests if σ correctly authenticates m.

A scheme MAC is correct if for all $\lambda \in \mathbb{N}$ and $m \in \mathcal{M}$, the probability $\Pr[\mathsf{Ver}(k, m, \sigma) = 1 : k \xleftarrow{\$} \mathsf{Gen}(1^\lambda), \sigma \xleftarrow{\$} \mathsf{Tag}(k, m)]$ is overwhelming. The security is defined via the following experiment:

```
Experiment $\mathbf{Exp}^{\mathsf{suf\text{-}cmva}}_{\mathcal{A},\mathsf{MAC}}(\lambda)$
    $k \xleftarrow{\$} \mathsf{Gen}(1^\lambda)$
    Run $\mathcal{A}^{\mathsf{Tag}(k,\cdot),\mathsf{Ver}(k,\cdot,\cdot)}(\lambda)$
    If $\mathcal{A}$ makes a verification query $(m^*, \sigma^*)$ such that $\mathsf{Ver}(k, m^*, \sigma^*) = 1$ and $(m^*, \sigma^*)$ is "new",
      then output 1.
    Else output 0
```

where for (m^*, σ^*) being "new" we mean that it must be different from all the pairs (m_i, σ_i) obtained from the tag oracle $\mathsf{Tag}(k, \cdot)$. The advantage of an adversary $\mathcal{A}$ in breaking the strong unforgeability against chosen-message and chosen verification queries attacks (suf-cmva) of MAC is $\mathbf{Adv}^{\mathsf{suf\text{-}cmva}}_{\mathcal{A},\mathsf{MAC}}(\lambda) = \Pr[\mathbf{Exp}^{\mathsf{suf\text{-}cmva}}_{\mathcal{A},\mathsf{MAC}}(\lambda) = 1]$.

Definition 17 (suf-cmva security). *A message authentication code* MAC *is* suf-cmva*-secure if for any PPT $\mathcal{A}$, $\mathbf{Adv}^{\mathsf{suf\text{-}cmva}}_{\mathcal{A},\mathsf{MAC}}(\lambda)$ is negligible.*

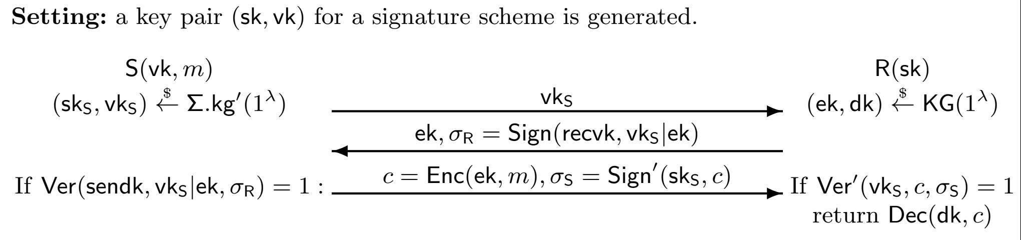

Setting: a key pair $(\mathsf{sk}, \mathsf{vk})$ for a signature scheme is generated.
$\mathsf{S}(\mathsf{vk}, m)$
$\mathsf{R}(\mathsf{sk})$
$(\mathsf{sk_S}, \mathsf{vk_S}) \overset{\$}{\leftarrow} \Sigma.\mathsf{kg}'(1^\lambda)$
$\mathsf{vk_S}$
$(\mathsf{ek}, \mathsf{dk}) \overset{\$}{\leftarrow} \mathsf{KG}(1^\lambda)$
$\mathsf{ek}, \sigma_\mathsf{R} = \mathsf{Sign}(\mathsf{recvk}, \mathsf{vk_S}|\mathsf{ek})$
If $\mathsf{Ver}(\mathsf{sendk}, \mathsf{vk_S}|\mathsf{ek}, \sigma_\mathsf{R}) = 1$:
$c = \mathsf{Enc}(\mathsf{ek}, m), \sigma_\mathsf{S} = \mathsf{Sign}'(\mathsf{sk_S}, c)$
If $\mathsf{Ver}'(\mathsf{vk_S}, c, \sigma_\mathsf{S}) = 1$
return $\mathsf{Dec}(\mathsf{dk}, c)$

Fig. 12. DDN 3-round iCCA-secure protocol from IND−CPA-secure encryption.

B DDN 3-round iCCA-Secure Encryption from IND−CPA Security

In this section we recall the 3-round iCCA-secure PKE proposed by Dolev, Dwork and Naor [21], which is based on IND−CPA-secure PKE and signature schemes.

Let $\Sigma = (\Sigma.\mathsf{kg}, \mathsf{Sign}, \mathsf{Ver})$ be a (regular) signature scheme, $\Sigma' = (\Sigma.\mathsf{kg}', \mathsf{Sign}', \mathsf{Ver}')$ be a one-time signature scheme, and $\mathcal{E} = (\mathsf{KG}, \mathsf{Enc}, \mathsf{Dec})$ be a (non-interactive) public key encryption scheme. Using our syntax, the DDN protocol $\Pi_{DDN} = (\mathsf{Setup}, \mathsf{S}, \mathsf{R})$ is as follows:

$\mathsf{Setup}(1^\lambda)$**:** run $(\mathsf{sk}, \mathsf{vk}) \overset{\$}{\leftarrow} \Sigma.\mathsf{kg}(1^\lambda)$ and output $\mathsf{sendk} = \mathsf{vk}$ and $\mathsf{recvk} = \mathsf{sk}$.

$\mathsf{S}(\mathsf{sendk}, m)$**:** first generate a fresh one-time signing key pair $(\mathsf{sk_S}, \mathsf{vk_S}) \overset{\$}{\leftarrow} \Sigma.\mathsf{kg}'(1^\lambda)$ and send $\mathsf{vk_S}$ to R.
Upon receiving the second message from R, the sender checks that $\mathsf{Ver}(\mathsf{sendk}, \mathsf{vk_S}|\mathsf{ek}, \sigma_\mathsf{R}) = 1$, and if so computes $c \overset{\$}{\leftarrow} \mathsf{Enc}(\mathsf{ek}, m)$ and $\sigma_\mathsf{S} \overset{\$}{\leftarrow} \mathsf{Sign}'(\mathsf{sk_S}, c)$, and sends (c, σ_S).

$\mathsf{R}(\mathsf{recvk})$**:** upon receipt of the first message $\mathsf{vk_S}$ from S, generate a fresh encryption key pair $(\mathsf{ek}, \mathsf{dk}) \overset{\$}{\leftarrow} \mathsf{KG}(1^\lambda)$, compute $\sigma_\mathsf{R} \overset{\$}{\leftarrow} \mathsf{Sign}(\mathsf{recvk}, \mathsf{vk_S}|\mathsf{ek})$ and send $(\mathsf{ek}, \sigma_\mathsf{R})$ to S.
Once the message (c, σ_S) is obtained, the receiver checks that $\mathsf{Ver}'(\mathsf{vk_S}, c, \sigma_\mathsf{S}) = 1$, and if so returns $m \leftarrow \mathsf{Dec}(\mathsf{dk}, c)$ as its private output.

A pictorial description of Π_{DDN} is given in Fig. 12 while we sketch its proof of security under our iCCA notion below.

Theorem 13. *If $\mathcal{E}$ is* IND−CPA*-secure, Σ is* uf-cma*-secure, and Σ' is one-time* uf-cma*-secure, then Π_{DDN} is* iCCA*-secure.*

Proof (Sketch). Consider the experiment $\mathbf{Exp}^{\mathsf{iCCA}}_{\Pi_{DDN},\mathcal{A}}$, and let vk^*_S and $(c^*, \sigma^*_\mathsf{S})$ be the first and the third protocol messages, respectively, generated by the honest sender in the challenge session. Let Forge_1 be the event that in the challenge session the adversary $\mathcal{A}$ outputs a second protocol message including a valid signature σ_R that was *not* obtained by the R oracle. It is not hard to see that under the assumption that the signature scheme Σ is secure we have that $\Pr[\mathsf{Forge}_1]$ is negligible. Also, let Forge_2 be the event that the adversary queries the receiver oracle with a third protocol message (c, σ_S) such that: the first protocol message of the given session is vk^*_S (i.e., the same verification key as in the challenge session), but c is "new", in particular $c \neq c^*$. Again, it is possible to show that under the assumption that Σ' is a one-time signature the probability $\Pr[\mathsf{Forge}_2]$ is negligible.

Finally, it is possible to show that $\Pr[\mathbf{Exp}^{\mathsf{iCCA}}_{\Pi_{DDN},\mathcal{A}} \mid \overline{\mathsf{Forge}_1} \wedge \overline{\mathsf{Forge}_2}]$ is negligible under the assumption that $\mathcal{E}$ is IND−CPA-secure. The main observation is that one can simulate answers of the receiver oracle to third messages (i.e., decryptions). This follows from the fact that when Forge_1 and Forge_2 do not occur, a non-ping-pong adversary cannot create a valid third message that uses the same one-time signature/encryption keys of the challenge session.

C A 1-bounded $\mathsf{IND-CCA}$-Secure Encryption Scheme

Here we recall a black-box construction of a 1-bounded $\mathsf{IND-CCA}$-secure encryption scheme from an $\mathsf{IND-CPA}$-secure one. The scheme is a (simplified) special case of the q-bounded one of Cramer et al. [14], but it is also very similar to the scheme of Dolev, Dwork and Naor [21] without NIZK proofs.

Let $\mathcal{E}' = (\mathsf{KG}', \mathsf{Enc}', \mathsf{Dec}')$ be a semantic secure public key encryption scheme. The $\mathsf{IND-CCA}$ scheme $\mathcal{E} = (\mathsf{KG}, \mathsf{Enc}, \mathsf{Dec})$ is defined as follows:

$\mathsf{KG}(1^\lambda)$**:** Generate $2n$ keys $(\mathsf{ek}_i^b, \mathsf{dk}_i^b) \xleftarrow{\$} \mathsf{KG}'(1^\lambda)$ $\forall i = 1, \ldots, n$, $b = 0, 1$, and a random universal one-way hash function h. Output the public key $\mathsf{ek} = (h, \mathsf{ek}_1^0, \mathsf{ek}_1^1, \ldots, \mathsf{ek}_n^0, \mathsf{ek}_n^1)$ and the secret key $\mathsf{dk} = (\mathsf{dk}_1^0, \mathsf{dk}_1^1, \ldots, \mathsf{dk}_n^0, \mathsf{dk}_n^1)$.

$\mathsf{Enc}(\mathsf{ek}, m)$**:** First, generate a pair of keys $(\mathsf{sk}, \mathsf{vk})$ for a one-time signature, and compute $v = h(\mathsf{vk})$ (write $v = v_1 \cdots v_n$ using its bitwise representation). Next, generate random messages $m_1, \ldots, m_n$ such that $m_1 \oplus m_2 \oplus \cdots \oplus m_n = m$. Then, for $i = 1$ to n, compute $c_i \xleftarrow{\$} \mathsf{Enc}(\mathsf{ek}_i^{v_i}, m_i)$, create a signature σ on $(c_1, \ldots, c_n)$ using sk, and output $C = (\mathsf{vk}, \sigma, c_1, \ldots, c_n)$.

$\mathsf{Dec}(\mathsf{dk}, C)$**:** Verify that σ is a valid signature on $(c_1, \ldots, c_n)$ using verification key vk. Then compute $v = h(\mathsf{vk})$ and decrypt each ciphertext computing $m_i \leftarrow \mathsf{Dec}(\mathsf{dk}_i^{v_i})$. If the signature is valid output $m = m_1 \oplus m_2 \oplus \cdots \oplus m_n$. Otherwise output $\perp$ (reject).

D iCCA-Secure Interactive Encryption with Labels

In an encryption scheme with "labels" the encryption and decryption algorithms are assumed to take a public string called label as an additional input. While this notion has been already introduced in the non-interactive setting [9], here we propose its natural extension to the interactive scenario. We will show an elegant application of labeled iCCA encryption to our notion of public key message authentication (see Section 2.3).

An interactive encryption protocol with labels is basically the same as the one defined in Section 2.2, except that both sender and receiver work with a public label L (i.e., an arbitrary binary string) as additional input. For any label L we denote the algorithms with $\mathsf{S}_L(\mathsf{sendk}, m)$ and $\mathsf{R}_L(\mathsf{recvk})$. For correctness, we require that for all honestly generated keys $(\mathsf{sendk}, \mathsf{recvk})$, all messages $m \in \mathcal{M}$ and all labels L, $\langle \mathsf{S}_L(\mathsf{sendk}, m), \mathsf{R}_L(\mathsf{recvk}) \rangle = m$ holds with all but negligible probability.

The iCCA security notion is extended to the labeled case as follows: when the adversary queries the receiver oracle, it also specifies a label L, i.e., $\mathcal{A}$ interacts with $\mathsf{R}_L(\mathsf{recvk})$. The adversary $\mathcal{A}$ is also required to output a label L^* along with the message pair (m_0, m_1), and the challenge session is $\langle \mathsf{S}_{L^*}(\mathsf{sendk}, m_b), \mathcal{A}^{\mathsf{R}_{(\cdot)}(\mathsf{recvk})} \rangle$. The restriction on $\mathcal{A}$ not to be ping-pong is extended in such a way that for every oracle session with transcript T and label L, it must hold $L \neq L^*$ or $T \not\equiv T^*$.

Building iCCA Encryption with Labels from iCCA Encryption. We show that iCCA encryption with labels can be realized from iCCA-secure encryption. The idea of the construction is the same as for the non-interactive case: if L is the label, then the encryption protocol is run with plaintext $m_L = L|m$, i.e., the concatenation of the label and the plaintext m; when the decryptor obtains a plaintext $m'_L = L'|m'$ it checks whether it obtained the label L, i.e., if $L' = L$.

A formal description of this construction follows. Let $\Pi = (\mathsf{Setup}, \mathsf{S}, \mathsf{R})$ be an interactive encryption protocol. We build an interactive encryption protocol with labels $\Pi' = (\mathsf{Setup}', \mathsf{S}', \mathsf{R}')$ as follows. The key generation is the same $\mathsf{Setup}' = \mathsf{Setup}$, thus $\mathsf{sendk}' = \mathsf{sendk}, \mathsf{recvk}' = \mathsf{recvk}$. The sender algorithm $\mathsf{S}'_L(\mathsf{sendk}, m)$ simply runs $\mathsf{S}(\mathsf{sendk}, L|m)$ (i.e., on the concatenation of the message m and the label L). The receiver algorithm $\mathsf{R}'_L(\mathsf{recvk})$ runs $\mathsf{R}(\mathsf{recvk})$. At the end of a session, if $m'_L = (L'|m')$ is the the message returned by R, then R' returns m' only if $L' = L$. Otherwise, it returns $\perp$.

We show the security of this construction via the following theorem.

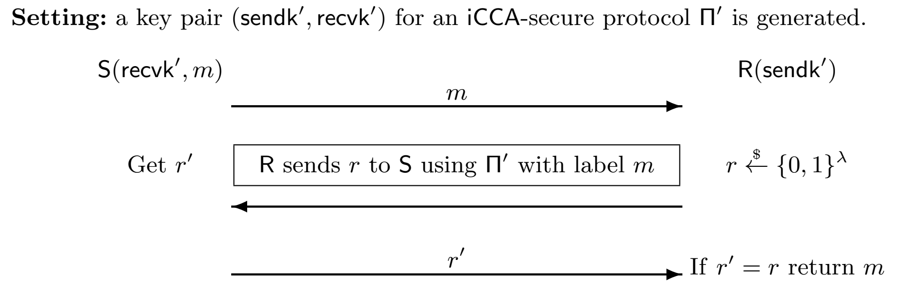

Setting: a key pair $(\mathsf{sendk}', \mathsf{recvk}')$ for an iCCA-secure protocol Π' is generated.
$\mathsf{S}(\mathsf{recvk}', m)$
$\mathsf{R}(\mathsf{sendk}')$
m
Get r'
R sends r to S using Π' with label m
$r \xleftarrow{\$} \{0,1\}^\lambda$
r'
If $r' = r$ return m

Fig. 13. iCMA-secure protocol from labeled iCCA-secure encryption.

Theorem 14. *If Π is a iCCA-secure encryption protocol, then Π' is a iCCA-secure encryption protocol with labels.*

Proof. Assume by contradiction that there exists a PPT adversary $\mathcal{A}$ that breaks the iCCA security of the labeled scheme Π', then we show how to build a PPT algorithm $\mathcal{B}$ that uses $\mathcal{A}$ as a subroutine to break the iCCA security of Π.

$\mathcal{B}$ is given sendk, it runs $\mathcal{A}(\mathsf{sendk})$ and answers oracle queries as follows. When $\mathcal{A}$ will query R' on a new session, $\mathcal{A}$ has to provide a label L. So, $\mathcal{B}$ stores L, queries its oracle, R, and returns its answer to $\mathcal{A}$. All the oracle queries to R' for sessions that already started are simply forwarded by $\mathcal{B}$ to its R oracle. When $\mathcal{A}$ queries R' on the last message, $\mathcal{B}$ first forwards this query to R, obtains $m = (L'|m')$, and returns to $\mathcal{A}$ m', if $L = L'$ (where L is the label stored for this session), and $\perp$ otherwise.

When $\mathcal{A}$ outputs a message pair (m_0, m_1) and a label L^*, $\mathcal{B}$ starts its challenge session by returning $(L^*|m_0, L^*|m_1)$ as its message pair. Next, for all subsequent messages $\mathcal{B}$ simply forwards them to its challenge session, and forwards back to $\mathcal{A}$ the respective responses. Let $T^* = \langle (M_1^*, t_1^*), \ldots, (M_n^*, t_n^*) \rangle$ be the transcript of the challenge session between $\mathcal{B}$ (who is simulating $\mathsf{S}'_{L^*}(\mathsf{sendk}, m_b)$) and $\mathcal{A}$.

In this second phase, $\mathcal{B}$ answers all oracle queries as before except for the following change. Assume that $\mathcal{A}$ sends the last message M_n in some session whose label is L and whose transcript is $T = \langle (M_1, t_1), \ldots, (M_n, t_n) \rangle$. If $T \equiv T^*$ but $L \neq L^*$, then $\mathcal{B}$ returns $\perp$ as the R' output. Note that by correctness of Π, R would decrypt T to some message of the form $L^*|m_b$. Hence, as $L \neq L^*$ R' would reject this message, that is $\mathcal{B}$'s answer is correctly distributed. If b' is the output of $\mathcal{A}$ at the end of the simulation, $\mathcal{B}$ outputs the same value. To complete the proof, we observe that if $\mathcal{A}$ is not ping-pong, then $\mathcal{B}$ is not ping-pong.

E iCMA-secure PKMA from labeled iCCA-secure encryption.

Here we consider another protocol that achieves public key message authentication based on iCCA-secure encryption. The basic protocol was first proposed by Dolev, Dwork, and Naor in [21], and later discussed by Dwork, Naor, and Sahai [23,24]. Here we revisit this construction by using the abstraction of encryption with labels, and by using our notion of (possibly) interactive iCCA-secure encryption (see Section D). A sketch of the protocol is summarized in Figure 13: (1) S sends the message m to R; (2) R picks a random string r and confidentially transmits r with label m to S by using the encryption protocol Π'; (3) S obtains r' and hands r' to the receiver. If R obtains the same r, then it accepts the message m.

To improve on round complexity, one can use even a 1-round (aka non-interactive) iCCA-secure labeled encryption scheme. In this case, our construction yields a 3-round iCMA-secure protocol. However, we note that the first round, where S sends m to R, can be dropped in all those applications where the message is implicitly known to the receiver (e.g., it is some public information). This way, one obtains an optimal 2-round protocol.

More formally, if $\Pi' = (\mathsf{Setup}', \mathsf{S}', \mathsf{R}')$ is an encryption protocol with labels with message space $\tilde{\mathcal{M}}$ such that $|\tilde{\mathcal{M}}| \approx 2^\lambda$, then we build a PKMA protocol $\Pi = (\mathsf{Setup}, \mathsf{S}, \mathsf{Ver})$ as follows.

$\mathsf{Setup}(1^\lambda)$**:** run $(\mathsf{sendk}', \mathsf{recvk}') \xleftarrow{\$} \mathsf{Setup}'(1^\lambda)$ and output $\mathsf{sendk} = \mathsf{recvk}'$ and $\mathsf{recvk} = \mathsf{sendk}'$.

$\mathsf{S}(\mathsf{sendk}, m)$**:** as first protocol message, send m. Next, run the encryption protocol Π' by playing the role of the receiver. More precisely, use m as the label, and run $\mathsf{R}'_m(\mathsf{recvk}')$. Let r' be the output of R' at the end of the encryption protocol. Finally, send r' to R as the last message.

$\mathsf{R}(\mathsf{recvk})$**:** first, wait for message m from S. Next, choose a random string $r \xleftarrow{\$} \tilde{\mathcal{M}}$ and run the encryption protocol Π' by playing the role of the sender. More precisely, use m as the label and run $\mathsf{S}'_m(\mathsf{sendk}', r)$. If the encryption protocol terminates correctly, then wait for the last message r' from S. If $r' = r$ then output m. Otherwise, output $\perp$.

We can now prove the following theorem.

Theorem 15. *If Π' is* iCCA*-secure encryption with labels, with message space of size at least 2^λ, then the protocol Π described above is* iCMA*-secure.*

Proof. To prove the theorem we define the following hybrid games and we denote with G_i the event that the outcome of Game i, run with $\mathcal{A}$, is 1.

Game 0: this is the real iCMA game.

Game 1: this is the same as Game 0 except that in the challenge session the challenger generates two random strings $r^* \xleftarrow{\$} \tilde{\mathcal{M}}$, and it runs the encryption protocol Π' with a fixed string r_1^* (e.g., $r_1^* = 0^\lambda$), i.e., $\mathcal{B}$ runs $\mathsf{S}'_{m^*}(\mathsf{sendk}', r_1^*)$. However, the challenger keeps using r_0^* as the random string associated with the run of the encryption protocol in the challenge session, i.e., the remaining part of the game (outside the encryption protocol) remains the same as Game 0. Precisely, this means that the challenger uses r_0^* to check the validity of the last message r' provided by the adversary in the challenge session, i.e., it checks whether $r' = r_0^*$.

Via a straightforward reduction to the iCCA-security of the encryption protocol, it is possible to show that there exists an adversary $\mathcal{B}$ such that

$$|\Pr[G_0] - \Pr[G_1]| \leq 2 \cdot \mathbf{Adv}^{\mathsf{iCCA}}_{\Pi',\mathcal{B}}(\lambda)$$

Finally, if we analyze Game 1, it is not hard to see that in order to let Game 1 output 1, the adversary has to correctly guess the value of r_0^*. However, in Game 1, the string r_0^* is randomly chosen and is not used in the encryption protocol. Hence, its value is information-theoretically hidden to any adversary $\mathcal{A}$. Hence, we have that $\Pr[G_1] \leq \frac{1}{2^\lambda}$, which is negligible, as desired.

F Postponed Proofs

F.1 Proof of Claim 1

Proof (Claim 1). Assume by contradiction that there exists an efficient adversary $\mathcal{A}$ such that $\Pr[\mathsf{Forge}] \geq \epsilon$ for a non-negligible value ϵ. Then we show how to build a PPT adversary $\mathcal{B}$ that has non-negligible advantage against the iCMA security of the message transmission protocol Π'.

$\mathcal{B}$ is given in input the public receiver key recvk' and it has oracle access to the sender $\mathsf{S}'(\mathsf{sendk}', \cdot)$. $\mathcal{B}$ proceeds as follows:

- Run $\mathcal{A}(\mathsf{sendk})$ with $\mathsf{sendk} = \mathsf{recvk}'$.
- For every oracle query asking to interact with a new copy of R: generate a new encryption key pair $(\mathsf{ek}, \mathsf{dk}) \xleftarrow{\$} \mathsf{KG}(1^\lambda)$, and query $\mathsf{S}'(\mathsf{sendk}', \mathsf{ek})$ to simulate the first part of the protocol.

- When the adversary makes the last query to R on a session that is already opened: let ek be the first message previously generated in the above step, and let c be the message sent by $\mathcal{A}$ for this query. $\mathcal{B}$ answers by computing $m \leftarrow \mathsf{Dec}(\mathsf{dk}, C)$ (where dk is the decryption key generated by $\mathcal{B}$ together with ek).
- When $\mathcal{A}$ outputs the message pair (m_0, m_1), $\mathcal{B}$ also starts its challenge session, and forwards all $\mathcal{A}$'s messages of the subprotocol Π' to its challenger. If Π' in the challenge session terminates with ek^*, then $\mathcal{B}$ returns ek^*.

If Forge occurs then, by the definition of the event, $\mathcal{A}$ is not ping-pong and thus $\mathcal{B}$ is not ping-pong either. Hence, $\mathbf{Adv}^{\mathsf{suf\text{-}cma}}_{\mathcal{B},\Sigma}(\lambda) = \Pr[\mathsf{Forge}] \geq \epsilon$, which concludes the proof of the claim. $\square$

F.2 Proof of Theorem 3 (iCCA security)

Assume by contradiction there exists a PPT adversary $\mathcal{A}$ that breaks the iCCA security of Π', then we show how to build a PPT algorithm $\mathcal{B}$ that uses $\mathcal{A}$ as a subroutine to break the iCCA security of Π.

Here too, we distinguish between the above two cases according to which party speaks first.

1. R **speaks first.** $\mathcal{B}$ is run on input sendk. It runs $\mathcal{A}(\mathsf{sendk})$ and answers oracle queries as follows. Since in Π' it is S' who speaks first, when $\mathcal{A}$ will query R' to start a new session, $\mathcal{A}$ will also send a nonce $r \in \{0,1\}^\lambda$. $\mathcal{B}$ stores r, queries R and returns its answer to $\mathcal{A}$. R' oracle queries for sessions that already started before are forwarded by $\mathcal{B}$ to R, except for the following change for the last message. $\mathcal{B}$ forwards the message received from $\mathcal{A}$ to R. Let $(m'|r')$ be its response, and let r be the string stored for this session by $\mathcal{B}$. If $r = r'$ then $\mathcal{B}$ returns m'. Otherwise, it returns $\perp$.
   When $\mathcal{A}$ outputs a message pair (m_0, m_1) $\mathcal{B}$ chooses a random string $r^* \overset{\$}{\leftarrow} \{0,1\}^\lambda \setminus \{r_1, \ldots, r_Q\}$ (where $r_1, \ldots, r_Q$ are all the strings sent by $\mathcal{A}$ to the oracle R' in the previous phase), and sends r^* to $\mathcal{A}$. Note that as long as the r string is sufficiently large, the distribution of this choice of r^* is statistically close to the one in the real experiment. So, $\mathcal{B}$ outputs $(r^*|m_0, r^*|m_1)$ as its message pair, and continues the simulation by forwarding the remaining messages of the challenge session to its challenger.
   In this second phase, all oracle queries are answered as before, except for the following change. Assume that the challenge session is completed, let $T^* = \langle (r^*, t_1^*), (M_1^*, t_2^*), \ldots, (M_n^*, t_{n+1}^*) \rangle$ be its transcript, and assume that $\mathcal{A}$ queries R' on an already opened session by sending the last message. $\mathcal{B}$ should then answer with the output of R'. Let $T = \langle (r, t_1), (M_1, t_2), \ldots, (M_n, t_{n+1}) \rangle$ be the transcript of the queried session. Since $\mathcal{A}$ is not ping-pong, $T \not\equiv T^*$, and we would like to argue that $\mathcal{B}$ must not be ping-pong as well while still being capable to answer queries correctly.
   Let us write $T^* = (r^*, t_1^*), T^{*'}$ and $T = (r, t_1), T'$ (i.e., we explicitly separate the first message from the transcript of the n-round protocol). If $T \not\equiv T^*$, either one of the following cases holds:
   - $T' \not\equiv T^{*'}$: $\mathcal{B}$ is not ping-pong in its game and it can answer the query by forwarding $\mathcal{A}$'s last message to its oracle R.
   - $T' \equiv T^{*'}$, $t_1^* > t_1$ and $r = r^*$: $t_1^* > t_1$ means that r^* was generated after r has been sent from $\mathcal{A}$. However, by construction of the simulation it must be $r^* \neq r$. So, this case cannot occur.
   - $T' \equiv T^{*'}$, $t_1^* < t_1$ and $r^* \neq r$: This means that all timestamps are correctly alternating, and the two sessions differ only in the first message, i.e., $r \neq r^*$. This is the case where $\mathcal{B}$ changes its way of answering: it does not forward the last message to its oracle R, and returns $\perp$ to $\mathcal{A}$. We argue that $\mathcal{B}$'s answer is correct. Indeed, since $T' \equiv T^{*'}$, by correctness of Π, R's private output will be a message of the form $r^*|m_b$. However, since $r^* \neq r$ R' would reject this message

   At the end, $\mathcal{B}$ returns the same output of $\mathcal{A}$.
2. S **speaks first.** $\mathcal{B}$ is run on input sendk. It runs $\mathcal{A}(\mathsf{sendk})$ and answers oracle queries as follows. Since in Π' it is R' who speaks first, when $\mathcal{A}$ will query R' on a new session, $\mathcal{A}$ does not send any message, and $\mathcal{B}$ simulates the answer by returning a randomly chosen string $r \in \{0,1\}^\lambda$. $\mathcal{B}$ stores r,

and then forward all subsequent queries on this session to its oracle R, forwarding the corresponding answers to $\mathcal{A}$. The only change is in simulating answers to the last message. $\mathcal{B}$ forwards the message received from $\mathcal{A}$ to R. Let $(m'|r')$ be its response, and let r be the string stored for this session by $\mathcal{B}$. If $r = r'$ then $\mathcal{B}$ returns m'. Otherwise, it returns $\perp$.
When $\mathcal{A}$ outputs a message pair (m_0, m_1) — recall that $\mathcal{A}$ (who is playing the role of the receiver) is supposed to speak first in the challenge session — $\mathcal{B}$ waits for the message $r^* \in \{0,1\}^\lambda$ from $\mathcal{A}$. $\mathcal{B}$ stores r^* and starts forwarding all messages to its own challenge session which is initialized by choosing $(r^*|m_0, r^*|m_1)$ as the challenge message pair.
In this second phase all oracle queries are answered as before, except for the following changes. First, in every query from $\mathcal{A}$ to R' to open a new session, $\mathcal{B}$ chooses the nonce $r \overset{\$}{\leftarrow} \{0,1\}^\lambda \backslash \{r^*\}$. It is easy to see that the distribution of this choice of r is statistically close to the one in the real experiment. Second, assume that the challenge session is completed, let $T^* = \langle (r^*, t_1^*), (M_1^*, t_2^*), \ldots, (M_n^*, t_{n+1}^*)\rangle$ be its transcript, and assume that $\mathcal{A}$ queries R' on an already opened session by sending the last message. $\mathcal{B}$ should then answer with the (simulated) output of R'. Let $T = \langle (r, t_1), (M_1, t_2), \ldots, (M_n, t_{n+1})\rangle$ be the transcript of the queried session. Since $\mathcal{A}$ is not ping-pong, $T \not\equiv T^*$, and we would like to argue that $\mathcal{B}$ must not be ping-pong as well while still being capable to answer queries correctly.
Let us write $T^* = (r^*, t_1^*), T^{*'}$ and $T = (r, t_1), T'$ (i.e., we explicitly separate the first message from the transcript of the n-round protocol). If $T \not\equiv T^*$, either one of the following cases holds:

- $T' \not\equiv T^{*'}$: $\mathcal{B}$ is clearly not ping-pong and it can answer forwarding this query to its oracle R.
- $T' \equiv T^{*'}$, $t_1 > t_1^*$ and $r = r^*$: $t_1 > t_1^*$ means that r was generated by $\mathcal{B}$ after r^* has been sent by $\mathcal{A}$. However, by construction of $\mathcal{B}$ (see above) we always have $r^* \neq r$. So this case cannot occur.
- $T' \equiv T^{*'}$, $t_1 < t_1^*$ and $r^* \neq r$: This means that all timestamps are correctly alternating, and the two sessions differ only in the first message, i.e., $r \neq r^*$. In this case $\mathcal{B}$ outputs $\perp$. By the same reasons in case (1) of the proof, this answer is correct.

At the end, $\mathcal{B}$ returns the same output of $\mathcal{A}$. □

F.3 Proof of Theorem 3 (iCMA security)

Assume by contradiction there exists a PPT adversary $\mathcal{A}$ that breaks the iCMA security of Π', then we show how to build a PPT algorithm $\mathcal{B}$ that uses $\mathcal{A}$ as a subroutine in order to break the iCMA security of Π. We distinguish between the following two cases according to which party speaks first.

1. R **speaks first.** $\mathcal{B}$ is run on input recvk. It runs $\mathcal{A}(\mathsf{recvk})$ and answers oracle queries as follows. Since in Π' it is S' who speaks first, when $\mathcal{A}$ queries S' on a new session (providing a plaintext m), $\mathcal{B}$ simulates the first message from S' by returning a random string $r \in \{0,1\}^\lambda$, which is stored by $\mathcal{B}$. All subsequent queries on this session are forwarded from $\mathcal{B}$ to its oracle S (initialized with the plaintext $m|r$).
When $\mathcal{A}$ starts the challenge session, $\mathcal{A}$ must start speaking by sending a string r^*. $\mathcal{B}$ stores r^*, starts its challenge session and forwards all messages to and from $\mathcal{A}$. After the challenge session has started, $\mathcal{B}$ chooses the nonces $r \overset{\$}{\leftarrow} \{0,1\}^\lambda \setminus \{r^*\}$. It is easy to see that the distribution of this choice of r is statistically close to the one in the real experiment.
Observe that by construction of R', if $\mathcal{A}$ makes $\mathsf{R}'(\mathsf{recvk})$ accept and return a message m^* in the simulated challenge session with $\mathcal{B}$, then $\mathcal{B}$ (who forwards the very same queries) will make $\mathsf{R}(\mathsf{recvk})$ accept with message $r^*|m^*$ in its challenge session. Therefore, to complete the proof, it is left to show that if $\mathcal{A}$ is not ping-pong, the same holds for $\mathcal{B}$. Namely, let $T^* = \langle (r^*, t_1^*), (M_1^*, t_2^*), \ldots, (M_n^*, t_{n+1}^*)\rangle$ be the transcript of the challenge session between $\mathcal{B}$ (simulating $\mathsf{R}'(\mathsf{recvk})$) and $\mathcal{A}$, and let $T = \langle (r, t_1), (M_1, t_2), \ldots, (M_n, t_{n+1})\rangle$ be the transcript of the any session between $\mathcal{A}$ and its oracle S' (simulated by $\mathcal{B}$). Since $\mathcal{A}$ is not ping-pong, $T \not\equiv T^*$. Let us write $T^* = (r^*, t_1^*), T^{*'}$ and $T = (r, t_1), T'$ (i.e., we explicitly separate the first message from the transcript of the n-round protocol). If $T \not\equiv T^*$, either one of the following cases holds:

- $T' \not\equiv T^{*'}$: $\mathcal{B}$ is also not ping-pong in its game.
- $T' \equiv T^{*'}$, $t_1^* > t_1$ and $r = r^*$: $t_1^* > t_1$ means that r^* was generated after r has been sent from $\mathcal{A}$. However, by construction of the simulation we only have $r \neq r^*$, and thus this case cannot occur.
- $T' \equiv T^{*'}$, $t_1^* < t_1$ and $r^* \neq r$: This means that all timestamps are correctly alternating, and the two sessions differ only in the first message, i.e., $r \neq r^*$. Since $T' \equiv T^{*'}$, by correctness of $\mathsf{\Pi}$, R would accept in the challenge session with a message of the form $m^*|r^*$, but this implies that R' rejects this message as $r^* \neq r$.

2. S **speaks first.** $\mathcal{B}$ is run on input recvk. It runs $\mathcal{A}(\mathsf{recvk})$ and answer oracle queries as follows. Since in $\mathsf{\Pi}'$ it is R' who speaks first, when $\mathcal{A}$ will query S' on a new session, $\mathcal{A}$ has to provide a string r (in addition to the chosen plaintext m). $\mathcal{B}$ then queries its oracle S with plaintext $m|r$ and starts forwarding all subsequent queries of this session to $\mathcal{A}$.

   When $\mathcal{A}$ starts the challenge session, $\mathcal{B}$ chooses a random string $r^* \xleftarrow{\$} \{0,1\}^\lambda \setminus \{r_1, \ldots, r_Q\}$ (where $r_1, \ldots, r_Q$ are all the strings sent by $\mathcal{A}$ to the oracle R' in the previous phase), and sends r^* to $\mathcal{A}$. Note that as long as the r string is sufficiently large, the distribution of this choice of r^* is statistically close to the one in the real experiment. Next, in the challenge session $\mathcal{B}$ simply relay all messages between $\mathcal{A}$ and its challenger.

   Observe that by construction of R', if $\mathcal{A}$ would make $\mathsf{R}'(\mathsf{recvk})$ accept and return a message m^* in the simulated challenge session with $\mathcal{B}$, then $\mathcal{B}$ (who forwards the very same messages) can make $\mathsf{R}(\mathsf{recvk})$ accept with message $m^*|r^*$ in its challenge session. Therefore, to complete the proof, it is left to show that if $\mathcal{A}$ is not ping-pong, the same holds for $\mathcal{B}$. Namely, let $T^* = \langle (r^*, t_1^*), (M_1^*, t_2^*), \ldots, (M_n^*, t_{n+1}^*) \rangle$ be the transcript of the challenge session between $\mathcal{B}$ (simulating $\mathsf{R}'(\mathsf{recvk})$) and $\mathcal{A}$, and let $T = \langle (r, t_1), (M_1, t_2), \ldots, (M_n, t_{n+1}) \rangle$ be the transcript of any session between $\mathcal{A}$ and its oracle S' (simulated by $\mathcal{B}$). Since $\mathcal{A}$ is not ping-pong, $T \not\equiv T^*$. Let us write $T^* = (r^*, t_1^*), T^{*'}$ and $T = (r, t_1), T'$ (i.e., we explicitly separate the first message from the transcript of the n-round protocol). If $T \not\equiv T^*$, either one of the following cases holds:
   - $T' \not\equiv T^{*'}$: $\mathcal{B}$ is clearly not ping-pong in its game.
   - $T' \equiv T^{*'}$, $t_1 > t_1^*$ and $r = r^*$: $t_1 > t_1^*$ means that r was generated after r^* has been sent from $\mathcal{A}$. However, by construction of the simulation we only have $r \neq r^*$, and thus this case cannot occur.
   - $T' \equiv T^{*'}$, $t_1 < t_1^*$ and $r^* \neq r$: This means that all timestamps are correctly alternating, and the two sessions differ only in the first message, i.e., $r \neq r^*$. Since $T' \equiv T^{*'}$, by correctness of $\mathsf{\Pi}$, R would accept in the challenge session with a message of the form $m^*|r^*$, but this implies that R' rejects this message as $r^* \neq r$. □