Robust Encryption, Revisited

Pooya Farshim[1], Benoît Libert[2], Kenneth G. Paterson[3], and Elizabeth A. Quaglia[4]

[1] Fachbereich Informatik, Technische Universität Darmstadt, Germany
[2] Technicolor, France
[3] Information Security Group, Royal Holloway, University of London, UK
[4] Département d'Informatique, École Normale Supérieure – Paris, France

Abstract. We revisit the notions of robustness introduced by Abdalla, Bellare, and Neven (TCC 2010). One of the main motivations for the introduction of strong robustness for public-key encryption (PKE) by Abdalla et al. to prevent certain types of attack on Sako's auction protocol. We show, perhaps surprisingly, that Sako's protocol is still vulnerable to attacks exploiting robustness problems in the underlying PKE scheme, even when it is instantiated with a *strongly* robust scheme. This demonstrates that current notions of robustness are insufficient even for one of its most natural applications. To address this and other limitations in existing notions, we introduce a series of new robustness notions for PKE and explore their relationships. In particular, we introduce *complete* robustness, our strongest new notion of robustness, and give a number of constructions for completely robust PKE schemes.

Keywords. Robustness, Anonymity, Public-key encryption, Security proofs.

1 Introduction

A commonly pursued goal in cryptography is message privacy, which is typically achieved by means of encryption. In recent years, the privacy of users has become an equally relevant concern. It has led the research community to strive for anonymity properties when designing cryptographic primitives. In public-key encryption, in particular, *key-privacy* (a.k.a. receiver anonymity) was introduced in [4] to capture the idea that a ciphertext does not leak any information about the public key under which it was created, thereby making the communication anonymous. In this context, Abdalla, Bellare, and Neven [2] raised a fundamental question: how does a legitimate user know if an anonymous ciphertext is intended for him? Moreover, what happens if he uses his secret key on a ciphertext *not* created under his public key? To address this question, Abdalla et al. formalized a property called *robustness*, which (informally speaking) guarantees that decryption attempts fail with high probability if the "wrong" private key is used. They argued that, in all applications requiring anonymous public-key encryption, robustness is usually needed as well. These applications include auction protocols with bid privacy [28], consistency [1] in searchable encryption [8] and anonymous broadcast encryption [3,24]. As shown by Mohassel [25], robustness is also important in guaranteeing the anonymity of hybrid encryption schemes resulting from the combination of anonymous asymmetric and symmetric components.

1.1 Robust public-key encryption

Robustness ensures that a ciphertext cannot correctly decrypt under two different secret keys. This notion has (often implicitly) been present in the literature (e.g., [28,8,10,22,3]), but formal definitions remained lacking until the recent foundational work of Abdalla et al. [2]. In particular, Abdalla et al. introduced two flavors of encryption robustness: *weak* and *strong* robustness.

Weak robustness is modeled as a game in which a winning adversary outputs a valid message M and two distinct public keys pk_0 and pk_1 such that the encryption of M under pk_0 decrypts to a valid message under sk_1, the secret key corresponding to pk_1. This notion is of interest since it precisely addresses the issue of *using the wrong key* that arises in anonymity contexts (such as anonymous broadcast encryption [3,24], for instance), but it is also useful in achieving the stronger notion of strong robustness.

Strong robustness—also called SROB-CCA when the adversary has access to a decryption oracle—allows for a more powerful adversary which chooses a ciphertext C (as opposed to a message which will be honestly encrypted) and two distinct public keys, and wins if C decrypts to a valid message under both corresponding secret keys. In [2] the need for this notion is motivated by scenarios where ciphertexts can be adversarially chosen. The authors of [2] give Sako's auction protocol [28] as an example of such a situation, explaining that strong robustness is required in order to prevent an attack on the fairness of this protocol by a cheating bidder and a colluding auctioneer.

As pointed out by Abdalla et al. [2], merely appending the receiver's public key to the ciphertext is not an option for providing robustness, since it destroys key-privacy properties. Abdalla et al. also showed that the seemingly natural solution of using an unkeyed redundancy function to modify the message before encryption does not achieve even weak robustness, thus demonstrating the non-triviality of the problem. They then gave several anonymity-preserving constructions to obtain both weak and strong robustness for public-key encryption. Using a simple tweak, they also showed how to render the Cramer–Shoup cryptosystem [13] strongly robust without introducing any overhead.

More recently, Mohassel [25] studied robustness in the context of hybrid encryption [14]. He showed that weak robustness (and not only anonymity) is needed in the asymmetric part of a hybrid encryption scheme to ensure anonymity of the overall scheme. Mohassel also considered relaxations, called *collision-freeness*, of both weak and strong robustness. He showed that many constructions in the literature are natively collision-free and showed how to generically turn any weakly (resp., strongly) collision-free scheme into a weakly (resp., strongly) robust one.

1.2 Our contributions

The need for stronger definitions. In this paper, we argue that some applications require even stronger forms of robustness than those considered in [2,25]. The first such application is, perhaps surprisingly, the construction of auction protocols with bid privacy, like that of Sako [28]. Recall that this was one of the initial motivations for analyzing robustness in [2]. Strong robustness actually turns out *not* to suffice for thwarting attacks against the fairness of Sako's auction protocol [28]: strong robustness assumes honestly generated public keys whereas, if the auctioneer can collude with cheating bidders (as assumed in [2]), what really needs to be considered is an adversary who can maliciously generate ciphertexts *and* the public keys. To illustrate this, we show an attack on the fairness of Sako's protocol when instantiated with $\mathcal{CS}^\star$, a variant of the Cramer–Shoup encryption scheme which was proven to be key-private and strongly robust in [2]. This observation, then, motivates us to introduce notions of robustness where keys may be maliciously generated. We do not offer a full treatment of the delicate issue of fairness in auction protocols and its relation to robustness, since that is beyond the scope of this paper. Rather, as with [2], we use Sako's protocol as a motivation for introducing and studying stronger robustness notions.

The limitations of existing robustness notions, and therefore the motivation for this work, are not solely restricted to Sako's protocol. For instance, existing notions are not necessarily strong enough to provide robustness guarantees if the scheme is used to encrypt *key-dependent* messages [7] or messages encrypted under *related keys* [6]. This is because the adversary is denied access to the secret keys in these notions. The strongest of our new notions gives the adversary sufficient power and automatically provides robustness in these more challenging settings.

New notions of robustness and their relations. Our strongest new notion is called *complete* robustness (CROB) and is obtained by progressively removing various restrictions on adversarial capabilities in the strong robustness security model. First, we give access to honestly generated secret keys and arrive at an intermediate notion which we term *unrestricted (strong) robustness* (USROB). Next, we also remove the honest key-generation requirement to get to the notion of *full robustness* (or FROB for short). We then

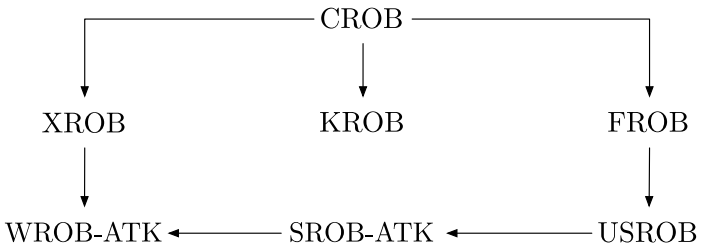

Fig. 1. Relations among notions of robustness.

view robustness in terms of the behavior of the encryption and decryption algorithms with respect to each other, and obtain our CROB notion. Roughly speaking, in CROB, the adversary should not be able to find "collisions" in the scheme beyond those which are already implied by the correctness property of the scheme. For example, he should not be able to "explain" a ciphertext C of his choice as an encryption under two different adversarially chosen public keys pk_0, pk_1 by revealing the plaintext and the encryption coins for pk_0 and the secret key sk_1 for pk_1. As we will see, full robustness can be viewed as the "decryption-only part" of CROB. Another natural notion of robustness, which we call *key-less robustness* (KROB), arises as the dual notion corresponding to the "encryption-only part" of CROB, and is also implied by CROB. Finally, XROB is a "mixed" notion derived from FROB and KROB that has no natural interpretation but is a useful tool in establishing results about these notions.

We next study how these new notions of robustness relate to each other and to existing notions. Figure 1 summarizes the main relations that we prove between our new and existing robustness notions. In this figure, the lack of an implication between two notions should be interpreted as meaning that we prove a separation. Thus, for example, we will show that CROB is strictly stronger than FROB. It is apparent from the figure that we provide a complete account of the pairwise relations between the various robustness notions. In addition to these relations, we can prove several pairwise separations. For example, we will show that no two of the three notions from $\{\text{FROB}, \text{KROB}, \text{XROB}\}$ are sufficient to prove CROB, but that their combination is. Thus we obtain a characterization of CROB in terms of the three intermediate notions. These separations are not displayed in the figure for ease of visual presentation.

That robustness can come in so many flavors may be unsettling to some readers. Certainly, one should not seek to clutter the definitional landscape unnecessarily. Yet, with the exception of XROB, all of our notions arise as natural generalizations of the existing notions. Exploring their relations is then a natural endeavor. This is not so different from the situation for, say, confidentiality and anonymity notions for public-key encryption, where we now have many different security definitions and developing an understanding of their relations has taken several years.

Constructions of completely robust encryption. Having defined CROB and its weaker relatives, we prove it to be achievable via a variety of efficient and natural constructions.

We first show that the generic construction for strong robustness presented in [2] is *already* powerful enough as to also achieve CROB. Further, we observe that a slight modification of this transformation allows dispensing with the weak robustness assumption—which was necessary in [2]—on the underlying PKE scheme. Moreover, we point out that the random-oracle-based generic transformation of Mohassel [25] also achieves CROB.

In the standard model, we also answer in a positive sense a question left open in [2] as to whether the Canetti–Halevi–Katz [12] (CHK) paradigm—which is known to provide chosen-ciphertext secure cryptosystems from weakly secure identity-based encryption (IBE) schemes—can be leveraged to construct systems that are simultaneously anonymous and offer message privacy under chosen-ciphertext attacks (AI-CCA security) *and* are robust in a strong sense. Answering this question is non-trivial: Abdalla et al.

pinpointed that applying the one-time-signature-based CHK transformation to the Boyen–Waters IBE [11], for example, does not provide SROB-CCA or even SROB-CPA. Here, we show how to obtain AI-CCA-secure, completely robust PKE schemes from weakly secure IBE schemes. Our construction is a variant of the Boneh–Katz construction for chosen-ciphertext security [9], and it only requires the underlying IBE to satisfy a weak level of security under chosen-plaintext attacks. In comparison, the most powerful transformation of [2] must start from a scheme that is already AI-CCA-secure to achieve a comparable result. Because our technique simultaneously provides complete robustness *and* AI-CCA security, it enjoys better efficiency than applying the strongest robustness-conferring transformation of [2] to an AI-CCA-secure scheme obtained from the original Boneh–Katz transformation.

Finally, we also ask whether we can improve upon the efficiency of generic constructions with concrete schemes whose security rests on specific computational assumptions. By giving a concrete construction of a scheme that is CROB and AI-CCA-secure, we present a different and potentially more efficient way of directly achieving CROB for certain hybrid encryption schemes such as the Hofheinz–Kiltz [20] or Kurosawa–Desmedt [23] schemes. To do so, we take advantage of certain properties in the underlying symmetric components. Namely, we consider hybrid schemes that build on the *encrypt-then-MAC* paradigm in their symmetric part to obtain a suitably secure symmetric cipher. We show that, if the message authentication code (MAC) is what we call *committing*, then a simple modification in the hybrid scheme gives complete robustness without any significant computational overhead. The use of committing MACs readily extends as a tool to design AI-CCA-secure CROB hybrid constructions via the KEM/DEM framework [14]. Concretely, Mohassel [25] showed that the KEM/DEM framework gives an AI-CCA-secure hybrid encryption scheme when the KEM component is weakly robust and AI-CCA, and the DEM component is an authenticated symmetric encryption scheme. If the latter part is furthermore realized using the encrypt-then-MAC approach with a committing MAC, we easily obtain complete robustness as well. As we will see, the committing MAC technique can also offer certain advantages.

Taken altogether, our constructions achieving CROB rely on different building blocks and, when fully instantiated, allow us to rely on a variety of different hardness assumptions. They demonstrate that CROB, while providing strong guarantees, is attainable in an efficient and flexible manner.

Organization. We start by reviewing the previous notions of robustness and highlighting their limitations in Section 2. Section 3 presents our new notions of robustness. In Section 4, we study the relations among notions of robustness. We describe our generic constructions in Section 5 and give an efficient construction in Section 6. We close by some concluding remarks in Section 7.

2 Previous Notions of Robustness and Their Limitations

We first briefly recall the existing notions of robustness, namely *weak* and *strong* robustness from [2].

2.1 Weak and strong robustness

Let $\mathcal{PKE} = (\mathsf{PG}, \mathsf{KG}, \mathsf{Enc}, \mathsf{Dec})$ be a public-key encryption scheme consisting of parameter-generation, key-generation, encryption, and decryption algorithms (see Appendix A for the detailed syntax). The authors of [2] distinguish between *weak* robustness, where the adversary has to output a message and two public keys, and *strong* robustness, where it outputs a ciphertext and two public keys. The corresponding games are recalled in Figure 2.

2.2 Strong robustness does not suffice for auction protocols

Sako's auction protocol [28] was the first practical protocol to ensure *bid privacy*, i.e., to hide the bids of losers. The basic idea is as follows. Let $V = \{v_1, ..., v_N\}$ be the set of possible bid values. The auctioneer prepares N key-pairs $(sk_i, pk_i)_{i \in \{1,...,N\}}$ and publishes the N public keys. To *bid* for a value v_i a bidder

proc Initialize
$\mathsf{EK}, \mathsf{DK}, U, V \leftarrow \emptyset$
$pars \leftarrow\$ \mathsf{PG}$
Return $pars$

proc GetEK(id)
$U \leftarrow U \cup \{id\}$
$(\mathsf{DK}[id], \mathsf{EK}[id]) \leftarrow\$ \mathsf{KG}(pars)$
Return $\mathsf{EK}[id]$

proc GetDK(id)
If $id \notin U$ Then Return $\perp$
$V \leftarrow V \cup \{id\}$
Return $\mathsf{DK}[id]$

proc Dec(C, id) // ATK = CCA
If $id \notin U$ Then Return $\perp$
$M \leftarrow \mathsf{Dec}(pars, \mathsf{EK}[id], \mathsf{DK}[id], C)$
Return M

proc Finalize(M, id_0, id_1) // WROB-ATK
If $(id_0 \notin U) \vee (id_1 \notin U)$ Then Return F
If $(id_0 \in V) \vee (id_1 \in V)$ Then Return F
If $(id_0 = id_1)$ Then Return F
$M_0 \leftarrow M$; $C \leftarrow\$ \mathsf{Enc}(pars, \mathsf{EK}[id_0], M_0)$
$M_1 \leftarrow \mathsf{Dec}(pars, \mathsf{EK}[id_1], \mathsf{DK}[id_1], C)$
Return $(M_0 \neq \perp) \wedge (M_1 \neq \perp)$

proc Finalize(C, id_0, id_1) // SROB-ATK
If $(id_0 \notin U) \vee (id_1 \notin U)$ Then Return F
If $(id_0 \in V) \vee (id_1 \in V)$ Then Return F
If $(id_0 = id_1)$ Then Return F
$M_0 \leftarrow \mathsf{Dec}(pars, \mathsf{EK}[id_0], \mathsf{DK}[id_0], C)$
$M_1 \leftarrow \mathsf{Dec}(pars, \mathsf{EK}[id_1], \mathsf{DK}[id_1], C)$
Return $(M_0 \neq \perp) \wedge (M_1 \neq \perp)$

Fig. 2. Games defining weak/strong robustness.

encrypts a pre-determined message M under the public key pk_i. This is signed and posted by the bidder. To *open* a bid the auctioneer attempts to decrypt the encrypted bids one by one using sk_N. If at least one decrypts to M, the auctioneer publishes the winning bid v_N, a list of all the winning bidders and the secret key sk_N for the bidders to verify correctness of the result. If no decryption returns M, the auctioneer repeats the procedure using sk_{N-1}, and so on. For the auction to hide the bid values, the underlying public-key encryption scheme needs to be key-private, in the sense of [4].

In [28], Sako provided an example of an auction protocol scheme based on the ElGamal public-key encryption scheme, which is key-private. In [2], Abdalla et al. gave an attack which allows a cheating bidder and a colluding auctioneer to break the fairness of the protocol. This attack is based on the fact that the ElGamal scheme is not robust and therefore the auctioneer can open the cheating bidder's bid to an arbitrary (winning) value. To prevent this attack, the authors of [2] suggest using any *strongly* robust scheme (strong robustness, instead of simply weak robustness, is required since the ciphertexts are generated adversarially).

We show that strong robustness is *not* sufficient to prevent an attack of the above type on Sako's protocol. More precisely, we present an attack on the protocol when it is instantiated with a variant of the Cramer–Shoup encryption scheme, $\mathcal{CS}^\star$, which is known to be key-private and strongly robust (the latter result was proved in [2]). Just as with the attack of Abdalla et al. [2], the attack we present assumes a cheating bidder and a colluding auctioneer. The key idea behind the attack (which is presented in detail in Appendix C) is that an auctioneer can maliciously prepare the public keys so that the cheating bidder's encryption decrypts to M under *any* secret key.

This attack shows that strong robustness is not enough to guarantee fairness in Sako's auction protocol. Intuitively what is needed here is a form of robustness wherein all the public keys and ciphertexts in the system may be adversarially generated. In the coming sections we will formalize stronger notions of robustness which rule out such attacks.

3 New Notions of Robustness

3.1 A direct strengthening: full robustness

Recall that an SROB adversary has to output a ciphertext C and two public keys pk_0 and pk_1 such that C decrypts to a message M_0 under (sk_0, pk_0) and a message M_1 under (sk_1, pk_1). The notion poses three restrictions on the adversary: (1) pk_0 and pk_1 have to be distinct; (2) The corresponding secret keys cannot have been queried by the adversary; (3) The public keys are honestly generated.

The first condition is *inherent* to modeling the behavior of an encryption scheme when used on different public keys, and removing it would make it trivial for an adversary to win.

We now look at the notion resulting from the removal of the second restriction, i.e., when the adversary is allowed to query secret keys even for the finally output public keys. We call this notion unrestricted strong robustness (USROB). This game therefore proceeds as the SROB game does except that the check $(id_0 \in V) \vee (id_1 \in V)$ is no longer present in the **Finalize** procedure. This notion is powerful enough to model scenarios where keys are honestly generated, but an adversary may know the secret keys. This, for example, includes robustness for the encryption of key-dependent messages as discussed in the introduction.

However, as we have seen in the previous section, if an adversary can control the generation of keys, it may be unreasonable to assume that it can only generate the keys honestly. We therefore can strengthen USROB further by removing the third restriction on the adversary. We, however, ask the adversary to return secret keys for the public keys that it chooses. Two points deserve further attention at this point. First, returning the secret keys is to allow for a polynomial-time game definition which is not excessively strong. Second, we do not require the secret keys to be valid. Indeed, it is the responsibility of the decryption algorithm to check that the key-pair it receives is valid. Note that as a result of removing the two restrictions, the adversary has now full control over the keys, and we no longer need to provide the adversary with the oracles present in the SROB and USROB games. These modifications result in a simple, but strong, notion we call *full robustness* (FROB), and formalize in Figure 3.

proc Initialize	**proc Finalize**$(C, pk_0, pk_1, sk_0, sk_1)$ // FROB
$pars \leftarrow\$ \mathsf{PG}$	If $(pk_0 = pk_1)$ Then Return F
Return $pars$	$M_0 \leftarrow \mathsf{Dec}(pars, pk_0, sk_0, C)$
	$M_1 \leftarrow \mathsf{Dec}(pars, pk_1, sk_1, C)$
	Return $(M_0 \neq \perp) \wedge (M_1 \neq \perp)$

Fig. 3. Game defining full robustness.

3.2 A unified approach: complete robustness

At this point it can be asked if there are attacks which fall outside the FROB model. To answer this question, we take a somewhat different approach towards robustness and view it in terms of the behavior of the encryption and decryption routines of a scheme with respect to each other. In fact, this is the underlying intuition behind not only the original weak robustness notion,[5] but also the standard correctness criterion for a PKE scheme (albeit for a single key). This leads us to a new notion which we term *complete robustness* (CROB). In this game the shared parameters of the system are passed to an adversary, which then arbitrarily interacts with the encryption and decryption routines on plaintexts, ciphertexts, keys, and even random coins of its choice. Its goal is to find an "unexpected collision" in the cryptosystem (i.e., one outside that imposed by the correctness criterion). We formalize the CROB game in Figure 4.

Key-less robustness. It can be seen through an easy inspection that full robustness is a sub-case of complete robustness where the adversary is restricted to querying the **Dec** oracle. One can also consider the dual case where the adversary only queries the **Enc** oracle. This results in a new notion which we call *key-less robustness* (KROB). Key-less robustness differs from full robustness in that an adversary no longer needs to return any secret keys, but instead "opens" a ciphertext by providing the random coins and the message used in the encryption. More precisely, the adversary outputs two messages, two distinct public keys and two sets of random coins, and its goal is to invoke a collision in the encryption algorithm. The game is shown in Figure 5.

[5] This then disappears in the SROB game as the adversary outputs ciphertexts.

proc Initialize
$\mathsf{List} \leftarrow [\,]$
$pars \leftarrow\!\!\$\ \mathsf{PG}$
Return $pars$

proc Enc(pk, M, r)
$C \leftarrow \mathsf{Enc}(pars, pk, M; r)$
$\mathsf{List} \leftarrow (pk, M, C) \cup \mathsf{List}$

proc Dec(pk, sk, C)
$M \leftarrow \mathsf{Dec}(pars, pk, sk, C)$
$\mathsf{List} \leftarrow (pk, M, C) \cup \mathsf{List}$

proc Finalize() // CROB
For each pair $(pk_0, M_0, C_0), (pk_1, M_1, C_1) \in \mathsf{List}$
  If $(C_0 = C_1 \neq \bot) \wedge (pk_0 \neq pk_1) \wedge (M_0 \neq \bot \wedge M_1 \neq \bot)$ Return T
Return F

Fig. 4. Game defining complete robustness.

proc Initialize
$pars \leftarrow\!\!\$\ \mathsf{PG}$
Return $pars$

proc Finalize$(M_0, M_1, pk_0, pk_1, r_0, r_1)$ // KROB
If $(pk_0 = pk_1)$ Then Return F
$C_0 \leftarrow \mathsf{Enc}(pars, M_0, pk_0; r_0)$
$C_1 \leftarrow \mathsf{Enc}(pars, M_1, pk_1; r_1)$
Return $(C_0 = C_1 \neq \bot)$

Fig. 5. Game defining key-less robustness.

Intuitively this notion appears to be the strongest amongst the ones considered so far, since the adversary has the liberty to choose the public keys and does not have to reveal any secret information. Surprisingly, we will see that key-less robustness does not imply *any* other of the other notions considered so far (FROB, SROB, and not even weak robustness). Furthermore, we will show that FROB does not imply KROB either. In the next section we give a complete treatment of relations among different notions.

Identity-based encryption. In the IBE setting the identities (analogous to public keys in the PKE setting) are already chosen maliciously, while the natural extension of our notions would allow the adversary to also choose the IBE master keys maliciously. In particular, the identity-based analogue of FROB would be strong enough to guarantee *well-addressedness* according to the definition proposed by Hofheinz and Weinreb [21] (see also Figure 9 in Appendix B), whereas SROB-CCA may not always do so. We leave the further development of the ID-based setting to future work.

4 Relations among Notions of Robustness

We now study how the various notions of robustness relate to each other. Starting with complete robustness, it may be asked if KROB and FROB are strong enough together to jointly imply CROB. We show that this is *not* the case. Indeed, there is a third "mixed" notion of robustness implicit in CROB, which we term XROB and formalize in Figure 6. As the next theorem shows, the XROB notion is necessary in the sense that it is not implied by KROB and FROB together.

In fact, no pair of the notions from $\{\text{FROB}, \text{KROB}, \text{XROB}\}$ implies the third. Furthermore, the conjunction of all three notions is sufficient to imply CROB.

Theorem 1 (CROB **Characterization).** *A PKE scheme is* CROB *if and only if it is simultaneously* FROB*,* KROB*, and* XROB*. Furthermore, no combination of at most two of* FROB*,* KROB*, and* XROB *is sufficient to provide the* CROB *guarantees.*

We prove the theorem via a sequence of propositions in Appendix E. To give a flavor of our results we present one of these next.

Proposition 1 (FROB $\wedge$ KROB $\not\Rightarrow$ XROB**).** *Let $\mathcal{PKE}$ be a public-key encryption scheme which is* FROB *and* KROB*. Then there is a scheme $\mathcal{PKE}'$ which is* FROB *and* KROB*, but fails to be* XROB*.*

Proof. We define the required scheme $\mathcal{PKE}'$ as follows.

$\mathsf{PG}'(1^\lambda)$**:** Run $\mathsf{PG}(1^\lambda)$ to obtain $pars$. Return $pars$.

proc Initialize	**proc Finalize**$(M_0, pk_0, r_0, C_1, pk_1, sk_1)$ // XROB
$pars \leftarrow\!\!\$\ \mathsf{PG}$	If $(pk_0 = pk_1)$ Then Return F
Return $pars$	$C_0 \leftarrow \mathsf{Enc}(pars, M_0, pk_0; r_0)$
	$M_1 \leftarrow \mathsf{Dec}(pars, pk_1, sk_1, C_1)$
	Return $(C_0 = C_1) \wedge (M_0 \neq \perp) \wedge (M_1 \neq \perp)$

Fig. 6. Game defining mixed robustness.

$\mathsf{KG}'(pars)$**:** Run $\mathsf{KG}(pars)$ to obtain (sk, pk). Return $(sk, 0\|pk)$.
$\mathsf{Enc}'(pars, b\|pk, M; r)$**:** Run $\mathsf{Enc}(pars, pk, M; r)$ to obtain C. Output $b\|C$.
$\mathsf{Dec}'(pars, b\|pk, sk, c\|C)$**:** If $b = 1$ return $\perp$. If $c = 1$ return a fixed (e.g., the lexicographically smallest) message $M^\star$ in the message space for pk. Else return $\mathsf{Dec}(pars, pk, sk, C)$.

Note that $\mathcal{PKE}'$ is a correct public-key encryption scheme. We show $\mathcal{PKE}'$ is not XROB. Consider the XROB adversary $\mathcal{A}$ which obtains $pars$, generates a key-pair $(sk, 0\|pk)$ and a set of coins r for the encryption algorithm, and returns $(M, 1\|pk, r, 1\|C, 0\|pk, sk)$ where M is any valid message and $C := \mathsf{Enc}(pars, pk, M; r)$. This tuple wins the XROB game with probability 1 as $1\|pk \neq 0\|pk$, the output of $\mathsf{Enc}'(pars, 1\|pk, M; r)$ is $1\|C$, and the output of $\mathsf{Dec}'(pars, 0\|pk, sk, 1\|C)$ is $M^\star \neq \perp$.

It is easy to see that $\mathcal{PKE}'$ is still KROB. Indeed, since Enc' attaches the first bit of its input public key to the ciphertext, if a collision in the output of Enc' for two distinct public keys arises, it must be that the public keys have the same starting bit and hence they must be differing in their remaining parts. This then translates into a KROB attack on the underlying scheme $\mathcal{PKE}$.

To see that the modified scheme is also FROB, observe that no adversary can win the FROB game by outputting any public key whose starting bit is a 1 as otherwise the decryption algorithm will reject. Therefore the public keys must start with a 0, and as before, in a successful FROB attack the remaining parts of the public keys must be differing. Now since the leading bit, c, of the ciphertexts does not affect Dec', we also obtain an FROB attack on the starting scheme $\mathcal{PKE}$. □

It is natural to study how our new notions relate to the existing notions from Abdalla et al. [2]. Since USROB is a natural intermediate notion, for the sake of completeness, we also investigate where it stands in relation to existing notions. We start by observing that FROB $\implies$ USROB $\implies$ SROB-CCA as the adversary becomes progressively more restricted in each game. Moreover, in the first part of the following theorem, we show that USROB is strictly stronger than SROB-CCA, and that FROB is strictly stronger than USROB. In the second part of the theorem we show that KROB does not even imply WROB-CPA, separating this notion from all notions other than complete robustness. Finally, we show that XROB implies WROB-CCA but *not* SROB-CPA. Hence XROB can be seen a strengthened version of weak robustness in a direction orthogonal to strong robustness.

Theorem 2 (Relation with WROB **and** SROB**).** *Let $\mathcal{PKE}$ be a PKE scheme. We have the following.*

- **FROB***: If $\mathcal{PKE}$ is* FROB*, then it is also* USROB*. If $\mathcal{PKE}$ is* USROB *then it is also* SROB-CCA*. Moreover, these implications are strict.*
- **KROB***:* KROB *does not imply* WROB-CPA *and* SROB-CCA *does not imply* KROB*.*
- **XROB***: If $\mathcal{PKE}$ is* XROB*, then it is also* WROB-CCA*. Furthermore,* XROB *does not imply* SROB-CPA *and* SROB-CCA *does not imply* XROB*.*

We prove the theorem in Appendix F. The results of [2] together with Theorems 1 and 2 resolve all the relations between any pair of robustness notions as we have summarized in Figure 1. For example, to see that KROB $\not\Longrightarrow$ FROB, we use the facts that FROB $\implies$ SROB-ATK but KROB $\wedge$ XROB $\not\Longrightarrow$ SROB-ATK. Moreover, although we do not formally prove it here, all our separating examples are designed such that they preserve the AI-ATK security of the underlying PKE schemes. Hence Figure 1 also applies in the presence of AI-ATK security.

5 Generic Constructions of Completely Robust Public-key Encryption

5.1 Mohassel's transformation

Mohassel [25] gives a generic transformation in the random-oracle model that converts an AI-ATK encryption scheme into one which is SROB-CCA without compromising its AI-ATK security. This construction also achieves complete robustness. In this construction, the hash value $H(pk, r, M)$, where r is the randomness used in the encryption, is attached to ciphertexts. This immediately rules out all forms of collisions between ciphertexts, as the hash values are unlikely to collide on two distinct public keys.

5.2 The ABN transformation

In [2, Theorem 4.2] the authors give a generic construction for a scheme $\overline{\mathcal{PKE}}$ which confers strong robustness and preserves the AI-ATK security of the starting scheme $\mathcal{PKE}$, provided that the latter scheme is additionally WROB. We briefly describe how the transformation works, and refer the reader to the original work for the details. At setup, include in *pars* for $\overline{\mathcal{PKE}}$ the parameters of a commitment scheme (see Appendix G for the definitions). When encrypting, commit to the public key, and encrypt the *de-commitment* along with the message. Also include the commitment as a ciphertext component. Decryption checks the commitment/de-commitment pair for consistency and rejects if this is not the case. We strengthen the result of [2], showing that this construction achieves complete robustness:

Theorem 3 (The ABN Transformation Achieves CROB**).** *Let $\mathcal{A}$ be a PPT* CROB *adversary against $\overline{\mathcal{PKE}}$. Then there exist PPT adversaries $\mathcal{B}_1$, $\mathcal{B}_2$, and $\mathcal{B}_3$ against the binding property of $\mathcal{CMT}$ such that*

$$\mathbf{Adv}^{\mathrm{crob}}_{\overline{\mathcal{PKE}}}(\mathcal{A}) \leq \mathbf{Adv}^{\mathrm{bind}}_{\mathcal{CMT}}(\mathcal{B}_1) + \mathbf{Adv}^{\mathrm{bind}}_{\mathcal{CMT}}(\mathcal{B}_2) + \mathbf{Adv}^{\mathrm{bind}}_{\mathcal{CMT}}(\mathcal{B}_3).$$

The proof of this theorem is given in Appendix H, where we show scheme $\overline{\mathcal{PKE}}$ is simultaneously FROB, KROB, and XROB.

5.3 A modification of the ABN transformation

While the original transformation [2] *does* provide AI-ATK and CROB guarantees, the AI-ATK security of the transformed scheme $\overline{\mathcal{PKE}}$ relies on the weak robustness of the underlying encryption scheme $\mathcal{PKE}$ in the case of chosen-ciphertext adversaries (i.e., when ATK = CCA). We show that, if the underlying encryption scheme supports labels [29] (in which case the encryption and decryption algorithms both take an additional public string L as input; see Appendix A), this assumption can be eliminated and we only need $\mathcal{PKE}$ to be AI-ATK-secure.

Although the weak robustness assumption is not too demanding in theory (since any encryption scheme can be made weakly robust by means of a keyed redundancy-based transformation [2]), our construction provides better efficiency in some settings since many AI-CCA encryption schemes, such as the Cramer–Shoup or the Kurosawa–Desmedt scheme, natively support labels.[6]

Our transformation, which relies on a commitment scheme $\mathcal{CMT} = (\mathsf{CPG}, \mathsf{Com}, \mathsf{Ver})$, is as follows.

$\overline{\mathsf{PG}}(1^\lambda)$**:** Run $pars \leftarrow\!\!\$\ \mathsf{PG}(1^\lambda)$ to obtain public parameters *pars* for $\mathcal{PKE}$. Then, generate $cpars \leftarrow\!\!\$\ \mathsf{CPG}(1^\lambda)$ for $\mathcal{CMT}$. Finally, return $(pars, cpars)$.

$\overline{\mathsf{KG}}(pars, cpars)$**:** Compute and return $(sk, pk) \leftarrow\!\!\$\ \mathsf{KG}(pars)$.

$\overline{\mathsf{Enc}}\big((pars, cpars), pk, M\big)$**:** The algorithm proceeds in two steps.

[6] In the worst case, labeled public-key encryption schemes can always be obtained by appending the label to the encrypted plaintext and checking whether the correct label is recovered at decryption.

1. Commit to pk by computing a pair $(com, dec) \leftarrow_{\$} \mathsf{Com}(cpars, pk)$.
2. Encrypt $M\|dec$ under the label $L = com$ by computing $C \leftarrow_{\$} \mathsf{Enc}(pars, pk, M\|dec, L)$.

Return (C, com) as the final ciphertext.

$\overline{\mathsf{Dec}}\big((pars, cpars), pk, sk, (C, com)\big)$**:** The algorithm proceeds in two steps.

1. Compute $M' \leftarrow \mathsf{Dec}\big(pars, pk, sk, (C, com), L\big)$, with $L = com$. Then, parse M' as $M\|dec$ (and return $\perp$ if M' cannot be parsed properly).
2. Return M if $\mathsf{Ver}(cpars, pk, com, dec) = 1$. Else return $\perp$.

Theorem 4, whose proof is in Appendix I, shows that thanks to the use of labels, we do not have to rely on any weaker form of robustness of $\mathcal{PKE}$ when proving the AI-ATK security of $\overline{\mathcal{PKE}}$.

Theorem 4. *If $\mathcal{PKE}$ is* AI-ATK*-secure and $\mathcal{CMT}$ is a hiding commitment, then $\overline{\mathcal{PKE}}$ is* AI-ATK*-secure. More precisely, for any PPT* AI-ATK *adversary $\mathcal{A}$ against $\overline{\mathcal{PKE}}$, there exists a PPT* AI-ATK *adversary $\mathcal{B}_1$ against $\mathcal{PKE}$ and a PPT distinguisher $\mathcal{B}_2$ against $\mathcal{CMT}$ such that*

$$\mathbf{Adv}^{\text{ai-atk}}_{\overline{\mathcal{PKE}}}(\mathcal{A}) \leq 2 \cdot \mathbf{Adv}^{\text{ai-atk}}_{\mathcal{PKE}}(\mathcal{B}_1) + \mathbf{Adv}^{\text{hide}}_{\mathcal{CMT}}(\mathcal{B}_2).$$

Furthermore, the above construction is CROB *if $\mathcal{CMT}$ is a binding commitment. More precisely, for any PPT* CROB *adversary $\mathcal{A}$, there exists a PPT adversary $\mathcal{B}$ against the binding property of the commitment scheme such that*

$$\mathbf{Adv}^{\text{crob}}_{\overline{\mathcal{PKE}}}(\mathcal{A}) \leq \mathbf{Adv}^{\text{bind}}_{\mathcal{CMT}}(\mathcal{B}).$$

5.4 Completely robust AI-CCA-secure PKE from selectively secure IBE

Next, we present a modification of the Boneh–Katz approach [9] which provides both CROB and AI-CCA security when applied to any IBE scheme that only provides TA anonymity in the multi-authority selective-ID setting (or sID-TAA-CPA security, as defined in Appendix J). In particular, this positively answers the question of whether CHK-like techniques can be used to achieve a strong flavor of robustness from weakly secure IBE.

Let $\mathcal{IBE}$ be an sID-TAA-CPA secure IBE scheme. We obtain a completely robust AI-CCA-secure public-key encryption scheme $\overline{\mathcal{PKE}}$ by combining $\mathcal{IBE}$ with a strongly secure message authentication code $\mathcal{MAC}$ and a trapdoor commitment scheme $\mathcal{TCMT}$.

Recall that a trapdoor commitment scheme $\mathcal{TCMT} = (\mathsf{CPG}, \mathsf{Com}, \mathsf{Ver}, \mathsf{Equiv})$ consists of efficient algorithms where $(\mathsf{CPG}, \mathsf{Com}, \mathsf{Ver})$ function as in an ordinary commitment except that CPG outputs public parameters $cpars$ and a trapdoor td. In addition, Equiv allows equivocating a commitment using the trapdoor td: for any two messages m_1, m_2 and any tuple (com, dec_1) produced as $(com, dec_1) \leftarrow_{\$} \mathsf{Com}(cpars, m_1)$, the trapdoor td allows computing $dec_2 \leftarrow_{\$} \mathsf{Equiv}(td, com, m_1, dec_1, m_2)$ such that $\mathsf{Ver}(cpars, com, m_2, dec_2) = 1$. Moreover, (com, dec_2) has the same distribution as $\mathsf{Com}(cpars, m_2)$.

Our IBE-based construction $\overline{\mathcal{PKE}} = (\overline{\mathsf{PG}}, \overline{\mathsf{KG}}, \overline{\mathsf{Enc}}, \overline{\mathsf{Dec}})$ is as follows.

$\overline{\mathsf{PG}}(1^\lambda)$**:** Run $pars \leftarrow_{\$} \mathcal{IBE}.\mathsf{PG}(1^\lambda)$ to obtain common public parameters $pars$. Also run $cpars \leftarrow_{\$} \mathsf{CPG}(1^\lambda)$ to obtain public parameters for a trapdoor commitment scheme $\mathcal{TCMT}$. Then, choose a message authentication code $\mathcal{MAC}$ with key length $\ell \in \text{poly}(\lambda)$. Finally, return $(pars, cpars, \mathcal{MAC})$.

$\overline{\mathsf{KG}}(pars, cpars, \mathcal{MAC})$**:** Generate a master key pair $(msk, mpk) \leftarrow_{\$} \mathcal{IBE}.\mathsf{MPG}(pars)$ for $\mathcal{IBE}$. Return the key pair $(sk, pk) := (msk, mpk)$.

$\overline{\mathsf{Enc}}\big((pars, cpars, \mathcal{MAC}), pk, M\big)$**:** To encrypt M under $pk = mpk$, the algorithm proceeds as follows.

1. Choose a random MAC key $k \leftarrow_{\$} \{0,1\}^\ell$.
2. Commit to $mpk\|k$ by computing a pair $(com, dec) \leftarrow_{\$} \mathsf{Com}(cpars, mpk\|k)$.
3. Encrypt $M\|k\|dec$ under the identity com by computing $C \leftarrow_{\$} \mathcal{IBE}.\mathsf{Enc}(pars, mpk, com, M\|k\|dec)$.

4. Compute $tag = \mathsf{MacGen}_k(C)$ and return (C, com, tag) as the final ciphertext.

$\overline{\mathsf{Dec}}\big((pars, cpars, \mathcal{MAC}), pk, sk, (C, com, tag)\big)$**:** Given $pk = mpk$ and $sk = msk$, conduct the following steps.

1. Compute $dk_{com} \leftarrow\!\!\$\ \mathcal{IBE}.\mathsf{KG}(pars, msk, com)$ and then $M' \leftarrow \mathcal{IBE}.\mathsf{Dec}\big(pars, mpk, dk_{com}, com, C\big)$. Then, parse M' as $M\|k\|dec$ (and return $\perp$ if $M' = \perp$ or if M' cannot be parsed properly).
2. If $\mathsf{MacVer}_k\big(C, tag\big) = 1$ and $\mathsf{Ver}(cpars, mpk\|k, com, dec) = 1$, return M. Otherwise, return $\perp$.

A difference with the original Boneh–Katz construction—which can use a weak form of commitment called *encapsulation*—is that our construction requires a full-fledged commitment scheme. This is because, in order to achieve complete robustness, we need to commit to the master public key of the scheme at the same time as the MAC key in the encryption algorithm. Moreover, the proof of AI-CCA security requires the commitment to be a *trapdoor* commitment: the trapdoor plays an essential role when it comes to reduce the sID-TAA-CPA security of the IBE to the AI-CCA security of the encryption scheme.

The proof of the following theorem can be found in Appendix J.

Theorem 5. *If $\mathcal{IBE}$ is* sID-TAA-CPA*-secure, $\mathcal{MAC}$ is strongly unforgeable, and $\mathcal{TCMT}$ is a computationally binding trapdoor commitment scheme, then $\overline{\mathcal{PKE}}$ is* AI-CCA*-secure. Moreover, the scheme $\overline{\mathcal{PKE}}$ is* CROB *if $\mathcal{TCMT}$ is computationally binding.*

6 A Concrete CROB Scheme

In this section, we describe a simple way to achieve complete robustness using hybrid encryption where the symmetric component uses the encrypt-then-MAC approach. To this end, we require the MAC to satisfy a "MAC analogue" of the notion of committing symmetric encryption [17]. Informally this notion requires that a given MAC tag is valid for a single message regardless of the key used.

Committing MAC. We say $\mathcal{MAC} = (\mathsf{MacGen}, \mathsf{MacVer})$ is *committing* if for any message m and any key k, there exists no message-key pair (m', k') such that $m' \neq m$ and $\mathsf{MacVer}_{k'}(m', \mathsf{MacGen}_k(m)) = 1$.

We also need the MAC to computationally hide the message. Note that the following definition is implied by the definition of message-hiding security used in [15, Definition 2.2].

Indistinguishable MAC. We say that a message authentication code $\mathcal{MAC} = (\mathsf{MacGen}, \mathsf{MacVer})$ with key space KSp provides *indistinguishability* if, for any two messages m_0, m_1, it is computationally infeasible to distinguish the distributions $\mathcal{D}_b := \{tag \leftarrow\!\!\$\ \mathsf{MacGen}_k(m_b) : k \leftarrow\!\!\$\ \mathsf{KSp}\}$ for $b \in \{0, 1\}$.

For our purposes, the MAC only has to provide one-time strong unforgeability. Namely, the adversary is allowed to see one pair of the form (m, tag), where $tag = \mathsf{MacGen}_k(m)$, and should not be able to produce a pair (m', tag') such that $(m', tag') \neq (m, tag)$ and $\mathsf{MacVer}_k(m', tag') = 1$.

Using ideas from [17], it is easy to construct a MAC which is simultaneously committing, indistinguishable, and strongly unforgeable. The idea is to use a family of *injective* and *key-binding* pseudorandom functions: for any distinct keys k_1, k_2, the functions $f_{k_1}(\cdot)$ and $f_{k_2}(\cdot)$ have disjoint ranges, i.e., there exist no two pairs $(k_1, x_1), (k_2, x_2)$ such that $k_1 \neq k_2$ and $f_{k_1}(x_1) = f_{k_2}(x_2)$. The key space of the MAC is that of the PRF. For any message $m \neq 1^\lambda$, the MAC generation computes and outputs the pair $(f_k(1^\lambda), f_k(m))$. The first component serves as a perfectly binding commitment to the key k while the injectivity of $f_k(\cdot)$ guarantees that the MAC is only valid for one message. In addition, its strong unforgeability and indistinguishability properties are both implied by the pseudorandomness of $\{f_k\}_k$ as long as the message space of the MAC, $\mathsf{MSp}^{\mathrm{mac}}$, does not include 1^λ (the proof is straightforward).

We show a simple variant of the Hofheinz–Kiltz (HK) hybrid encryption scheme [20] that provides CROB and AI-CCA security when the underlying authenticated symmetric encryption scheme uses a MAC with the aforementioned properties. Besides providing new ways to achieve robustness, our scheme

comes with the advantage that its computational efficiency is the same as the original HK scheme and in particular it is more efficient than combining HK with a commitment using the ABN transformation.

$\mathsf{PG}(1^\lambda)$**:** Choose a group $\mathbb{G}$ of prime order $p > 2^\lambda$ with $g \leftarrow\!\!\$\ \mathbb{G}$. Also, choose a symmetric encryption scheme (E, D) of key length ℓ_0 and a message authentication code $\mathcal{MAC} = (\mathsf{MacGen}, \mathsf{MacVer})$ of key length ℓ_1. Finally, choose a key-derivation function $\mathsf{KDF} : \mathbb{G} \to \{0,1\}^{\ell_0+\ell_1}$, a target collision-resistant hash function[7] $\mathsf{TCR} : \mathbb{G} \to \mathbb{Z}_p$, and a collision-resistant hash function $H : \{0,1\}^* \to \mathsf{MSp}^{\mathrm{mac}}$, where $\mathsf{MSp}^{\mathrm{mac}}$ is the message space of $\mathcal{MAC}$. The public parameters consist of $pars := \big(\mathbb{G},\ p,\ g,\ (\mathsf{E}, \mathsf{D}),\ \mathcal{MAC},\ \mathsf{TCR},\ \mathsf{KDF},\ H\big)$.

$\mathsf{KG}(pars)$**:** Choose $x, y, z \leftarrow\!\!\$\ \mathbb{Z}_p^*$ and compute $u = g^x$, $v = g^y$, and $h = g^z$. The public key is $pk = \big(u, v, h\big)$ and the private key is $sk = (x, y, z) \in (\mathbb{Z}_p^*)^3$.

$\mathsf{Enc}\big(pars, pk, M\big)$**:** To encrypt M under the public key pk, choose $s \leftarrow\!\!\$\ \mathbb{Z}_p^*$ and compute

$$C_1 = g^s, \qquad C_2 = (u^\tau \cdot v)^s, \qquad C_3 \leftarrow\!\!\$\ \mathsf{E}_{K_0}(M), \qquad tag = \mathsf{MacGen}_{K_1}(H(C_3, u, v, h))$$

where $\tau = \mathsf{TCR}(C_1) \in \mathbb{Z}_p^*$ and $(K_0, K_1) = \mathsf{KDF}(h^s) \in \{0,1\}^{\ell_0+\ell_1}$. Return $C = (C_1, C_2, C_3, tag)$.

$\mathsf{Dec}\big(pars, pk, sk, C\big)$**:** Given $C = (C_1, C_2, C_3, tag)$, return $\perp$ if $C_2 \neq C_1^{\tau \cdot x + y}$, where $\tau = \mathsf{TCR}(C_1)$. Otherwise, compute $(K_0, K_1) = \mathsf{KDF}(C_1^z)$ and then $M \leftarrow \mathsf{D}_{K_0}(C_3)$. Return M if $\mathsf{MacVer}_{K_1}(H(C_3, pk), tag) = 1$. Otherwise, return $\perp$.

The scheme was known to be IND-CCA-secure. We are also able to prove that it provides AI-CCA security, essentially because the ciphertexts can be shown to be indistinguishable from dummy ciphertexts that are statistically independent of the public key, even in the presence of a decryption oracle. Proofs of the following results may be found in Appendix K.

Theorem 6. *The scheme provides* AI-CCA *security assuming that: (1) The DDH assumption holds in* $\mathbb{G}$*; (2)* (E, D) *is a semantically secure symmetric encryption scheme; (3)* KDF *is a secure key-derivation function;*[8] *(4)* $\mathcal{MAC}$ *is a one-time strongly unforgeable MAC and provides indistinguishability; (5)* H *and* TCR *are collision-resistant and target collision-resistant, respectively. Furthermore, the scheme is* CROB *if* H *is collision-resistant and* $\mathcal{MAC}$ *is committing.*

Interestingly, if the construction of Section 5.4 is modified to use a committing MAC, it can be instantiated using any commitment scheme and in particular a perfectly binding commitment or even an encapsulation scheme (as in the original Boneh–Katz construction) also work. In this case, the sender no longer needs to commit to the master public key: (com, dec) is generated by committing to the MAC key only. Instead, the sender computes tag as $tag = \mathsf{MacGen}_k(H(C, mpk))$ using a collision-resistant hash function H. If the MAC is committing, the resulting construction is easily seen to provide complete robustness. It also remains AI-CCA-secure provided the MAC satisfies the notion of indistinguishability.

7 Closing Remarks

Motivated in part by the shortcomings of existing definitions of robustness, we have made a thorough exploration of the landscape of robustness definitions and their relations, and given a suite of flexible and efficient methods for obtaining completely robust AI-CCA-secure public-key encryption schemes. In future work, one could explore the situation in the ID-based setting. Another open question, well beyond the remit of this paper, is to formalize the fairness of auctions and formally prove that our CROB notion is strong enough to ensure this property for Sako's protocol or its variants.

[7] As in [20], this function can be replaced by an injective encoding from $\mathbb{G}$ to $\mathbb{Z}_p$.

[8] The standard KDF security requires that no distinguisher can tell if it is given the output of the KDF for a random input or just a random element in the range of the KDF.

Acknowledgments. The authors would like to thank Mihir Bellare for his valuable comments. Pooya Farshim is supported by grant Fi 940/4-1 of the German Research Foundation (DFG). Part of this work was done while Benoît Libert was an F.R.S.-F.N.R.S. scientific collaborator at the Université catholique de Louvain (Belgium). Kenneth G. Paterson was supported by an EPSRC Leadership Fellowship, EP/H005455/1.

References

1. Michel Abdalla, Mihir Bellare, Dario Catalano, Eike Kiltz, Tadayoshi Kohno, Tanja Lange, John Malone-Lee, Gregory Neven, Pascal Paillier, and Haixia Shi. Searchable encryption revisited: Consistency properties, relation to anonymous IBE, and extensions. *J. Cryptology*, 21(3):350–391, 2008. (Cited on page 1.)
2. Michel Abdalla, Mihir Bellare, and Gregory Neven. Robust encryption. In Daniele Micciancio, editor, *TCC 2010*, volume 5978 of *Lecture Notes in Computer Science*, pages 480–497. Springer, 2010. (Cited on pages 1, 2, 3, 4, 5, 8, 9, 14, 15, 17, and 18.)
3. Adam Barth, Dan Boneh, and Brent Waters. Privacy in encrypted content distribution using private broadcast encryption. In Giovanni Di Crescenzo and Aviel D. Rubin, editors, *Financial Cryptography 2006*, volume 4107 of *Lecture Notes in Computer Science*, pages 52–64. Springer, 2006. (Cited on pages 1 and 15.)
4. Mihir Bellare, Alexandra Boldyreva, Anand Desai, and David Pointcheval. Key-privacy in public-key encryption. In Colin Boyd, editor, *ASIACRYPT 2001*, volume 2248 of *Lecture Notes in Computer Science*, pages 566–582. Springer, 2001. (Cited on pages 1 and 5.)
5. Mihir Bellare, Eike Kiltz, Chris Peikert, and Brent Waters. Identity-based (lossy) trapdoor functions. In *Advances in Cryptology – Eurocrypt 2012*, Lecture Notes in Computer Science. Springer, 2012. (Cited on page 25.)
6. Mihir Bellare and Tadayoshi Kohno. A theoretical treatment of related-key attacks: Rka-prps, rka-prfs, and applications. In Eli Biham, editor, *EUROCRYPT*, volume 2656 of *Lecture Notes in Computer Science*, pages 491–506. Springer, 2003. (Cited on page 2.)
7. John Black, Phillip Rogaway, and Thomas Shrimpton. Encryption-scheme security in the presence of key-dependent messages. In Kaisa Nyberg and Howard M. Heys, editors, *Selected Areas in Cryptography*, volume 2595 of *Lecture Notes in Computer Science*, pages 62–75. Springer, 2002. (Cited on page 2.)
8. Dan Boneh, Giovanni Di Crescenzo, Rafail Ostrovsky, and Giuseppe Persiano. Public key encryption with keyword search. In Ronald Cramer, editor, *EUROCRYPT 2004*, volume 3027 of *Lecture Notes in Computer Science*, pages 506–522. Springer, 2004. (Cited on page 1.)
9. Dan Boneh and Jonathan Katz. Improved efficiency for CCA-secure cryptosystems built using identity-based encryption. In Alfred Menezes, editor, *CT-RSA 2005*, volume 3376 of *Lecture Notes in Computer Science*, pages 87–103. Springer, 2005. (Cited on pages 4, 10, 25, and 26.)
10. Dan Boneh and Brent Waters. Conjunctive, subset, and range queries on encrypted data. In Vaudenay [30], pages 535–554. (Cited on page 1.)
11. Xavier Boyen and Brent Waters. Anonymous hierarchical identity-based encryption (without random oracles). In Cynthia Dwork, editor, *CRYPTO 2006*, volume 4117 of *Lecture Notes in Computer Science*, pages 290–307. Springer, 2006. (Cited on pages 4 and 25.)
12. Ran Canetti, Shai Halevi, and Jonathan Katz. Chosen-ciphertext security from identity-based encryption. In Christian Cachin and Jan Camenisch, editors, *EUROCRYPT 2004*, volume 3027 of *Lecture Notes in Computer Science*, pages 207–222. Springer, 2004. (Cited on page 3.)
13. Ronald Cramer and Victor Shoup. A practical public key cryptosystem provably secure against adaptive chosen ciphertext attack. In Hugo Krawczyk, editor, *CRYPTO 1998*, volume 1462 of *Lecture Notes in Computer Science*, pages 13–25. Springer, 1998. (Cited on page 2.)
14. Ronald Cramer and Victor Shoup. Design and analysis of practical public-key encryption schemes secure against adaptive chosen ciphertext attack. *SIAM Journal of Computing*, 33:167–226, 2003. (Cited on pages 2 and 4.)
15. Yevgeniy Dodis, Eike Kiltz, Krzysztof Pietrzak, and Daniel Wichs. Message authentication, revisited. In *Eurocrypt 2012*, volume 7237 of *Lecture Notes in Computer Science*, pages 355–374. Springer, 2012. (Cited on page 11.)
16. Léo Ducas. Anonymity from asymmetry: New constructions for anonymous HIBE. In *Topics in Cryptology – CT-RSA 2010*, volume 5985 of *Lecture Notes in Computer Science*, pages 148–164. Springer, 2010. (Cited on page 25.)
17. Marc Fischlin. Pseudorandom function tribe ensembles based on one-way permutations: Improvements and applications. In *Eurocrypt'99*, volume 1592 of *Lecture Notes in Computer Science*, pages 432–445. Springer, 1999. (Cited on page 11.)
18. Rosario Gennaro and Victor Shoup. A note on an encryption scheme of Kurosawa and Desmedt. Cryptology ePrint Archive: Report 2004/194, 2004. (Cited on pages 26 and 29.)
19. Craig Gentry. Practical identity-based encryption without random oracles. In Vaudenay [30], pages 445–464. (Cited on page 25.)
20. Dennis Hofheinz and Eike Kiltz. Secure hybrid encryption from weakened key encapsulation. In *Crypto'07*, volume 4622 of *Lecture Notes in Computer Science*, pages 553–571. Springer, 2007. (Cited on pages 4, 11, 12, 29, and 30.)

21. Dennis Hofheinz and Enav Weinreb. Searchable encryption with decryption in the standard model. *IACR Cryptology ePrint Archive*, 2008:423, 2008. (Cited on pages 7 and 15.)
22. Jonathan Katz, Amit Sahai, and Brent Waters. Predicate encryption supporting disjunctions, polynomial equations, and inner products. In Nigel P. Smart, editor, *EUROCRYPT 2008*, volume 4965 of *Lecture Notes in Computer Science*, pages 146–162. Springer, 2008. (Cited on page 1.)
23. Kaoru Kurosawa and Yvo Desmedt. A new paradigm of hybrid encryption scheme. In *Crypto 2004*, volume 3152 of *Lecture Notes in Computer Science*, pages 426–442. Springer, 2004. (Cited on pages 4, 17, and 29.)
24. Benoît Libert, Kenneth G. Paterson, and Elizabeth A. Quaglia. Anonymous broadcast encryption: adaptive security and efficient constructions in the standard model. In *Public Key Cryptography 2012 (PKC 2012)*, Lecture Notes in Computer Science. Springer, 2012. Available from http://eprint.iacr.org/2011/476. (Cited on pages 1, 17, and 18.)
25. Payman Mohassel. A closer look at anonymity and robustness in encryption schemes. In Masayuki Abe, editor, *ASIACRYPT*, volume 6477 of *Lecture Notes in Computer Science*, pages 501–518. Springer, 2010. (Cited on pages 1, 2, 3, 4, 9, and 15.)
26. Kenneth G. Paterson and Sriramkrishnan Srinivasan. Security and anonymity of identity-based encryption with multiple trusted authorities. In Steven D. Galbraith and Kenneth G. Paterson, editors, *Pairing 2008*, volume 5209 of *Lecture Notes in Computer Science*, pages 354–375. Springer, 2008. (Cited on page 25.)
27. Kenneth G. Paterson and Sriramkrishnan Srinivasan. Building key-private public-key encryption schemes. In Colin Boyd and Juan Manuel González Nieto, editors, *ACISP 2009*, volume 5594 of *Lecture Notes in Computer Science*, pages 276–292. Springer, 2009. (Cited on page 25.)
28. Kazue Sako. An auction protocol which hides bids of losers. In Hideki Imai and Yuliang Zheng, editors, *Public Key Cryptography*, volume 1751 of *Lecture Notes in Computer Science*, pages 422–432. Springer, 2000. (Cited on pages 1, 2, 4, 5, and 18.)
29. Victor Shoup. A proposal for an ISO standard for public key encryption (version 2.1). Manuscript, 2001. (Cited on pages 9 and 14.)
30. Serge Vaudenay, editor. *Advances in Cryptology - EUROCRYPT 2006, 25th Annual International Conference on the Theory and Applications of Cryptographic Techniques, St. Petersburg, Russia, May 28 - June 1, 2006, Proceedings*, volume 4004 of *Lecture Notes in Computer Science*. Springer, 2006. (Cited on page 13.)

A (Labeled) PKE Schemes and AI-ATK Security

We recall the syntax of a (labeled) public-key encryption scheme. We chose not to use the syntax of a Generalized Encryption (GE) scheme [2], which simultaneously formalizes PKE and IBE schemes, for two reasons. First, we will be mainly treating the robustness of public-key encryption schemes in this paper. Second, the GE syntax is not flexible enough to allow defining the identity-based analogs of the notions that we present for PKE schemes.

Labeled public-key encryption. A labeled public-key encryption scheme [29] is defined through a four-tuple of algorithms as follows.

1. $\mathsf{PG}(1^\lambda)$: This is the parameter generation algorithm. On input a security parameter $\lambda \in \mathbb{N}$, it outputs a set *pars* of common public parameters shared by all users in the scheme.
2. $\mathsf{KG}(pars)$: This is the key generation algorithm. On input of the public parameters *pars*, it outputs a key-pair (sk, pk). Implicit in pk are descriptions of the message space MSp and the label space LSp.
3. $\mathsf{Enc}(pars, pk, M, L)$: This is the encryption algorithm. On input of the public parameters *pars*, a public key pk, a message M, and a label L, it outputs a ciphertext C or the special error symbol $\perp$.
4. $\mathsf{Dec}(pars, pk, sk, C, L)$: This is the encryption algorithm. On input of the public parameters *pars*, a public key pk, a secret key sk, a ciphertext C, and a label L, it outputs a message M or the special error symbol $\perp$.

A (standard) public-key encryption scheme is a labeled public-key encryption where $\mathsf{LSp} = \{\epsilon\}$.

AI security. We recall the definition of AI-CCA security given in [2] and extend it to the case of encryption schemes with labels. This notion models the usual IND-CCA and anonymity (also known as key-privacy)

of a PKE scheme in a single game. The advantage of an adversary in the AI-ATK game is defined in the usual way:

$$\mathbf{Adv}^{\text{ai-atk}}_{\mathcal{PKE}}(\mathcal{A}) := |2 \cdot \Pr[\text{AI-ATK}^{\mathcal{A}}_{\mathcal{PKE}} \Rightarrow \mathsf{T}] - 1|$$

where game AI-ATK is shown in Figure 7. Note that we omit the security parameter from the games, and deal with concrete security.

proc Initialize
$T \leftarrow \emptyset$; $b \leftarrow\!\!\$\ \{0,1\}$
$pars \leftarrow\!\!\$\ \mathsf{PG}$
$(sk_0, pk_0) \leftarrow\!\!\$\ \mathsf{KG}(pars)$
$(sk_1, pk_1) \leftarrow\!\!\$\ \mathsf{KG}(pars)$
Return $(pars, pk_0, pk_1)$

proc Dec(b, C, L) // ATK = CCA
If $b \notin \{0,1\}$ Then Return $\perp$
If $(C, L) \in T$ Then Return $\perp$
$M \leftarrow \mathsf{Dec}(pars, pk_b, sk_b, C, L)$
Return M

proc LR$(L^\star, M_0^\star, M_1^\star)$
$C^\star \leftarrow\!\!\$\ \mathsf{Enc}(pars, pk, M_b^\star)$
$T \leftarrow T \cup \{(C^\star, L^\star)\}$
Return $C^\star$

proc Finalize(b')
Return $(b' = b)$

Fig. 7. Games defining AI-ATK security for encryption schemes with labels.

Correctness. We call scheme $\mathcal{PKE}$ correct if for any $\lambda \in \mathbb{N}$, any $pars \leftarrow\!\!\$\ \mathsf{PG}(1^\lambda)$, any $(sk, pk) \leftarrow\!\!\$\ \mathsf{KG}(pars)$, any $M \in \mathsf{MSp}$, any $L \in \mathsf{LSp}$, and any $C \leftarrow\!\!\$\ \mathsf{Enc}(pars, pk, M, L)$, we have that $\mathsf{Dec}(pars, pk, sk, C, L) = M$.

Correctness definitions do not specify how the cryptosystem behaves should the incorrect key pair be used upon decryption. They are merely a property of the scheme when decryption is run on the correct keys. Robustness definitions, on the other hand, model how this behavior on different keys, and were implicit in a number of works prior to [2]. We briefly recall a number of robustness notions that exist in the literature.

B Collision Freeness and Well-addressedness

Collision freeness. This notion was introduced in [25] and captures the idea that a ciphertext should decrypt to two *distinct* messages when decrypted using two distinct secret keys. While this notion seems to be of standalone interest only in limited scenarios (in the case of *expected* messages for example) The author considers both *weak* and *strong* collision freeness, both in the CPA and CCA setting (in the former case there is no access to the Dec procedure). These notions were introduced as intermediate steps to achieve strong robustness. The corresponding games are described in Figure 8. Note that, since it requires the two messages to also collide (as well as being valid), collision freeness can be considered as a relaxation of the notion of robustness (which only requires validity).

Strong correctness. The notion of strong correctness was introduced by Barth et al. [3]. This notion is similar to the weak robustness notion except that the adversary is required to output the message before receiving the public parameters of the system. This notion is therefore is even weaker than WROB. As it this notion is about two public keys, in our view it is not a "correctness" condition.

Well-addressedness. This notion was introduced By Hofheinz and Weinreb [21] and has applications in the context of public-key encryption with keyword search. This notion is defined for identity-based encryption schemes, and is *incomparable* to strong robustness for IBE schemes. We have included the

proc Initialize
$pars \leftarrow\$ \mathsf{PG}$
$(sk_0, pk_0) \leftarrow\$ \mathsf{KG}(pars)$
$(sk_1, pk_1) \leftarrow\$ \mathsf{KG}(pars)$
Return $(pars, pk_0, pk_1)$

proc Dec(b, C) // ATK = CCA
If $b \notin \{0,1\}$ Then Return $\perp$
$M \leftarrow \mathsf{Dec}(pars, pk_b, sk_b, C)$
Return M

proc Finalize(M) // Weak collision freeness
$M_0 \leftarrow M$; $C \leftarrow\$ \mathsf{Enc}(pars, pk_0, M_0)$
$M_1 \leftarrow \mathsf{Dec}(pars, pk_1, sk_1, C)$
Return $(M_0 \neq \perp) \wedge (M_1 \neq \perp) \wedge (M_0 = M_1)$

proc Finalize(C) // Strong collision freeness
$M_0 \leftarrow \mathsf{Dec}(pars, pk_0, sk_0, C)$
$M_1 \leftarrow \mathsf{Dec}(pars, pk_1, sk_1, C)$
Return $(M_0 \neq \perp) \wedge (M_1 \neq \perp) \wedge (M_0 = M_1)$

Fig. 8. Games defining (two-user) weak/strong collision freeness.

definition below. As we shall see, the ID-analogues of notion of robustness that we introduce are strong enough to imply well-addressedness.

proc Initialize
$pars \leftarrow\$ \mathcal{IBE}.\mathsf{PG}$
$(msk, mpk) \leftarrow\$ \mathcal{IBE}.\mathsf{MPG}(pars)$
Return (msk, mpk)

proc Encrypt(id_0)
$M_0 \leftarrow\$ \mathsf{MSp}$
$C \leftarrow\$ \mathcal{IBE}.\mathsf{Enc}(pars, mpk, id_0, M_0)$
Return C

proc Finalize(id_1)
If $(id_0 = id_1)$ Then Return F
$sk_{id_1} \leftarrow\$ \mathcal{IBE}.\mathsf{KG}(pars, msk, id_1)$
$M_1 \leftarrow \mathcal{IBE}.\mathsf{Dec}(pars, mpk, id_1, sk_{id_1}, C)$
Return $(M_1 \neq \perp)$

Fig. 9. Games defining well-addressedness of an IBE scheme. An adversary is legitimate if it calls **Encrypt** exactly once.

C On the Fairness of Sako's Protocol using SROB Encryption Schemes

We first recall the $\mathcal{CS}^\star$ scheme. The common public parameters for $\mathcal{CS}^\star$ consist of a group $\mathbb{G}$ of prime order p and the description of a family of functions $H : \mathsf{Keys}(H) \times \mathbb{G}^3 \to \mathbb{G}$. The algorithms of $\mathcal{CS}^\star$ are as follows:

PG: Choose $K \leftarrow\$ \mathsf{Keys}(H)$, $g_1 \leftarrow\$ \mathbb{G}$, and $w \leftarrow\$ \mathbb{Z}_p^*$. Let $g_2 \leftarrow g_1^w$. Return (g_1, g_2, K).
KG(g_1, g_2, K)**:** Choose random exponents $x_1, x_2, y_1, y_2, z_1, z_2 \leftarrow\$ \mathbb{Z}_p$ and compute

$$e = g_1^{x_1} g_2^{x_2}, \qquad f = g_1^{y_1} g_2^{y_2}, \qquad h = g_1^{z_1} g_2^{z_2}.$$

The public key is $pk = (e, f, h)$ and the private key is $sk = (x_1, x_2, y_1, y_2, z_1, z_2)$.

Enc$(pars, pk, M)$**:** To encrypt a message $M \in \mathbb{G}$,

1. Pick $u \leftarrow\$ \mathbb{Z}_p^*$ and compute $a_1 = g_1^u$, $a_2 = g_2^u$, and $b = h^u$.
2. Let $c \leftarrow b \cdot M$, $v \leftarrow H(K, (a_1, a_2, c))$, $d \leftarrow e^u f^{uv}$.

The ciphertext is $C = (a_1, a_2, c, d)$.

Dec$(pars, pk, sk, C)$**:** Parse the ciphertext C as (a_1, a_2, c, d). Compute $v \leftarrow H(K, (a_1, a_2, c))$, $M \leftarrow c \cdot a_1^{-z_1} a_2^{-z_2}$. If $d \neq a_1^{x_1+y_1 v} a_2^{x_2+y_2 v}$ then set $M \leftarrow \perp$. If $a_1 = 1$ then set $M \leftarrow \perp$. Return M.

The attack works as follows. Let $V = \{v_1, ..., v_N\}$ be the set of possible bid values. The auctioneer runs PG to obtain the public parameters (K, g_1, g_2). He chooses a fixed message $M \in \mathbb{G}$ as per Sako's protocol.

He selects $u, z_1, z_2, \alpha_1, \alpha_2 \leftarrow \mathbb{Z}_p^*$ and computes $a_1 = g_1^u$, $a_2 = g_2^u$, $b = a_1^{z_1} a_2^{z_2}$, $c = b \cdot M$, and $d = a_1^{\alpha_1} a_2^{\alpha_2}$. He then computes $v = H(K, (a_1, a_2, c))$. If $v = 0$, the auctioneer re-samples and re-computes the values, until $v \neq 0$. He then considers the following system of multivariate linear equations

$$\begin{cases} x_1 + vy_1 = \alpha_1 \\ x_2 + vy_2 = \alpha_2 \end{cases}$$

and finds N solutions $(x_{i_1}, x_{i_2}, y_{i_1}, y_{i_2})$ with $i \in \{1, ..., N\}$, where all the values are in $\mathbb{Z}_p$.

The auctioneer sets sk_i to be $(x_{i_1}, x_{i_2}, y_{i_1}, y_{i_2}, z_1, z_2)$ for $i \in \{1, ..., N\}$. He passes u to the cheating bidder and publishes all the public keys $pk_i = (g_1^{x_{i_1}} g_2^{x_{i_1}}, g_1^{y_{i_1}} g_2^{y_{i_2}}, g_1^{z_1} g_2^{z_2})$ with $i \in \{1, ..., N\}$.

The cheating bidder can now bid for the value v_i by encrypting M with randomness u under the public key pk_i to get ciphertext C. Such an encrypted bid C will decrypt to M under **any** sk_j with $j \in \{1, ..., N\}$, since $x_{i_1} + vy_{i_1} = x_{j_1} + vy_{j_1}$ and $x_{i_2} + vy_{i_2} = x_{j_2} + vy_{j_2}$, by construction. This means that during the protocol, the auctioneer can first observe the highest honest bid (say $h < N$). Then, he can declare the cheating bidder as the winner (for the bid $h + 1$) by revealing the private key sk_{h+1}. This clearly gives the dishonest bidder and colluding auctioneer a cheating strategy and breaks the fairness of the protocol.

Remark 1. It may be argued that the above attack can be detected by the bidders, as the maliciously generated public keys share the same third component. Although this is a valid point, it may be unreasonable to assume that the bidders perform such checks outside the protocol description. Indeed, one (or the) goal of robustness is to ensure that such checks are *already* implemented within the decryption algorithm. Let us note that the attack of Abdalla et al. on the robustness of ElGamal also falls within the category of such "traceable" attacks, as the ciphertexts are of the form $(1, C)$. Despite this, and in order to further justify the relevance of the new notions, we demonstrate in Appendix D an untraceable attack on the modified Kurosawa–Desmedt encryption scheme (which was proven strongly robust under chosen-ciphertext attacks in [24]).

Remark 2. The alert reader might notice that, if the auctioneer is allowed to collude with bidders, then other, more direct attacks on the protocol are possible: indeed, the auctioneer can first decrypt all honest bidders' ciphertexts and tell the cheating bidder what his incremental bid has to be. These attacks can all be prevented, informally at least, by having all players initially send a commitment to their ciphertexts (as already suggested in [2, Appendix C]), so that a colluding bidder and auctioneer do not see honest bidders' ciphertexts before a cheating bid is sent. In contrast, our attack works even in this case.

D A Variant of Kurosawa–Desmedt Is SROB-CCA but Not CROB

We recall the Kurosawa–Desmedt ($\mathcal{KD}$) cryptosystem [23]. Here, the common public parameters consist of a group $\mathbb{G}$ of prime order $p > 2^\lambda$, with generators $g_1, g_2 \in_R \mathbb{G}$. They also include the description of a universal one-way hash function $H : \{0,1\}^* \to \mathbb{Z}_p$, a key-derivation function $\mathsf{KDF} : \mathbb{G} \to \{0,1\}^k$, for some integer $k \in \text{poly}(1^\lambda)$, a symmetric *authenticated* encryption scheme (E, D) of key length k.

$\mathsf{KG}(pars)$**:** Given common public parameters $pars = (\mathbb{G}, g_1, g_2, H)$, choose $x_1, x_2, y_1, y_2 \leftarrow_\$ \mathbb{Z}_p$ and compute

$$e = g_1^{x_1} g_2^{x_2}, \qquad f = g_1^{y_1} g_2^{y_2}$$

The public key is $pk = (e, f)$ and the private key is $sk = (x_1, x_2, y_1, y_2)$.

$\mathsf{Enc}(pars, pk, M)$**:** To encrypt a message $M \in \mathbb{G}$,

1. Pick $u \leftarrow_\$ \mathbb{Z}_p$ and compute

$$a_1 = g_1^u, \qquad a_2 = g_2^u, \qquad d = (e \cdot f^v)^u,$$

where $v = H(a_1, a_2) \in \mathbb{Z}_p$.

2. Compute $K = \mathsf{KDF}(d) \in \{0,1\}^k$, $c = \mathsf{E}_K(M)$.

The ciphertext is $C = (a_1, a_2, c)$.

$\mathsf{Dec}(pars, pk, sk, C)$**:** Parse the ciphertext C as (a_1, a_2, c). Compute $v = H(a_1, a_2)$, $d = a_1^{x_1+v\cdot y_1} \cdot a_2^{x_2+v\cdot y_2}$, and $K = \mathsf{KDF}(d) \in \{0,1\}^k$. Then, return $m = \mathsf{D}_K(c)$ (which may be $\perp$ if c fails to properly decrypt under the key K).

The above algorithms describe the original Kurosawa–Desmedt encryption scheme. Following [2], we denote by $\mathcal{KD}^\star$ the modified $\mathcal{KD}$ scheme where the encryption exponent $u = 0$ is explicitly disallowed: namely, the sender chooses $u \leftarrow\!\!\$\ \mathbb{Z}_p^*$ (instead of $u \leftarrow\!\!\$\ \mathbb{Z}_p$) at encryption and the receiver outputs $\perp$ upon receiving a ciphertext (a_1, a_2, c) such that $a_1 = 1_{\mathbb{G}}$. We prove in [24] that $\mathcal{KD}^\star$ is strongly robust (with some conditions on the symmetric components).

We will next see that $\mathcal{KD}^\star$ is *not* fully robust (and hence neither completely robust). We construct an adversary $\mathcal{A}$ which gets as input *pars*, picks $m \leftarrow\!\!\$\ \mathbb{G}$ and $u, \alpha_1, \alpha_2 \leftarrow\!\!\$\ \mathbb{Z}_p^*$. It then computes

$$a_1 = g_1^u, \qquad a_2 = g_2^u, \qquad v = H(a_1, a_2), \qquad d = a_1^{\alpha_1} a_2^{\alpha_2}.$$

Now consider the following system of multivariate linear equations

$$\begin{cases} x_1 + vy_1 = \alpha_1 \\ x_2 + vy_2 = \alpha_2 \end{cases}$$

Unless $v = 0$ such system has an infinite number of solutions. (In the case $v = 0$, $\mathcal{A}$ re-samples u). In particular, let $(x_{10}, x_{20}, y_{10}, y_{20})$ and $(x_{11}, x_{21}, y_{11}, y_{21})$ be two integer solutions to the system.

Now $\mathcal{A}$ sets $pk = (e, f)$ and $pk' = (e', f')$ to be

$$e = g_1^{x_{10}} g_2^{x_{20}}, \qquad f = g_1^{y_{10}} g_2^{y_{20}}, \qquad e' = g_1^{x_{11}} g_2^{x_{21}}, \qquad f' = g_1^{y_{11}} g_2^{y_{21}},$$

i.e., the public keys corresponding to secret keys $sk = (x_{10}, x_{20}, y_{10}, y_{20})$ and $sk' = (x_{11}, x_{21}, y_{11}, y_{21})$, respectively. The adversary $\mathcal{A}$ finally computes $d = a_1^{\alpha_1} a_2^{\alpha_2}$, $K = \mathsf{KDF}(d)$, and $c = \mathsf{E}_K(M)$ and obtains a ciphertext $C = (a_1, a_2, c)$. Its output for the full robustness game will be $\big(C, pk, pk', sk, sk'\big)$. By the choice of sk and sk', it is clear that C, decrypted under both secret keys, will return a valid message $M \neq \perp$ with probability 1.

In [24], the $\mathcal{KD}^\star$ scheme was proved SROB-CCA assuming that the key-derivation function is collision-resistant and that the symmetric encryption scheme is key binding (i.e., each ciphertext is only valid for one key). Even when these conditions are satisfied, the above attack still works since both keys recover the same $d = a_1^{\alpha_1} a_2^{\alpha_2}$ at decryption.

When $\mathcal{KD}^\star$ is used to implement the auction protocol of [28], this attack is much harder to detect than that of Section 2.2 and the one of [2]: the only apparent way to make it evident is to audit the entire system and force the auctioneer to reveal all private keys.

E Proof of the Characterization Theorem

We prove the theorem via a sequence of propositions as follows.

Proposition 2 (CROB $\iff$ FROB $\wedge$ KROB $\wedge$ XROB**).** *A PKE scheme is* CROB *if and only if it is simultaneously* FROB*,* KROB*, and* XROB*.*

Proof. For the forward direction, suppose there is an adversary which wins one of the FROB, KROB, or XROB games. Then this adversary also wins the CROB game by querying the FROB winning tuples to the **Dec** oracle, the KROB winning tuples to the **Enc** oracle, and finally the first XROB winning tuple to the **Enc** oracle and the second to the **Dec** oracle.

For the backward direction, note that a pair of winning tuples for the CROB game can arise in one of three possible ways: (1) Both tuples were added to the list through decryption queries. This translates into a winning output for an FROB adversary; (2) Both tuples were added to the list through encryption queries. This translates into a winning output for a KROB adversary; (3) One tuple was added to the list through an encryption query and the other through a decryption query. This translates into a winning output for an XROB adversary. □

For the first separation, we prove the stronger statement that KROB and XROB together are insufficient to guarantee SROB-CPA (and hence also fail to imply FROB). As we shall see in Proposition 8, we have that XROB $\implies$ WROB-CCA, and hence this is the "best separation" one can hope for.

Proposition 3 (KROB $\wedge$ XROB $\not\Longrightarrow$ SROB-CPA)**.** *Let $\mathcal{PKE}$ be a public-key encryption scheme which is* KROB *and* XROB*. Then there is a scheme $\mathcal{PKE}'$ which is* KROB *and* XROB*, but fails to be* SROB-CPA*.*

Proof. We define the required scheme $\mathcal{PKE}'$ to be identical to $\mathcal{PKE}$ except for its encryption and decryption algorithms, which we modify as follows:

$\mathsf{Enc}'(pars, pk, M)$**:** Run $\mathsf{Enc}(pars, pk, M)$ to obtain ciphertext C. Return $0\|C$.
$\mathsf{Dec}'(pars, pk, sk, c\|C)$**:** If $c = 0$ return $\mathsf{Dec}(pars, pk, sk, C)$. If $c = 1$ return a fixed (e.g., the lexicographically smallest) message $M^\star$ in the message space for pk.

Note that $\mathcal{PKE}'$ is a correct public-key encryption scheme. Scheme $\mathcal{PKE}'$ is not SROB-CPA. Consider the adversary $\mathcal{A}$ which queries **GetEK** twice to obtain, with overwhelming probability, two distinct public keys pk_0 and pk_1. Now $\mathcal{A}$ picks a random C from the ciphertext space and gives $(1\|C, pk_0, pk_1)$ as its final output. It is easy to see that $\mathcal{A}$ wins with an overwhelming probability: $pk_0 \neq pk_1$ and the decryption of $1\|C$ always returns a valid message by construction.

Next we show $\mathcal{PKE}'$ is still KROB. It is easy to see that the tweaks in the modified scheme do not affect the KROB game, since the new encryption algorithm prepends a zero-bit to *all* ciphertexts.

In order to show that $\mathcal{PKE}'$ is still XROB, suppose an adversary $\mathcal{A}(pars)$ outputs a winning tuple $(M, pk_0, r_0, C_1, pk_1, sk_1)$, where pk_0 and pk_1 are two distinct public keys. Note it must be the case that $C_1 = 0\|\tilde{C}_1$, as otherwise $\mathcal{A}$ cannot win the XROB game (M is encrypted using Enc in the game). Now an adversary $\mathcal{B}$ can win the XROB game for $\mathcal{PKE}$ by outputting $(M, pk_0, r_0, \tilde{C}_1, pk_1, sk_1)$. □

Proposition 4 (FROB $\wedge$ XROB $\not\Longrightarrow$ KROB)**.** *Let $\mathcal{PKE}$ be a public-key encryption scheme which is* FROB *and* XROB*. Then there is a scheme $\mathcal{PKE}'$ which is* FROB *and* XROB*, but fails to be* KROB*.*

Proof. We define the required scheme $\mathcal{PKE}'$ as follows.

$\mathsf{PG}'(1^\lambda)$**:** Run $\mathsf{PG}(1^\lambda)$ to obtain $pars$. Return $pars$.
$\mathsf{KG}'(pars)$**:** Run $\mathsf{KG}(pars)$ to obtain (sk, pk). Return $(sk, 0\|pk)$.
$\mathsf{Enc}'(pars, b\|pk, M)$**:** If $b = 1$, output 1^λ. If $b = 0$, run $\mathsf{Enc}(pars, pk, M)$, obtain ciphertext C and output $0\|C$.
$\mathsf{Dec}'(pars, b\|pk, sk, c\|C)$**:** If $b = 1$ or $c = 1$ return $\perp$. Else, run and output $\mathsf{Dec}(pars, pk, sk, C)$.

Note that $\mathcal{PKE}'$ is a correct public-key encryption scheme. To see that $\mathcal{PKE}'$ is not KROB, note that an adversary which outputs $1\|pk_0$ and $1\|pk_1$, for two valid public keys pk_1 and pk_0 wins the KROB game (for any pair of messages and any pair of random coins) as the resulting ciphertext in both cases is 1^λ.

Next we show that $\mathcal{PKE}'$ is still FROB Suppose an adversary $\mathcal{A}(pars)$ outputs a winning tuple $(c\|C, b_0\|pk_0, b_1\|pk_1, sk_0, sk_1)$, where $b_0\|pk_0$ and $b_1\|pk_1$ are two distinct public keys. Note it must be the case that $b_0 = b_1 = 0$, as otherwise decryption rejects and $\mathcal{A}$ cannot win the FROB game. Therefore it is necessarily the case that $pk_0 \neq pk_1$, and an adversary $\mathcal{B}$ can also win the FROB game against $\mathcal{PKE}$ by outputting $(C, pk_0, pk_1, sk_0, sk_1)$.

In order to show that $\mathcal{PKE}'$ is also XROB, suppose an adversary $\mathcal{A}(pars)$ outputs a winning tuple $(M, b_0\|pk_0, r_0, c\|C, b_1\|pk_1, sk_1)$, where $b_0\|pk_0$ and $b_1\|pk_1$ are two distinct public keys. We must have that $b_1 = 0$ and $c = 0$ as otherwise decryption rejects. We must also have that $b_0 = c$, as the first bit of a ciphertext output by encryption matches the first bit of the public key. Therefore $b_0 = b_1 = c = 0$, and an adversary $\mathcal{B}$ can also win the XROB game against $\mathcal{PKE}$ by outputting $(M, pk_0, r_0, C, pk_1, sk_1)$. □

These propositions, together with Proposition 1, complete the proof of Theorem 1.

F Relation with WROB and SROB

It is clear that FROB $\implies$ USROB $\implies$ SROB-CCA as the adversary becomes progressively more restricted in each game. We prove the rest of the theorem via a sequence of propositions.

Proposition 5 (USROB $\not\Longrightarrow$ FROB**).** *Let $\mathcal{PKE}$ be a public-key encryption scheme which is* USROB*. Then there is a scheme $\mathcal{PKE}'$ which is* USROB*, but fails to be* FROB*.*

Proof. We define the required scheme $\mathcal{PKE}'$ as follows.

$\mathsf{PG}'(1^\lambda)$**:** Run $\mathsf{PG}(1^\lambda)$ to obtain $pars$. Return $pars$.
$\mathsf{KG}'(pars)$**:** Run $\mathsf{KG}(pars)$ to obtain (sk, pk). Return $(sk, 0\|pk)$.
$\mathsf{Enc}'(pars, b\|pk, M)$**:** Run $\mathsf{Enc}(pars, pk, M)$ to obtain C. Return C.
$\mathsf{Dec}'(pars, b\|pk, sk, C)$**:** Return $\mathsf{Dec}(pars, pk, sk, C)$.

Observe that $\mathcal{PKE}'$ is a correct public-key encryption scheme. Furthermore, $\mathcal{PKE}'$ is not FROB. Consider the adversary $\mathcal{A}$ which runs $\mathsf{KG}'(pars)$ to get a valid key-pair $(sk, 0\|pk)$, picks a random message M and runs $\mathsf{Enc}'(pars, 0\|pk, M)$ to obtain a ciphertext C. The adversary $\mathcal{A}$ gives $(C, 0\|pk, 1\|pk, sk, sk)$ as its final output. It is easy to see that $\mathcal{A}$ wins with probability 1: $0\|pk \neq 1\|pk$ and the decryption of C with the secret key sk returns a valid message due to correctness.

We now show $\mathcal{PKE}'$ is USROB. Suppose there is an adversary $\mathcal{A}$ which wins the USROB game against $\mathcal{PKE}'$. We construct an adversary $\mathcal{B}$ that interacts with $\mathcal{A}$ to win the USROB game against $\mathcal{PKE}$ with the same probability. Algorithm $\mathcal{B}(pars)$ runs $\mathcal{A}(pars)$ and handles $\mathcal{A}$'s queries by forwarding them to its own oracles simply prepending a zero-bit to all the public keys sent and received. Finally, when $\mathcal{A}$ outputs (C, pk_0, pk_1) with $pk_0 \neq pk_1$, $\mathcal{B}$ also outputs the same tuple. It's easy to see that $\mathcal{B}$ provides a perfect simulation of $\mathcal{A}$'s environment, and that if $\mathcal{A}$ wins so does $\mathcal{B}$. □

Proposition 6 (SROB-CCA $\not\Longrightarrow$ USROB**).** *Let $\mathcal{PKE}$ be a public-key encryption scheme which is* SROB-CCA*. Then there is a scheme $\mathcal{PKE}'$ which is* SROB-CCA*, but fails to be* USROB*.*

Proof. We define the required scheme $\mathcal{PKE}'$ as follows.

$\mathsf{PG}'(1^\lambda)$**:** Run $\mathsf{PG}(1^\lambda)$ to obtain $pars$. Return $pars$.
$\mathsf{KG}'(pars)$**:** Run $\mathsf{KG}(pars)$ to obtain (sk, pk). Sample $s \leftarrow\!\!\$\ \{0,1\}^\lambda$ and return $(sk\|s, pk)$.
$\mathsf{Enc}'(pars, pk, M)$**:** Run $\mathsf{Enc}(pars, pk, M)$ to obtain ciphertext C. Return C.
$\mathsf{Dec}'(pars, pk, sk\|s, C)$**:** Parse C as $C_1\|C_2$. Run $\mathsf{Dec}(pars, pk, sk, C_1)$. If the output is a valid message M, return M. Otherwise, check if $C_2 = s$, and if so return a fixed (e.g., the lexicographically smallest) message $M^\star$ in the message space for pk. Else, return $\perp$.

$\mathcal{PKE}'$ is a correct public-key encryption scheme.[9] We first prove $\mathcal{PKE}'$ is not USROB by constructing an adversary $\mathcal{A}$ which wins the USROB game against $\mathcal{PKE}'$. Algorithm $\mathcal{A}(pars)$ queries **GetEK** on two identities id_0, id_1 to obtain, with overwhelming probability, two distinct public keys pk_0 and pk_1. It then queries **GetDK**(id_0) and receives $sk_0 \| s_0$. Algorithm $\mathcal{A}$ runs $\mathsf{Enc}'(pars, pk_1, M_1)$, where M_1 is any (valid) message, obtaining C_1. Adversary $\mathcal{A}$ then sets $C_2 := s_0$ and outputs $(C := (C_1 \| C_2), pk_0, pk_1)$ as its final output. It is easy to see that this is a winning strategy for $\mathcal{A}$: C when decrypted with respect to pk_1 would return M_1 due to the correctness of the scheme. Furthermore, when decrypted with respect to pk_0 we obtain a valid message M since $C_2 = s_0$.

We now prove that $\mathcal{PKE}'$ is SROB-CCA. Suppose there is an adversary $\mathcal{A}$ which wins the SROB-CCA game against $\mathcal{PKE}'$. We construct an adversary $\mathcal{B}$ that interacts with $\mathcal{A}$ to win the SROB-CCA game against $\mathcal{PKE}$. Algorithm $\mathcal{B}(pars)$ runs $\mathcal{A}(pars)$ and handles its queries as follows:

- **GetEK**(id) query: $\mathcal{B}$ queries its own **GetEK**(id) to get a public key pk. It then selects and stores a random bit-string s of length λ. Algorithm $\mathcal{B}$ gives pk to $\mathcal{A}$.
- **GetDK**(id) query: $\mathcal{B}$ queries its own **GetDK**(id) for the corresponding secret key sk. Algorithm $\mathcal{B}$ then retrieves and appends s to sk and gives $sk \| s$ as the response to $\mathcal{A}$.
- **Dec**$(C_1 \| C_2, pk)$ query: $\mathcal{B}$ passes (C_1, pk) to its own oracle. If the answer is a valid message M, $\mathcal{B}$ forwards M to $\mathcal{A}$. If the output is $\perp$, $\mathcal{B}$ checks whether C_2 is equal to s (which it holds). If so, $\mathcal{B}$ outputs $M^\star$ as the response to $\mathcal{A}$'s query. If not, $\mathcal{B}$ returns $\perp$.

Finally, when $\mathcal{A}$ outputs $(C_1 \| C_2, pk_0, pk_1)$, $\mathcal{B}$ outputs (C_1, pk_0, pk_1).

We note that $\mathcal{B}$ provides a perfect simulation of $\mathcal{A}$'s environment and that $\mathcal{B}$ wins whenever $\mathcal{A}$ wins unless $\mathcal{A}$ does so by guessing the s-component of either pk_0 or pk_1. Since these s-components are random and information theoretically hidden from $\mathcal{A}$'s view, the probability of this event is at most $2 \cdot \frac{1}{2^\lambda}$. This completes the proof. □

Note that by Proposition 4 we also have that SROB-CCA $\not\Longrightarrow$ KROB. The following proposition provides a separation in the reverse direction.

Proposition 7 (KROB $\not\Longrightarrow$ WROB-CPA)**.** *Let $\mathcal{PKE}$ be a public-key encryption scheme which is* KROB*. Then there is a scheme $\mathcal{PKE}'$ which is* KROB*, but fails to be* WROB-CPA*.*

Proof. We define the required scheme $\mathcal{PKE}'$ to be identical to $\mathcal{PKE}$ except its decryption algorithm, which we modify as follows:

$\mathsf{Dec}'(pars, pk, sk, C)$**:** If $\mathsf{Dec}(pars, pk, sk, C)$ is a valid message M, return M. If not return a fixed (e.g., the lexicographically smallest) message $M^\star$ in the message space for pk.

It is easy to see that $\mathcal{PKE}'$ is correct but it is not WROB-CPA: the decryption algorithm never returns $\perp$. However, the modified scheme is still KROB as the tweaked decryption algorithm does not affect the KROB game. □

By Proposition 1 we have that SROB-CCA $\not\Longrightarrow$ XROB, and by Proposition 3 we have that XROB $\not\Longrightarrow$ SROB-CPA. The implication to WROB is proved next.

Proposition 8 (XROB $\Longrightarrow$ WROB-CCA)**.** *Let $\mathcal{PKE}$ be a public-key encryption scheme which is* XROB*. Then $\mathcal{PKE}$ is also* WROB-CCA*.*

Proof. Let $\mathcal{A}$ be a WROB-CCA adversary. We construct an XROB adversary $\mathcal{B}$ as follows. Algorithm $\mathcal{B}$ on input *pars* runs $\mathcal{A}(pars)$ and answers its various queries as follows:

[9] If the decryption rule was defined to check for the equality before running Dec, correctness would only hold with overwhelming probability.

– **GetEK**(id) query: $\mathcal{B}$ generates a new key-pair (sk, pk), stores them on a list, and passes the public key to $\mathcal{A}$.
– **GetDK**(pk) query: $\mathcal{B}$ retrieves the secret key corresponding to pk and pass it on to $\mathcal{A}$.
– **Dec**(C, pk) query: $\mathcal{B}$ retrieves the secret key corresponding to pk, uses it to decrypt the ciphertext, and returns the result to $\mathcal{A}$.

When $\mathcal{A}$ terminates by outputting (M, pk_0, pk_1), algorithm $\mathcal{B}$ proceeds as follows. It first samples coins r_0 for the encryption algorithm and computes $C_1 := \mathsf{Enc}(pars, pk_1, M; r)$. It then returns $(M, pk_0, r_0, C_1, pk_1, sk_1)$ as its final output, where sk_1 is the secret key corresponding to pk_1. It is easy to see that $\mathcal{B}$ runs $\mathcal{A}$ in an environment identical to WROB-CCA. Furthermore, whenever $\mathcal{A}$ is successful in winning this game, algorithm $\mathcal{B}$ also breaks the XROB property. □

G Definitions for Commitment Schemes

Recall that a commitment scheme $(\mathsf{CPG}, \mathsf{Com}, \mathsf{Ver})$ is a triple of probabilistic polynomial-time (PPT) algorithms where, on input of a security parameter λ, CPG outputs public parameters $cpars$; Com takes as input a message m and outputs a commitment/de-commitment pair $(com, dec) \leftarrow\!\!\$\ \mathsf{Com}(cpars, m)$, and $\mathsf{Ver}(cpars, m, com, dec)$ is deterministic and outputs 0 or 1. The correctness property guarantees that Ver always outputs 1 whenever (com, dec) is obtained by committing to m using honestly generated parameters.

The *binding* property demands that, given $cpars$, no PPT adversary should be able to produce a commitment that can be opened to two distinct messages. More precisely, for any PPT adversary $\mathcal{A}$ we require that the following advantage term is negligible as a function of λ.

$$\mathbf{Adv}^{\mathrm{bind}}_{\mathcal{CMT}}(\mathcal{A}) := \Pr[\ \mathsf{Ver}(cpars, m_0, com, dec_0) = \mathsf{Ver}(cpars, m_1, com, dec_1) = 1 \ \wedge\ m_0 \neq m_1\ : \\ cpars \leftarrow\!\!\$\ \mathsf{CPG}(1^\lambda);\ (com, m_0, dec_0, m_1, dec_1) \leftarrow\!\!\$\ \mathcal{A}(cpars)\ ]$$

A commitment is also said *hiding* if for any PPT adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ the following advantage term is negligible as a function of λ.

$$\mathbf{Adv}^{\mathrm{hide}}_{\mathcal{CMT}}(\mathcal{A}) := |2 \cdot \Pr[\ b = b'\ :\ cpars \leftarrow\!\!\$\ \mathsf{CPG}(1^\lambda);\ b \leftarrow\!\!\$\ \{0,1\};\ (m_0, m_1, st) \leftarrow\!\!\$\ \mathcal{A}_1(cpars); \\ (com, dec) \leftarrow\!\!\$\ \mathsf{Com}(cpars, m_b);\ b' \leftarrow\!\!\$\ \mathcal{A}_2(com, st)\ ] - 1|$$

H Proof of Theorem 3

Proof. We treat the three possible cases corresponding to FROB, KROB, and XROB.

Given an FROB adversary $\mathcal{A}$ against $\overline{\mathcal{PKE}}$ we construct an adversary $\mathcal{B}_1$ that will interact with $\mathcal{A}$ to break the binding property of $\mathcal{CMT}$. The game proceeds as follows.

Let $\mathcal{C}$ be $\mathcal{B}_1$'s challenger. $\mathcal{C}$ runs CPG to obtain the commitment schemes's parameters $cpars$ and passes them on to $\mathcal{B}_1$. Algorithm $\mathcal{B}_1$ runs PG to obtain $pars$, which it passes to $\mathcal{A}$ together with $cpars$. Algorithm $\mathcal{B}_1$ handles all of $\mathcal{A}$'s queries.

Finally, $\mathcal{A}$ outputs $(C, pk_0, pk_1, sk_0, sk_1)$, where $C = (c, com)$ and $pk_0 \neq pk_1$. Now, $\mathcal{B}_1$ runs the decryption algorithm twice, obtaining $M_0 \leftarrow \mathsf{Dec}(pars, pk_0, sk_0, C)$ and $M_1 \leftarrow \mathsf{Dec}(pars, pk_1, sk_1, C)$. Let Succ be the event that $M_0 \neq \perp$ and $M_1 \neq \perp$. If Succ occurs then $\mathcal{B}_1$ parses M_0 and M_1 into $\tilde{M}_0 \| dec_0$ and $\tilde{M}_1 \| dec_1$, respectively. It then gives $(com, pk_0, pk_1, dec_0, dec_1)$ to $\mathcal{C}$ as its final output.

$\mathcal{B}_1$ provides a perfect simulation for $\mathcal{A}$ as well as a legal strategy for attacking the binding property of $\mathcal{CMT}$, provided Succ occurs. Since this happens whenever $\mathcal{A}$ is a winning adversary against the full robustness of $\overline{\mathcal{PKE}}$, we have that $\mathcal{B}_1$'s advantage is the same as $\mathcal{A}$'s.

Given a KROB adversary $\mathcal{A}$ against $\overline{\mathcal{PKE}}$ we construct an adversary $\mathcal{B}_2$ that will interact with $\mathcal{A}$ to break the binding property of $\mathcal{CMT}$. The game proceeds as follows.

Let $\mathcal{C}$ be $\mathcal{B}_2$'s challenger. $\mathcal{C}$ runs CPG to obtain the commitment schemes's parameters *cpars* and passes them on to $\mathcal{B}_2$. Algorithm $\mathcal{B}_2$ runs PG to obtain *pars*, which it passes to $\mathcal{A}$ together with *cpars*. Algorithm $\mathcal{B}_2$ handles all of $\mathcal{A}$'s queries.

$\mathcal{A}$ outputs $(M_0, M_1, pk_0, pk_1, r_0, r_1)$, where $pk_0 \neq pk_1$, $r_0 = (r_{0_{cmt}}, r_{0_{enc}})$ and $r_1 = (r_{1_{cmt}}, r_{1_{enc}})$. Now, $\mathcal{B}_2$ runs $\mathsf{Com}(cpars, pk_0; r_{0_{com}})$, obtaining (com_0, dec_0), and $\mathsf{Com}(cpars, pk_1; r_{1_{com}})$, obtaining (com_1, dec_1). Then $\mathcal{B}_2$ computes ciphertexts $c_0 = \mathsf{Enc}(pars, pk_0, M_0 \| dec_0; r_{0_{enc}})$ and $c_1 = \mathsf{Enc}(pars, pk_1, M_1 \| dec_1; r_{1_{enc}})$. Let $C_0 = (c_0, com_0)$ and $C_1 = (c_1, com_1)$. Let Succ be the event that $C_0 = C_1$ (and therefore $c_0 = c_1 = c$ and $com_0 = com_1 = com$). If Succ occurs then $\mathcal{B}_2$ outputs $(com, pk_0, pk_1, dec_0, dec_1)$ and gives it to $\mathcal{C}$.

$\mathcal{B}_2$ provides a perfect simulation for $\mathcal{A}$ as well as a legal strategy for attacking the binding property of $\mathcal{CMT}$, provided Succ occurs. Since this happens whenever $\mathcal{A}$ is a winning adversary against the key-less robustness of $\overline{\mathcal{PKE}}$, we have that $\mathcal{B}_2$'s advantage is the same as $\mathcal{A}$'s.

Given an XROB adversary $\mathcal{A}$ against $\overline{\mathcal{PKE}}$ we construct an adversary $\mathcal{B}_3$ that will interact with $\mathcal{A}$ to break the binding property of $\mathcal{CMT}$. The game proceeds as follows.

Let $\mathcal{C}$ be $\mathcal{B}_3$'s challenger. $\mathcal{C}$ runs CPG to obtain the commitment scheme's parameters *cpars* and passes them on to $\mathcal{B}_3$. The latter runs PG to obtain *pars*, which it passes to $\mathcal{A}$ together with *cpars*. Then, $\mathcal{B}_3$ handles all of $\mathcal{A}$'s queries during the game.

Eventually, $\mathcal{A}$ outputs $(M_0, pk_0, r_0, C_1, pk_1, sk_1)$, where $pk_0 \neq pk_1$, $r_0 = (r_{0_{cmt}}, r_{0_{enc}})$. At this point, $\mathcal{B}_3$ runs $\mathsf{Com}(cpars, pk_0; r_{0_{com}})$, obtaining (com_0, dec_0), and $\mathsf{Dec}(pars, pk_1, sk_1, C_1)$, obtaining M_1. Then $\mathcal{B}_3$ computes $c_0 = \mathsf{Enc}(pars, pk_0, M_0 \| dec_0; r_{0_{enc}})$. Let $C_0 = (c_0, com_0)$ and $C_1 = (c_1, com_1)$. Let Succ be the event that $C_0 = C_1$ (and therefore $c_0 = c_1 = c$ and $com_0 = com_1 = com$). If Succ occurs then $\mathcal{B}_3$ outputs $(com, pk_0, pk_1, dec_0, dec_1)$ and gives it to $\mathcal{C}$.

$\mathcal{B}_3$ provides a perfect simulation for $\mathcal{A}$ as well as a legal strategy for attacking the binding property of $\mathcal{CMT}$, provided Succ occurs. Since this happens whenever $\mathcal{A}$ is a winning adversary against the key-less robustness of $\overline{\mathcal{PKE}}$, we have that $\mathcal{B}_3$'s advantage is the same as $\mathcal{A}$'s. □

I Proof of Theorem 4

Proof. The proof proceeds with a sequence of games. For each i, we call S_i that event that $\mathcal{A}$ calls **Finalize** on input $b' = 0$ in Game i.

Game 1: is the real game when the challenger's bit equals $b = 0$. Namely, the adversary $\mathcal{A}$ interacts with an actual AI-ATK challenger who sets $b = 0$ in the **Setup** procedure. The challenger always runs the **Dec** procedure according to its specification. When $\mathcal{A}$ makes its unique **LR** query $(M_0^\star, M_1^\star)$, the challenger computes $(com^\star, dec^\star) \leftarrow\!\!\$\ \mathsf{Com}(cpars, pk_0)$ and $C^\star \leftarrow\!\!\$\ \mathsf{Enc}\big(pars, pk_0, M_0^\star \| dec^\star, com^\star\big)$. The adversary $\mathcal{A}$ is given $(C^\star, com^\star)$ as a challenge ciphertext. The game ends with $\mathcal{A}$ invoking **Finalize** with a bit $b' \in \{0, 1\}$.

Game 2: is exactly like Game 1 with one difference in the description of the **LR** procedure. Namely, the challenge ciphertext $(C^\star, com^\star)$ is generated by computing a pair $(com^\star, dec^\star) \leftarrow\!\!\$\ \mathsf{Com}(cpars, pk_0)$ and a ciphertext $C^\star \leftarrow\!\!\$\ \mathsf{Enc}\big(pars, pk_1, M_1^\star \| 0^{|dec^\star|}, com^\star\big)$, where $0^{|dec^\star|}$ denotes the all-zeroes string of the same length as $dec^\star$.

We claim that there is a PPT algorithm $\mathcal{B}_1$ such that $|\Pr[S_2] - \Pr[S_1]| \leq \mathbf{Adv}^{\text{ai-atk}}_{\mathcal{PKE}}(\mathcal{B}_1)$. Namely, $\mathcal{B}_1$ interacts with its own AI-ATK challenger and plays the role of $\mathcal{A}$'s challenger. At the beginning of the game, $\mathcal{B}_1$ generates *cpars* by running CPG and, when receiving $(pars, pk_0, pk_1)$ of $\mathcal{PKE}$ from its challenger, it gives $(pars, cpars, pk_0, pk_1)$ to $\mathcal{A}$. When $\mathcal{A}$ decides to invoke the **LR** oracle on the input of $(M_0^\star, M_1^\star)$, $\mathcal{B}_1$ first computes a commitment/de-commitment pair $(com^\star, dec^\star) \leftarrow\!\!\$\ \mathsf{Com}(cpars, pk_0)$. It then defines messages

$\overline{M}_0^\star := M_0^\star \| dec^\star$, $\overline{M}_1^\star := M_1^\star \| 0^{|dec^\star|}$, sets $L^\star := com^\star$, and queries its own **LR** oracle on $(L^\star, \overline{M}_0^\star, \overline{M}_1^\star)$. The latter oracle replies with $C^\star \leftarrow_\$ \mathsf{Enc}(pars, pk_\beta, \overline{M}_\beta^\star, L^\star)$, for some random bit $\beta \in_R \{0,1\}$, and $\mathcal{B}_1$ gives $(C^\star, com^\star)$ to $\mathcal{A}$. Whenever $\mathcal{A}$ makes a **Dec** query for a pair $\big(b, (C, com)\big)$, $\mathcal{B}_1$ queries its own **Dec** oracle for the input (b, C, com) (note that this is a decryption query for the label $L = com$). When receiving the answer $M \| dec$, $\mathcal{B}_1$ checks if $\mathsf{Ver}(cpars, pk_b, com, dec) = 1$ and, if so, returns M to $\mathcal{A}$. If its decryption oracle returns $\perp$ or if it happens that $\mathsf{Ver}(cpars, pk_b, com, dec) = 0$, $\mathcal{B}_1$ returns $\perp$ to $\mathcal{A}$. We observe that, due to the use of labels, $\mathcal{B}_1$ can always answer $\mathcal{A}$'s decryption queries using its own **Dec** oracle. Indeed, after the challenge phase, $\mathcal{A}$ is disallowed to query the decryption oracle on $(d, C^\star, com^\star)$, for each $d \in \{0,1\}$. This implies that $\mathcal{B}_1$ will *never* query its **Dec** oracle on an input of the form $\big(d, C^\star, com^\star\big)$, with $d \in \{0,1\}$: if $C = C^\star$, it must be the case that $com \neq com^\star = L^\star$.

It is easy to see that, if $\mathcal{B}_1$'s challenger chooses the bit $\beta = 0$, $\mathcal{B}_1$ is playing Game 1 with $\mathcal{A}$. In the situation where $\beta = 1$, $\mathcal{B}_1$ and $\mathcal{A}$ are playing Game 2.

Game 3: In this game, we bring a new change in the generation of the challenge ciphertext $(C^\star, com^\star)$. Now $\mathcal{A}$'s challenger generates a commitment/de-commitment pair with respect to pk_1: more precisely, it commits to pk_1 by computing $(com^\star, dec^\star) \leftarrow_\$ \mathsf{Com}(cpars, pk_1)$. Then, as in Game 2, it computes $C^\star \leftarrow_\$ \mathsf{Enc}(pars, pk_1, M_1^\star \| 0^{|dec^\star|}, L^\star)$, where $L^\star = com^\star$. If $\mathcal{CMT}$ is a hiding commitment, Game 3 is clearly indistinguishable from Game 2. Indeed, it is straightforward to construct a distinguisher $\mathcal{B}_2$ such that $|\Pr[S_3] - \Pr[S_2]| \leq \mathbf{Adv}^{\text{hide}}_{\mathcal{CMT}}(\mathcal{B}_2)$.

Game 4: We bring a final change to the computation of the challenge ciphertext $(C^\star, com^\star)$. Namely, $\mathcal{A}$'s challenger commits to the public key pk_1 by computing $(com^\star, dec^\star) \leftarrow_\$ \mathsf{Com}(cpars, pk_1)$. Then, it computes $C^\star \leftarrow_\$ \mathsf{Enc}(pars, pk_1, M_1^\star \| dec^\star, L^\star)$, where $L^\star = com^\star$.

This game is exactly the real game when $\mathcal{A}$'s challenger chooses the bit $b = 1$ at the beginning of the experiment. There exists a PPT algorithm $\mathcal{B}_3$ such that $|\Pr[S_4] - \Pr[S_3]| \leq \mathbf{Adv}^{\text{ai-atk}}_{\overline{\mathcal{PKE}}}(\mathcal{B}_3)$. Algorithm $\mathcal{B}_3$ is defined exactly as algorithm $\mathcal{B}_1$ in the transition from Game 1 to Game 2. The only differences are: (1) $\mathcal{B}_3$ is challenged on a single public key $pk_0 = pk_1 = pk$; (2) $\overline{M}_0^\star$ and $\overline{M}_1^\star$ are now defined as $\overline{M}_0^\star = M_1 \| 0^{|dec^\star|}$ and $\overline{M}_1^\star = M_1 \| dec^\star$, where $dec^\star$ is the de-commitment information in $(com^\star, dec^\star) \leftarrow_\$ \mathsf{Com}(cpars, pk_1)$. Clearly, if $\mathcal{B}_3$'s challenger encrypts $\overline{M}_0^\star$, $\mathcal{B}_3$ is playing Game 3 with $\mathcal{A}$. Otherwise, $\mathcal{B}_1$ and $\mathcal{A}$ are rather playing Game 4. □

We now prove that the scheme is completely robust.

Proof. We show that the scheme is simultaneously FROB, KROB, and XROB. Let us first assume that $\mathcal{A}$ defeats the FROB property and outputs (C, com) and two private keys sk_0, sk_1 such that, for each $b \in \{0,1\}$, the ciphertext C decrypts to messages $M_b \| dec_b = \mathsf{Dec}(pars, pk_b, sk_b, (C, com), com)$ such that $\mathsf{Ver}(cpars, pk_0, com, dec_0) = 1$ and $\mathsf{Ver}(cpars, pk_1, com, dec_1) = 1$ although $pk_0 \neq pk_1$. Then, the binding property of $\mathcal{CMT}$ is necessarily broken.

Similar arguments apply in the KROB scenario, where the adversary outputs two pairs (M_b, pk_b, r_b), with $b \in \{0,1\}$ and $pk_0 \neq pk_1$, resulting in colliding ciphertexts $C = \overline{\mathsf{Enc}}\big((pars, cpars), pk_b, M_b, r_b\big)$ for $\overline{\mathcal{PKE}}$. It necessarily means that r_b contains incorporated random coins r'_b for which we have $(com_b, dec_b) \leftarrow \mathsf{Com}(cpars, pk_b; r'_b)$ with $com = com_0 = com_1$. This implies

$$\mathsf{Ver}(cpars, pk_0, com, dec_0) = 1 \quad \text{and} \quad \mathsf{Ver}(cpars, pk_1, com, dec_1) = 1,$$

which also contradicts the assumption that $\mathcal{CMT}$ is binding.

Finally, in the XROB case, the adversary outputs $(M_0, pk_0, r_0, C_1, pk_1, sk_1)$ such that $M_0 \neq \perp$ and, if we compute $(C, com) \leftarrow \overline{\mathsf{Enc}}\big((pars, cpars), pk_0, M_0; r_0\big)$ and $M_1 \| dec_1 \leftarrow \mathsf{Dec}(pars, pk_1, sk_1, (C, com), com)$, it holds that $\mathsf{Ver}(cpars, pk_1, com, dec_1) = 1$. In this case, r_0 must also contain random commitment coins such that $(com, dec_0) \leftarrow \mathsf{Com}(cpars, pk_0; r'_0)$ and thus $\mathsf{Ver}(cpars, pk_0, com, dec_0) = 1$. It means that we also end up with openings (pk_b, dec_b) of the same commitment com to distinct messages pk_0 and pk_1. □

J Proof of Theorem 5

Let us first briefly recall that a multi-authority IBE scheme [26] consists of algorithms $\mathcal{IBE} = (\mathcal{IBE}.\mathsf{PG}, \mathcal{IBE}.\mathsf{MPG}, \mathcal{IBE}.\mathsf{KG}, \mathcal{IBE}.\mathsf{Enc}, \mathcal{IBE}.\mathsf{Dec})$, corresponding to cross-TA common parameter generation, master key generation for each TA, user key generation, encryption, and decryption routines respectively. The notion of sID-TAA-CPA security for IBE schemes in the multi-authority setting is formalized in Figure 10. It is shown in [26,27] that applying the original CHK transformation to an sID-TAA-CPA secure IBE scheme provides an AI-CCA-secure PKE. We will prove that our modification of the Boneh–Katz transformation gives the same result.

It is worth noting that sID-TAA-CPA secure IBE schemes are available in the literature. Indeed, a sufficient condition for IBE schemes to satisfy the notion of sID-TAA-CPA security is to have pseudorandom ciphertexts: namely, ciphertexts should be computationally indistinguishable from a sequence of random group elements. Many anonymous IBE schemes (e.g., [19,11,16,5]) have this property.

proc Initialize$(id^\star)$
$pars \leftarrow\!\!\$\ \mathcal{IBE}.\mathsf{PG}$
$(msk_0, mpk_0) \leftarrow\!\!\$\ \mathcal{IBE}.\mathsf{MPG}(pars)$
$(msk_1, mpk_1) \leftarrow\!\!\$\ \mathcal{IBE}.\mathsf{MPG}(pars)$
$b \leftarrow\!\!\$\ \{0,1\}$
Return $(pars, mpk_0, mpk_1)$

proc GetDK(d, id)
If $d \notin \{0,1\}$ Then Return $\perp$
If $id = id^\star$ Then Return $\perp$
$dk \leftarrow\!\!\$\ \mathcal{IBE}.\mathsf{KG}(pars, msk_d, id)$
Return dk

proc LR(M_0, M_1)
$C \leftarrow\!\!\$\ \mathcal{IBE}.\mathsf{Enc}(pars, mpk_b, id^\star, M_b)$
Return C

proc Finalize(b')
Return $(b' = b)$

Fig. 10. Game defining sID-TAA-CPA security for IBE schemes. An adversary is legitimate if it calls **LR** exactly once on two messages of equal lengths.

We now prove Theorem 5.

Proof. The proof uses a sequence of games which is similar to that of [9]. For each i, we call S_i that event that **Finalize** receives the input $b' = 0$ from $\mathcal{A}$ in Game i.

Game 1: is the real game when the challenger's bit is $b = 0$. Namely, the adversary $\mathcal{A}$ interacts with an actual AI-CCA challenger that runs the actual **Setup** procedure. The challenger always runs the **GetDK** procedure by following its exact specification. When the adversary $\mathcal{A}$ sends its unique **LR** query $(M_0^\star, M_1^\star)$, the challenger picks $k^\star \leftarrow\!\!\$\ \{0,1\}^\ell$ at random, computes a commitment/de-commitment pair $(com^\star, dec^\star) \leftarrow\!\!\$\ \mathsf{Com}(cpars, mpk_0 \| k^\star)$ and an IBE ciphertext $C^\star \leftarrow\!\!\$\ \mathcal{IBE}.\mathsf{Enc}\big(mpk_0, com^\star, M_0^\star \| k^\star \| dec^\star\big)$.[10] Then, $\mathcal{A}$ is given $(C^\star, com^\star, tag^\star)$, where $tag^\star = \mathsf{MacGen}_{k^\star}(C^\star)$, as a challenge ciphertext. At the end of the game, $\mathcal{A}$ calls **Finalize** with an input bit $b' \in \{0,1\}$. We observe that, without loss of generality, $k^\star$ and $(com^\star, dec^\star) \leftarrow\!\!\$\ \mathsf{Com}(cpars, mpk_0 \| k^\star)$ can be chosen at the beginning of the game, during the execution of **Setup**.

Game 2: is identical to Game 1, except that the **Dec** oracle now rejects all decryption queries (C, com, tag) such that $com = com^\star$. Clearly, Game 2 is identical to Game 1 until **Dec** rejects a ciphertext that would not have been rejected in Game 1. We distinguish the following cases in this event:

[10] We omit *pars* as an explicit input to various $\mathcal{IBE}$ algorithms to ease notation.

- At some **Dec** query of the form $(C, com^\star, tag)$ with respect to the public key pk_d (with $d \in \{0,1\}$), the execution of algorithm $\mathcal{IBE}.\mathsf{Dec}(mpk_d, \mathcal{IBE}.\mathsf{KG}(msk_d, com^\star), com^\star, C)$ uncovers a message $(M\|k\|dec)$ such that $\mathsf{Ver}(cpars, mpk_d\|k, com^\star, dec) = 1$ and $k \neq k^\star$. We call this event E_2.
- In all decryption queries $(C, com^\star, tag)$, the IBE ciphertext C decrypts to a message of the form $M\|k^\star\|dec$ under the key $dk_{com^\star} = \mathcal{IBE}.\mathsf{KG}(msk_d, com^\star)$. We call this event F_2.

It is clear that $|\Pr[S_2] - \Pr[S_1]| \leq \Pr[E_2] + \Pr[F_2]$. If E_2 occurs, it necessarily means that $\mathcal{A}$ provides its challenger with correct openings $(mpk_0\|k^\star, dec^\star)$ and $(mpk_d\|k, dec)$ of a given commitment $com^\star$ for two distinct messages $mpk_0\|k^\star$ and $mpk_d\|k$. If $\mathcal{TCMT}$ is binding, E_2 only occurs with negligible probability and there exists a PPT algorithm $\mathcal{B}_0$ such that $\Pr[E_2] \leq \mathbf{Adv}^{\text{bind}}_{\mathcal{TCMT}}(\mathcal{B}_0)$.

We thus have to argue that $\Pr[F_2]$ is negligible. To do this, we use the "deferred analysis" technique [18] in the same way as in [9]. Namely, we consider games Game 2.1 and 2.2 where we argue that F_2 occurs with about the same probability as in Game 2 and where it will be much easier to bound $\Pr[F_2]$.

Game 2.1: we modify the generation of the challenge ciphertext. Namely, instead of computing the pair $(com^\star, dec^\star)$ as $(com^\star, dec^\star) \leftarrow_\$ \mathsf{Com}(cpars, mpk_0\|k^\star)$ at the beginning of the game, the challenger computes $(com^\star, dec^\star)$ as a commitment to a random message R and, in the challenge phase, uses the trapdoor td of $\mathcal{TCMT}$ to equivocate $com^\star$ (note that the trapdoor can be used since we do not appeal to the binding property for now) and compute

$$dec' \leftarrow_\$ \mathsf{Equiv}(td, (mpk_0\|k^\star), R, dec^\star)$$

such that $\mathsf{Ver}(cpars, mpk_0\|k^\star, com^\star, dec') = 1$. Then, the challenge ciphertext $(C^\star, com^\star, tag^\star)$ is obtained by computing $C^\star \leftarrow_\$ \mathcal{IBE}.\mathsf{Enc}(mpk_0, com^\star, M_0^\star\|k^\star\|dec')$. Since $\mathcal{TCMT}$ is a trapdoor commitment scheme, this change is purely conceptual since the pair $(com^\star, dec')$ has the same distribution as $(com^\star, dec^\star)$ in Game 2. For this reason, we know that $\Pr[F_{2.1}] = \Pr[F_2]$, where $F_{2.1}$ denotes the equivalent of event F_2 in Game 2.1.

Game 2.2: is like Game 2.1 with the difference that, when the challenge ciphertext $(C^\star, com^\star, tag^\star)$ is generated by the **LR** procedure, $com^\star$, $dec^\star$ and dec' are computed as previously (in such a way that dec' explains $com^\star$ as a commitment to $mpk_b\|k^\star$) but $C^\star$ is now computed as $\mathcal{IBE}.\mathsf{Enc}\big(mpk_0, com^\star, M_0^\star\|0^\lambda\|0^{|dec^\star|}\big)$, where 0^λ and $0^{|dec^\star|}$ denote all-zeroes strings of lengths λ and $|dec^\star|$, respectively. Observe that, in this game, $com^\star$ and $C^\star$ are now completely independent of $k^\star$. In the following, we call $F_{2.2}$ the counterpart of event F_2 in Game 2.2.

We show that any noticeable change in $\mathcal{A}$'s behavior in Games 2.1 and 2.2 would contradict the IND-sID-CPA security (i.e., its sID-TAA-CPA security in the case $mpk_0 = mpk_1$) of the IBE system. There is a PPT algorithm $\mathcal{B}_1$ such that $|\Pr[F_{2.2}] - \Pr[F_{2.1}]| \leq \mathbf{Adv}^{\text{ind-sID-cpa}}_{\mathcal{IBE}}(\mathcal{B}_1)$. Namely, $\mathcal{B}_1$ interacts with its own IND-sID-CPA challenger and emulates $\mathcal{A}$'s challenger. At the outset of the game, $\mathcal{B}_1$ chooses the description $\mathcal{MAC}$ of a MAC and generates $cpars$ by running CPG. It also computes $(com^\star, dec^\star)$ by committing to a random value R and declares $com^\star$ as its target identity to the IND-sID-CPA challenger. When obtaining the common public parameters $pars$ and the master public key mpk of $\mathcal{IBE}$ from its challenger (note that there is only one master public key in the definition of IND-sID-CPA security), it generates a pair $(mpk', msk') \leftarrow_\$ \mathcal{IBE}.\mathsf{MPG}(pars)$ of its own and gives $(pars, cpars, \mathcal{MAC})$ as well as $mpk_0 = mpk$ and $mpk_1 = mpk'$ to $\mathcal{A}$. At this point, $\mathcal{B}_1$ knows mpk_0 and thus equivocates $com^\star$ using the trapdoor td to find a de-commitment dec' such that $\mathsf{Ver}(cpars, mpk_0\|k^\star, com^\star, dec') = 1$. When $\mathcal{A}$ invokes the **LR** oracle on the input of $(M_0^\star, M_1^\star)$, $\mathcal{B}_1$ constructs the messages $\overline{M}_0^\star = M_0^\star\|k^\star\|dec'$, $\overline{M}_1^\star = M_0^\star\|0^\lambda\|0^{|dec^\star|}$ which it sends to its own **LR** oracle in the IND-sID-CPA security game. The latter **LR** oracle replies with $C^\star \leftarrow_\$ \mathcal{IBE}.\mathsf{Enc}(mpk, com^\star, \overline{M}_\beta^\star)$, for some random bit $\beta \in_R \{0,1\}$, and $\mathcal{B}_1$ provides $\mathcal{A}$ with the challenge ciphertext $(C^\star, com^\star, tag^\star)$, where $tag^\star = \mathsf{MacGen}_{k^\star}(C^\star)$. Algorithm $\mathcal{B}_1$ is able to faithfully answer all **Dec** queries with respect to $mpk_1 = mpk'$ since it knows the underlying master secret key msk'. Whenever $\mathcal{A}$

makes a **Dec** query (C, com, tag) involving mpk_0, $\mathcal{B}_1$ queries its own **GetDK** oracle whenever $com \neq com^\star$ to decrypt (when obtaining the answer $M\|k\|dec$, it returns M if $\mathsf{Ver}(cpars, mpk_0\|k, com, dec) = 1$ and $\mathsf{MacVer}_k(C) = 1$ and $\perp$ otherwise). If $com = com^\star$, $\mathcal{B}_1$ checks if $\mathsf{MacVer}_{k^\star}(C, tag) = 1$ and, if so, halts and output 1. If we call $F_{2.2}$ the latter event, it is clear that $|\Pr[F_{2.2}] - \Pr[F_{2.1}]| \leq \mathbf{Adv}^{\text{ind-sID-cpa}}_{\mathcal{IBE}}(\mathcal{B}_1)$.

In Game 2.2, we claim that $\Pr[F_{2.2}] \leq q \cdot \mathbf{Adv}^{\text{suf-cma}}_{\mathcal{MAC}}(\mathcal{B}_2)$, for some efficient algorithm $\mathcal{B}_2$. Indeed, assuming that $F_{2.2}$ occurs with noticeable probability in Game 2.2, a standard technique allows building an algorithm that breaks the strong (one-time) unforgeability of $\mathcal{MAC}$ with probability $\Pr[F_{2.2}]/q$. The forger $\mathcal{B}_2$ simply chooses a random index $i \leftarrow\!\!\$\ \{1, \ldots, q\}$, invokes its MAC generation oracle to compute $tag^\star$ in the challenge ciphertext and outputs the pair (C_i, tag_i) contained in the i-th decryption query.

When combining the above steps, we find that

$$\Pr[F_2] \leq \mathbf{Adv}^{\text{ind-sID-cpa}}_{\mathcal{IBE}}(\mathcal{B}_1) + q \cdot \mathbf{Adv}^{\text{suf-cma}}_{\mathcal{MAC}}(\mathcal{B}_2).$$

Now, we are ready to proceed with Game 3, where we only have to worry about decryption queries (C, com, tag) such that $com \neq com^\star$.

Game 3: we modify the generation of the challenge ciphertext analogously to the transition from Game 2 to Game 2.1. Namely, instead of computing $(com^\star, dec^\star) \leftarrow\!\!\$\ \mathsf{Com}(cpars, mpk_0\|k^\star)$ at the outset of the game, $(com^\star, dec^\star)$ is computed by committing to a random message R. In the challenge phase, the challenger uses the trapdoor of $\mathcal{TCMT}$ to equivocate $com^\star$ and find

$$dec' \leftarrow\!\!\$\ \mathsf{Equiv}(td, (mpk_0\|k^\star), R, dec^\star).$$

The challenge ciphertext $(C^\star, com^\star, tag^\star)$ is computed as

$$C^\star \leftarrow\!\!\$\ \mathcal{IBE}.\mathsf{Enc}(mpk_0, com^\star, M_0^\star\|k^\star\|dec').$$

Since $\mathcal{TCMT}$ is a trapdoor commitment scheme, this change is purely conceptual since the pair $(com^\star, dec')$ has the same distribution as if it were produced by committing to $mpk_0\|k^\star$. For this reason, we know that $\Pr[S_3] = \Pr[S_2]$ although the distribution of $com^\star$ does not depend on $mpk_0\|k^\star$.

Game 4: we change again the generation of the challenge ciphertext. In this game, the pair $(com^\star, dec^\star)$ is again generated by committing to a random value. However, the IBE part $C^\star$ of the challenge ciphertext is now computed as

$$dec' \leftarrow\!\!\$\ \mathsf{Equiv}(td, (mpk_1\|k^\star), R, dec^\star)$$
$$C^\star \leftarrow\!\!\$\ \mathcal{IBE}.\mathsf{Enc}(mpk_1, com^\star, M_1^\star\|k^\star\|dec')$$

To justify the transition from Game 3 to Game 4, we describe a sID-TAA-CPA adversary $\mathcal{B}_3$ against the IBE scheme that "bridges" between the two games in such a way that $|\Pr[S_4] - \Pr[S_3]| \leq \mathbf{Adv}^{\text{sID-taa-cpa}}_{\mathcal{IBE}}(\mathcal{B}_3)$. Concretely, $\mathcal{B}_3$ interacts with its own sID-TAA-CPA challenger and embodies $\mathcal{A}$'s challenger. At the beginning of its interaction with $\mathcal{A}$, $\mathcal{B}_3$ chooses the description $\mathcal{MAC}$ and generates $cpars$ using CPG. It computes $(com^\star, dec^\star)$ by committing to a random value R and announces $com^\star$ as the identity that it wishes to be challenged upon. When obtaining the common public parameters $pars$ and master public keys mpk_0, mpk_1 from its sID-TAA-CPA challenger, it gives $(pars, cpars, \mathcal{MAC})$ as well as mpk_0, mpk_1 to $\mathcal{A}$. At this point, $\mathcal{B}_3$ equivocates $com^\star$ twice to find suitable de-commitments dec'_0, dec'_1 such that $\mathsf{Ver}(cpars, mpk_0\|k^\star, com^\star, dec'_0) = 1$, $\mathsf{Ver}(cpars, mpk_1\|k^\star, com^\star, dec'_1) = 1$. When the AI-CCA adversary $\mathcal{A}$ calls the **LR** oracle with input values $(M_0^\star, M_1^\star)$, $\mathcal{B}_3$ defines messages $\overline{M}_0^\star = M_0^\star\|k^\star\|dec'_0$, $\overline{M}_1^\star = M_1^\star\|k^\star\|dec'_1$ which are sent to $\mathcal{B}_3$'s **LR** oracle in the sID-TAA-CPA security game. The latter oracle returns $C^\star \leftarrow\!\!\$\ \mathcal{IBE}.\mathsf{Enc}(mpk, com^\star, \overline{M}_\beta^\star)$, where $\beta \in_R \{0, 1\}$, and $\mathcal{B}_3$ prepares $\mathcal{A}$'s challenge ciphertext as $(C^\star, com^\star, tag^\star)$, where $tag^\star = \mathsf{MacGen}_{k^\star}(C^\star)$. Whenever $\mathcal{A}$ makes a **Dec** query (C, com, tag) involving

mpk_0 or mpk_1, $\mathcal{B}_3$ queries its own **Dec** oracle when $com \neq com^\star$ (upon receiving the answer $M\|k\|dec$, it proceeds as in step 2 of $\overline{\mathsf{Dec}}$). If $com = com^\star$, $\mathcal{B}_3$ returns $\perp$. By inspection, it can be checked that, if $\mathcal{B}_3$'s challenger chooses $\beta = 0$, $\mathcal{B}_3$ and $\mathcal{A}$ are playing Game 3. If $\beta = 1$, they are playing Game 4. As claimed, it comes that $|\Pr[S_3] - \Pr[S_4]| \leq \mathbf{Adv}^{\text{sID-taa-cpa}}_{\mathcal{IBE}}(\mathcal{B}_3)$.

Game 5: is identical to Game 4 except that the commitment/de-commitment pair $(com^\star, dec^\star)$ is now computed by committing to $mpk_1\|k^\star$ (and $(M_1^\star\|k^\star\|dec^\star)$ is IBE-encrypted in the challenge phase, as previously) instead of equivocating $com^\star$. This change does not affect $\mathcal{A}$'s view since $(com^\star, dec^\star)$ has the same distribution either way. We thus have $\Pr[S_5] = \Pr[S_4]$.

Game 6: is exactly the same as Game 5 but the **Dec** oracle does no longer reject any decryption queries (C, com, tag) such that $com = com^\star$. Instead, the real $\overline{\mathsf{Dec}}$ algorithm is always used. This transition can be justified completely analogously to that between Game 2 and Game 3 (in other words, the probability that one ciphertext gets rejected in one game and not in the previous one is the same) and we can write:

$$|\Pr[S_6] - \Pr[S_5]| \leq \mathbf{Adv}^{\text{bind}}_{\mathcal{TCMT}}(\mathcal{B}_0) + \mathbf{Adv}^{\text{sID-taa-cpa}}_{\mathcal{IBE}}(\mathcal{B}_1) + q \cdot \mathbf{Adv}^{\text{suf-cma}}_{\mathcal{MAC}}(\mathcal{B}_2),$$

for certain PPT algorithms $\mathcal{B}_0, \mathcal{B}_1$, and $\mathcal{B}_2$.

It is easy to see that Game 6 is the real game when the challenger's bit is $b = 1$. By collecting probabilities, the announced result follows.

The proof of complete robustness of the scheme is similar to that of Theorem 4 and omitted. □

K Proof of Theorem 6

Proof. To argue for the AI-CCA security, we prove that, even if the adversary has access to a decryption oracle, ciphertexts are computationally indistinguishable from "dummy" ciphertexts that are statistically independent of the plaintext or the receiver's public key. Since all ciphertexts live in the same space, regardless of the public key, this guarantees AI-CCA security.

Concretely, we show that the adversary cannot distinguish a real game from an ideal game. In the former, he has access to a decryption oracle and, in the challenge phase, obtains a properly generated encryption of the plaintext $M^\star$ of his choice. In the latter, the challenge ciphertext is a dummy ciphertext that has nothing to do with the plaintext chosen by the adversary or the receiver's public key. We prove that, even with the help of a decryption oracle, the adversary $\mathcal{A}$ cannot distinguish the two worlds. The proof uses a sequence of games. For each i, we call S_i the event that the adversary outputs $b' = 1$ at the end of Game i.

Game 1: is the real game with the difference that, in the challenge phase, the adversary is given an encryption of a random plaintext. The adversary is given a public key pk and access to a decryption oracle. In the challenge phase, it outputs a plaintext $M^\star$. The challenger responds by encrypting $M^\star$. At the end of the game, after a second series of decryption queries, the adversary outputs a bit $b' \in \{0, 1\}$.

Game 2: is like Game 1 but we change the generation of the public key pk. Namely, u and v are chosen as

$$u = g^{x_1} \cdot h^{x_2}, \qquad v = g^{y_1} \cdot h^{y_2}$$

using random $x_1, x_2, y_1, y_2 \leftarrow_\$ \mathbb{Z}_p$. If we define $\omega = \log_g(h)$, this implicitly defines $x = x_1 + \omega x_2$ and $y = y_1 + \omega y_2$. As in Game 1, the challenger rejects all decryption queries (C_1, C_2, C_3, tag) for which $C_2 \neq C_1^{x \cdot \tau + y}$. It is clear that these changes are purely conceptual since pk has the same distribution as before. For, this reason, we have $\Pr[S_2] = \Pr[S_1]$.

Game 3: is identical to Game 2 but the challenger rejects all decryption queries (C_1, C_2, C_3, tag) such that $C_1 \neq C_1^\star$ and $\mathsf{TCR}(C_1) = \mathsf{TCR}(C_1^\star)$ (we assume, without loss of generality, that $C_1^\star$ is computed at the beginning of the game). If the adversary $\mathcal{A}$ manages to create a ciphertext that is rejected in Game 3 but would not have been rejected in Game 2, the challenger $\mathcal{B}$ can clearly use $\mathcal{A}$ to break the target collision-resistance of TCR. We can thus write $|\Pr[S_3] - \Pr[S_2]| \leq \mathbf{Adv}^{\mathrm{tcr}}_{\mathsf{TCR}}(\mathcal{B})$.

Game 4: is like Game 3 but the decryption oracle rejects all post-challenge queries (C_1, C_2, C_3, tag) such that $C_3^\star \neq C_3$ and $H(C_3, pk) = H(C_3^\star, pk)$. This time, we have $|\Pr[S_4] - \Pr[S_3]| \leq \mathbf{Adv}^{\mathrm{cr}}_{H}(\mathcal{B})$ and rely on the collision-resistance of H to argue that Game 4 and Game 3 are indistinguishable.

Game 5: we modify the decryption oracle. At each query (C_1, C_2, C_3, tag), the challenger computes $\tau = \mathsf{TCR}(C_1)$, $(K_0, K_1) = \mathsf{KDF}(\Omega)$, where

$$\Omega = \Big(\frac{C_2}{C_1^{\tau \cdot x_1 + y_1}}\Big)^{\frac{1}{\tau \cdot x_2 + y_2}}, \tag{K.1}$$

and $M = \mathsf{D}_{K_0}(C_3)$. If $\mathsf{MacVer}_{K_1}(H(C_3, pk), tag) = 1$, it outputs M. Otherwise, it output $\perp$. We claim that, if KDF is a secure key-derivation function and if $\mathcal{MAC}$ is unforgeable, the modified decryption oracle does not reject a ciphertext that would not have been rejected in Game 5. The technique of [23,18,20] allows proving that $|\Pr[S_5] - \Pr[S_4]| \leq q \cdot \big(\mathbf{Adv}^{\text{suf-cma}}_{\mathcal{MAC}}(\mathcal{B}_1) + \mathbf{Adv}^{\mathrm{kdf}}_{\mathsf{KDF}}(\mathcal{B}_2)\big)$ for some efficient algorithms $\mathcal{B}_1$ and $\mathcal{B}_2$. The argument is as follows. Suppose that a ciphertext (C_1, C_2, C_3, tag) is invalid because $C_2 \neq C_1^{\tau \cdot x + y}$. It means that we have $C_1 = g^r$ and $C_2 = (u^\tau \cdot v)^r \cdot g^{r'}$ for some $r \in \mathbb{Z}_p$ and some $r' \neq 0$. The decryption oracle thus computes $\Omega = h^r \cdot g^{r'/(\tau \cdot x_2 + y_2)}$, which is uniformly random from $\mathcal{A}$'s view. Indeed, it is the product of h^r, which is uniquely determined by C_1, and $g^{r'/(\tau \cdot x_2 + y_2)}$ that is completely independent of $\mathcal{A}$'s view because the public key reveals no information about (x_2, y_2). The same analysis as in [23,18,20] shows that creating an invalid ciphertext that does not get rejected amounts to forging a MAC for a randomly chosen key or breaking the security of the key-derivation function. We also note that the value $\omega = \log_g(h)$ is not used by the decryption oracle anymore.

Game 6: We modify the generation of the challenge ciphertext. Namely, the challenger $\mathcal{B}$ first defines $\eta_1 = g^{r^\star}$, $\eta_2 = h^{r^\star}$ and $\tau^\star = \mathsf{TCR}(\eta_1)$, for a randomly chosen $r^\star \leftarrow\!\!\$\ \mathbb{Z}_p$, and computes

$$C_1^\star = \eta_1, \qquad\qquad C_2^\star = \eta_1^{\tau^\star \cdot x_1 + y_1} \cdot \eta_2^{\tau^\star \cdot x_2 + y_2} \tag{K.2}$$

$(K_0^\star, K_1^\star) = \mathsf{KDF}(\eta_2)$, $C_3^\star = \mathsf{E}_{K_0^\star}(M^\star)$ and $tag^\star = \mathsf{MacGen}_{K_1^\star}(H(C_3^\star, pk))$. This modification does not affect $\mathcal{A}$'s view since $(C_1^\star, C_2^\star, C_3^\star, tag^\star)$ has the same distribution as in Game 5. So, $\Pr[S_6] = \Pr[S_5]$.

Game 7: We change again the generation of the challenge $(C_1^\star, C_2^\star, C_3^\star, tag^\star)$. Namely, η_2 is now chosen at random in $\mathbb{G}$ instead of being defined as $\eta_2 = h^{r^\star}$, where $r^\star = \log_g(\eta_1)$. Under the DDH assumption, this modification should not significantly affect $\mathcal{A}$'s view. It comes that $|\Pr[S_7] - \Pr[S_6]| \leq \mathbf{Adv}^{\mathrm{ddh}}_{\mathbb{G}}(\mathcal{B})$.

Game 8: In this game, we can use the value $\omega = \log_g(h)$ since we are done with the DDH assumption. Here, the decryption oracle rejects again all ciphertexts (C_1, C_2, C_3, tag) for which $C_2 \neq C_1^{\tau \cdot x + y}$, where $\tau = \mathsf{TCR}(C_1)$, $x = x_1 + \omega x_2$ and $y = y_1 + \omega y_2$. For each ciphertext (C_1, C_2, C_3, tag) such that $C_1 = g^r$ and $C_2 = (u^\tau \cdot v)^r \cdot g^{r'}$, with $r' \neq 0$, the decryption oracle of Game 7 derives (K_0, K_1) from

$$\Omega = h^r \cdot g^{r'/(\tau \cdot x_2 + y_2)}.$$

In (K.2), we now have $\eta_1 = g^{r^\star}$ and $\eta_2 = h^{r^\star + r''}$, for some $r^\star \in_R \mathbb{Z}_p$ and $r'' \neq 0$. The challenge ciphertext is thus distributed as

$$C_1^\star = g^{r^\star}, \qquad\qquad C_2^\star = (u^{\tau^\star} \cdot v)^{r^\star} \cdot h^{r'' \cdot (\tau^\star \cdot x_2 + y_2)}, \tag{K.3}$$

and it reveals $h^{r''\cdot(\tau^\star\cdot x_2+y_2)}$ in the information theoretic sense. However, we have $\tau \neq \tau^\star$ unless the failure event of Game 3 occurs, so that $\tau^\star \cdot x_2 + y_2$ and $\tau \cdot x_2 + y_2$ are independent. This means that Ω is uniformly random from $\mathcal{A}$'s view. The only way for $\mathcal{A}$ to create an invalid ciphertext that would be rejected in Game 8 and not in Game 7 is to break the unforgeability of $\mathcal{MAC}$ or the security of the key-derivation function. Using the same arguments as in, e.g., [20], we can thus write the inequality $|\Pr[S_8] - \Pr[S_7]| \leq q \cdot \left(\mathbf{Adv}^{\text{suf-cma}}_{\mathcal{MAC}}(\mathcal{B}_3) + \mathbf{Adv}^{\text{kdf}}_{\mathsf{KDF}}(\mathcal{B}_4)\right)$, for some efficient algorithms $\mathcal{B}_3$ and $\mathcal{B}_4$.

In Game 8, we claim that $C_1^\star$, $C_2^\star$, and η_2 (which is used to derive the keys $(K_0^\star, K_1^\star) = \mathsf{KDF}(\eta_2)$ in the challenge ciphertext) look independent to $\mathcal{A}$. Indeed, from (K.3) and since the value $\tau^\star \cdot x_2 + y_2$ is independent of $\mathcal{A}$'s view, $(C_1^\star, C_2^\star, \eta_2)$ is statistically indistinguishable from a triple of three random and independent group elements.

Game 9: We change again the construction of the challenge ciphertext. Namely, we choose $K_0^\star$ and $K_1^\star$ at random instead of deriving them from η_2. Since KDF is a secure key-derivation function, we have $|\Pr[S_9] - \Pr[S_8]| \leq \mathbf{Adv}^{\text{kdf}}_{\mathsf{KDF}}(\mathcal{B})$.

Game 10: We bring one more change to the computation of the challenge ciphertext $(C_1^\star, C_2^\star, C_3^\star, tag^\star)$. Namely, instead of computing $tag^\star = \mathsf{MacGen}_{K_1^\star}(H(C_1^\star, pk))$, we compute the MAC as $tag^\star = \mathsf{MacGen}_{K_1^\star}(R)$, where $R \leftarrow\!\!\$\ \mathsf{MSp}^{\text{mac}}$ is chosen independently of $C_1^\star$ or pk. Due the indistinguishability property of $\mathcal{MAC}$, however, this change is not noticeable to $\mathcal{A}$ and we have $|\Pr[S_{10}] - \Pr[S_9]| \leq \mathbf{Adv}^{\text{indist}}_{\mathcal{MAC}}(\mathcal{B})$.

Game 11: We introduce a final change in the generation of $(C_1^\star, C_2^\star, C_3^\star, tag^\star)$. Namely, instead of computing $C_3^\star$ as $C_3^\star = \mathsf{E}_{K_0^\star}(M^\star)$, we symmetrically encrypt a random plaintext M' which is chosen independently of $M^\star$. Clearly, any significant change in $\mathcal{A}$'s behavior would contradict the semantic security of the symmetric cipher (E, D) and we can write $|\Pr[S_{11}] - \Pr[S_{10}]| \leq \mathbf{Adv}^{\text{ind-cpa}}_{\mathsf{E},\mathsf{D}}(\mathcal{B})$ for some efficient adversary $\mathcal{B}$ against (E, D).

In Game 11, the challenge ciphertext $(C_1^\star, C_2^\star, C_3^\star, tag^\star)$ is perfectly independent of the public key pk and the message $M^\star$ chosen by $\mathcal{A}$. □

We now prove the second part of the theorem.

Proof. Let us assume that an adversary can break one of the FROB, KROB or XROB notions. We show that it can either find a collision on H or contradict the assumption of $\mathcal{MAC}$ being committing.

We first consider the FROB case and assume that, on input of *pars*, an adversary $\mathcal{A}$ is able to output a ciphertext $C = (C_1, C_2, C_3, tag)$ and two keys pairs (sk, pk), (sk', pk') such that C correctly decrypts under both $sk = (x, y, z)$ and $sk' = (x', y', z')$. If we define $(K_0, K_1) = \mathsf{KDF}(C_1^z)$ and $(K_0', K_1') = \mathsf{KDF}(C_1^{z'})$, we must have $\mathsf{MacVer}_{K_1}(H(C_3, pk), tag) = 1$ and $\mathsf{MacVer}_{K_1'}(H(C_3, pk'), tag) = 1$. It means that either:

- $H(C_3, pk) = H(C_3, pk')$, in which case the collision resistance of H is broken since $pk \neq pk'$.
- $\mathcal{MAC}$ is not a committing MAC because there exist two keys K_1, K_1' and two distinct messages $R = H(C_3, pk)$, $R' = H(C_3, pk')$ such that $\mathsf{MacVer}_{K_1}(R, tag) = 1$ and $\mathsf{MacVer}_{K_1'}(R', tag) = 1$.

It is straightforward that the same arguments apply in KROB and XROB cases. □