Robust Encryption

Michel Abdalla[1], Mihir Bellare[2], and Gregory Neven[3,4]

[1] Departement d'Informatique, École normale supérieure, Paris, France.
Michel.Abdalla@ens.fr ; http://www.di.ens.fr/users/mabdalla
[2] Department of Computer Science & Engineering, University of California San Diego, USA. mihir@cs.ucsd.edu ; http://www.cs.ucsd.edu/users/mihir
[3] Department of Electrical Engineering, Katholieke Universiteit Leuven, Belgium.
[4] IBM Research – Zurich, Switzerland.
nev@zurich.ibm.com ; http://www.neven.org

Abstract. We provide a provable-security treatment of "robust" encryption. Robustness means it is hard to produce a ciphertext that is valid for two different users. Robustness makes explicit a property that has been implicitly assumed in the past. We argue that it is an essential conjunct of anonymous encryption. We show that natural anonymity-preserving ways to achieve it, such as adding recipient identification information before encrypting, fail. We provide transforms that do achieve it, efficiently and provably. We assess the robustness of specific encryption schemes in the literature, providing simple patches for some that lack the property. We present various applications. Our work enables safer and simpler use of encryption.

1 Introduction

This paper provides a provable-security treatment of encryption "robustness." Robustness reflects the difficulty of producing a ciphertext valid under two different encryption keys. The value of robustness is conceptual, "naming" something that has been undefined yet at times implicitly (and incorrectly) assumed. Robustness helps make encryption more mis-use resistant. We provide formal definitions of several variants of the goal; consider and dismiss natural approaches to achieve it; provide two general robustness-adding transforms; test robustness of existing schemes and patch the ones that fail; and discuss some applications.

The definitions. Both the PKE and the IBE settings are of interest and the explication is simplified by unifying them as follows. Associate to each identity an *encryption key*, defined as the identity itself in the IBE case and its (honestly generated) public key in the PKE case. The adversary outputs a pair id_0, id_1 of distinct identities. For strong robustness it also outputs a ciphertext C^*; for weak, it outputs a message M^*, and C^* is defined as the encryption of M^* under the encryption key ek_1 of id_1. The adversary wins if the decryptions of C^* under the decryption keys dk_0, dk_1 corresponding to ek_0, ek_1 are *both* non-$\perp$. Both weak and strong robustness can be considered under chosen plaintext or chosen ciphertext attacks, resulting in four notions (for each of PKE and IBE) that we denote WROB-CPA, WROB-CCA, SROB-CPA, SROB-CCA.

WHY ROBUSTNESS? The primary security requirement for encryption is data-privacy, as captured by notions IND-CPA or IND-CCA [18, 21, 16, 5, 11]. Increasingly, we are also seeing a market for *anonymity*, as captured by notions ANO-CPA and ANO-CCA [4, 1]. Anonymity asks that a ciphertext does not reveal the encryption key under which it was created.

Where you need anonymity, there is a good chance you need robustness too. Indeed, we would go so far as to say that robustness is an essential companion of anonymous encryption. The reason is that without it we would have security without basic communication correctness, likely upsetting our application. This is best illustrated by the following canonical application of anonymous encryption, but shows up also, in less direct but no less important ways, in other applications. A sender wants to send a message to a *particular* target recipient, but, to hide the identity of this target recipient, anonymously encrypts it under her key and broadcasts the ciphertext to a larger group. But as a member of this group I need, upon receiving a ciphertext, to know whether or not I am the target recipient. (The latter typically needs to act on the message.) Of course I can't tell whether the ciphertext is for me just by looking at it since the encryption is anonymous, but decryption should divulge this information. It does, unambiguously, if the encryption is robust (the ciphertext is for me iff my decryption of it is not $\perp$) but otherwise I might accept a ciphertext (and some resulting message) of which I am not the target, creating mis-communication. Natural "solutions," such as including the encryption key or identity of the target recipient in the plaintext before encryption and checking it upon decryption, are, in hindsight, just attempts to add robustness without violating anonymity and, as we will see, don't work.

We were lead to formulate robustness upon revisiting Public key Encryption with Keyword Search (PEKS) [9]. In a clever usage of anonymity, Boneh, Di Crescenzo, Ostrovsky and Persiano (BDOP) [9] showed how this property in an IBE scheme allowed it to be turned into a privacy-respecting communications filter. But Abdalla et. al [1] noted that the BDOP filter could lack *consistency*, meaning turn up false positives. Their solution was to modify the construction. What we observed instead was that consistency would in fact be provided by the *original* construct if the IBE scheme was robust. PEKS consistency turns out to correspond exactly to communication correctness of the anonymous IBE scheme in the sense discussed above. (Because the PEKS messages in the BDOP scheme are the recipients identities from the IBE perspective.) Besides resurrecting the BDOP construct, the robustness approach allows us to obtain the first consistent IND-CCA secure PEKS without random oracles.

Sako's auction protocol [23] is important because it was the first truly practical one to hide the bids of losers. It makes clever use of anonymous encryption for privacy. But we present an attack on fairness whose cause is ultimately a lack of robustness in the anonymous encryption scheme (cf. [2]).

All this underscores a number of the claims we are making about robustness: that it is of conceptual value; that it makes encryption more resistant to

mis-use; that it has been implicitly (and incorrectly) assumed; and that there is value to making it explicit, formally defining and provably achieving it.

Weak versus strong. The above-mentioned auction protocol fails because an adversary can create a ciphertext that decrypts correctly under any decryption key. Strong robustness is needed to prevent this. Weak robustness (of the underlying IBE) will yield PEKS consistency for honestly-encrypted messages but may allow spammers to bypass all filters with a single ciphertext, something prevented by strong robustness. Strong robustness trumps weak for applications and goes farther towards making encryption mis-use resistant. We have defined and considered the weaker version because it can be more efficiently achieved, because some existing schemes achieve it and because attaining it is a crucial first step in our method for attaining strong robustness.

Achieving robustness. As the reader has surely already noted, robustness (even strong) is trivially achieved by appending the encryption key to the ciphertext and checking for it upon decryption. The problem is that the resulting scheme is not anonymous and, as we have seen above, it is exactly for anonymous schemes that robustness is important. Of course, data privacy is important too. Letting AI-ATK = ANO-ATK + IND-ATK for ATK $\in \{\mathrm{CPA}, \mathrm{CCA}\}$, our goal is to achieve AI-ATK + XROB-ATK, ideally for both ATK $\in \{\mathrm{CPA}, \mathrm{CCA}\}$ and X $\in \{\mathrm{W}, \mathrm{S}\}$. This is harder.

Transforms. It is natural to begin by seeking a general transform that takes an arbitrary AI-ATK scheme and returns a AI-ATK + XROB-ATK one. This allows us to exploit known constructions of AI-ATK schemes, supports modular protocol design and also helps understand robustness divorced from the algebra of specific schemes. Furthermore, there is a natural and promising transform to consider. Namely, before encrypting, append to the message some redundancy, such as the recipient encryption key, a constant, or even a hash of the message, and check for its presence upon decryption. (Adding the redundancy before encrypting rather than after preserves AI-ATK.) Intuitively this should provide robustness because decryption with the "wrong" key will result, if not in rejection, then in recovery of a garbled plaintext, unlikely to possess the correct redundancy.

The truth is more complex. We consider two versions of the paradigm and summarize our findings in Fig. 1. In encryption with *unkeyed redundancy*, the redundancy is a function RC of the message and encryption key alone. In this case we show that the method fails spectacularly, not providing even *weak* robustness *regardless of the choice of the function* RC. In encryption with *keyed redundancy*, we allow RC to depend on a key K that is placed in the public parameters of the transformed scheme, out of direct reach of the algorithms of the original scheme. In this form, the method can easily provide weak robustness, and that too with a very simple redundancy function, namely the one that simply returns K.

But we show that even encryption with keyed redundancy fails to provide *strong* robustness. To achieve the latter we have to step outside the encryption with redundancy paradigm. We present a strong robustness conferring transform

that uses a (non-interactive) commitment scheme. For subtle reasons, for this transform to work the starting scheme needs to already be weakly robust. If it isn't already, we can make it so via our weak robustness transform.

In summary, on the positive side we provide a transform conferring weak robustness and another conferring strong robustness. Given any AI-ATK scheme the first transform returns a WROB-ATK + AI-ATK one. Given any AI-ATK + WROB-ATK scheme the second transform returns a SROB-ATK + AI-ATK one. In both cases it is for both ATK = CPA and ATK = CCA and in both cases the transform applies to what we call general encryption schemes, of which both PKE and IBE are special cases, so both are covered.

Robustness of specific schemes. The robustness of existing schemes is important because they might be in use. We ask which specific existing schemes are robust, and, for those that are not, whether they can be made so at a cost lower than that of applying one of our general transforms. There is no reason to expect schemes that are only AI-CPA to be robust since the decryption algorithm may never reject, so we focus on schemes that are known to be AI-CCA. This narrows the field quite a bit. Our findings and results are summarized in Fig. 1.

Canonical AI-CCA schemes in the PKE setting are Cramer-Shoup ($\mathcal{CS}$) in the standard model [15, 4] and $\mathcal{DHIES}$ in the random oracle (RO) model [3, 4]. We show that both are WROB-CCA but neither is SROB-CCA, the latter because encryption with 0 randomness yields a ciphertext valid under any encryption key. We present modified versions $\mathcal{CS}^*, \mathcal{DHIES}^*$ of the schemes that we show are SROB-CCA. Our proof that $\mathcal{CS}^*$ is SROB-CCA builds on the information-theoretic part of the proof of [15]. The result does not need to assume hardness of DDH. It relies instead on pre-image security of the underlying hash function for random range points, something not implied by collision-resistance but seemingly possessed by candidate functions.

In the IBE setting, the CCA version $\mathcal{BF}$ of the RO model Boneh-Franklin scheme is AI-CCA [10, 1], and we show it is SROB-CCA. The standard model Boyen-Waters scheme $\mathcal{BW}$ is AI-CCA [13], and we show it is neither WROB-CCA nor SROB-CCA. It can be made either via our transforms but we don't know of any more direct way to do this.

$\mathcal{BF}$ is obtained via the Fujisaki-Okamoto (FO) transform [17] and $\mathcal{BW}$ via the Canetti-Halevi-Katz (CHK) transform [14, 8]. We can show that neither transform *generically* provides strong robustness. This doesn't say whether they do or not when applied to specific schemes, and indeed the first does for $\mathcal{BF}$ and the second does not for $\mathcal{BW}$.

Summary. Protocol design suggests that designers have the intuition that robustness is naturally present. This seems to be more often right than wrong when considering *weak* robustness of *specific* AI-CCA schemes. Prevailing intuition about *generic* ways to add even weak robustness is wrong, yet we show it can be done by an appropriate tweak of these ideas. Strong robustness is more likely to be absent than present in specific schemes, but important schemes can be patched. Strong robustness can also be added generically, but with more work.

Transform	WROB-ATK	SROB-ATK
Encryption with unkeyed redundancy (EuR)	No	No
Encryption with keyed redundancy (EkR)	Yes	No

Scheme	**setting**	AI-CCA	WROB-CCA	SROB-CCA	**RO model**
$\mathcal{CS}$	PKE	Yes [15, 4]	Yes	No	No
$\mathcal{CS}^*$	PKE	Yes	Yes	Yes	No
$\mathcal{DHIES}$	PKE	Yes [3]	Yes	No	Yes
$\mathcal{DHIES}^*$	PKE	Yes	Yes	Yes	Yes
$\mathcal{BF}$	IBE	Yes [10, 1]	Yes	Yes	Yes
$\mathcal{BW}$	IBE	Yes [13]	No	No	No

Fig. 1. Achieving Robustness. The first table summarizes our findings on the encryption with redundancy transform. "No" means the method fails to achieve the indicated robustness for *all* redundancy functions, while "yes" means there exists a redundancy function for which it works. The second table summarizes robustness results about some specific AI-CCA schemes.

Related work. There is growing recognition that robustness is important in applications and worth defining explicitly, supporting our own claims to this end. In particular the correctness requirement for predicate encryption [20] includes a form of weak robustness and, in recent work concurrent to, and independent of, ours, Hofheinz and Weinreb [19] introduced a notion of *well-addressedness* of IBE schemes that is just like weak robustness except that the adversary gets the IBE master secret key. Neither work considers or achieves strong robustness, and neither treats PKE.

2 Definitions

Notation and conventions. If x is a string then $|x|$ denotes its length, and if S is a set then $|S|$ denotes its size. The empty string is denoted ε. By $a_1 \| \ldots \| a_n$, we denote a string encoding of $a_1, \ldots, a_n$ from which $a_1, \ldots, a_n$ are uniquely recoverable. (Usually, concatenation suffices.) By $a_1 \| \ldots \| a_n \leftarrow a$, we mean that a is parsed into its constituents $a_1, \ldots, a_n$. Similarly, if $a = (a_1, \ldots, a_n)$ then $(a_1, \ldots, a_n) \leftarrow a$ means we parse a as shown. Unless otherwise indicated, an algorithm may be randomized. By $y \xleftarrow{\$} A(x_1, x_2, \ldots)$ we denote the operation of running A on inputs $x_1, x_2, \ldots$ and fresh coins and letting y denote the output. We denote by $[A(x_1, x_2, \ldots)]$ the set of all possible outputs of A on inputs $x_1, x_2, \ldots$. We assume that an algorithm returns $\perp$ if any of its inputs is $\perp$.

Games. Our definitions and proofs use code-based game-playing [6]. Recall that a game —look at Fig. 2 for an example— has an **Initialize** procedure, procedures to respond to adversary oracle queries, and a **Finalize** procedure. A game G is executed with an adversary A as follows. First, **Initialize** executes and its outputs are the inputs to A. Then A executes, its oracle queries being answered

proc Initialize
$(pars, msk) \xleftarrow{\$} \mathsf{PG}$; $b \xleftarrow{\$} \{0,1\}$
$S, T, U, V \leftarrow \emptyset$
Return $pars$

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

proc GetDK(id)
If $id \notin U$ then return $\bot$
If $id \in S$ then return $\bot$
$V \leftarrow V \cup \{id\}$
Return $\mathsf{DK}[id]$

proc Dec(C, id)
If $id \notin U$ then return $\bot$
If $(id, C) \in T$ then return $\bot$
$M \leftarrow \mathsf{Dec}(pars, \mathsf{EK}[id], \mathsf{DK}[id], C)$
Return M

proc LR$(id_0^*, id_1^*, M_0^*, M_1^*)$
If $(id_0^* \notin U) \vee (id_1^* \notin U)$ then return $\bot$
If $(id_0^* \in V) \vee (id_1^* \in V)$ then return $\bot$
If $|M_0^*| \neq |M_1^*|$ then return $\bot$
$C^* \xleftarrow{\$} \mathsf{Enc}(pars, \mathsf{EK}[id_b], M_b^*)$
$S \leftarrow S \cup \{id_0^*, id_1^*\}$
$T \leftarrow T \cup \{(id_0^*, C^*), (id_1^*, C^*)\}$
Return C^*

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

Fig. 2. Game $\mathrm{AI}_{\mathcal{GE}}$ defining AI-ATK security of general encryption scheme $\mathcal{GE} = (\mathsf{PG}, \mathsf{KG}, \mathsf{Enc}, \mathsf{Dec})$.

by the corresponding procedures of G. When A terminates, its output becomes the input to the **Finalize** procedure. The output of the latter, denoted G^A, is called the output of the game, and we let "G^A" denote the event that this game output takes value true. Boolean flags are assumed initialized to false. Games $\mathrm{G}_i, \mathrm{G}_j$ are *identical until* bad if their code differs only in statements that follow the setting of bad to true. Our proofs will use the following.

Lemma 1 [6] *Let* $\mathrm{G}_i, \mathrm{G}_j$ *be identical until* bad *games, and* A *an adversary. Then*

$$\left|\Pr\left[\,\mathrm{G}_i^A\,\right] - \Pr\left[\,\mathrm{G}_j^A\,\right]\right| \;\leq\; \Pr\left[\,\mathrm{G}_j^A \text{ sets } \mathsf{bad}\,\right] . \blacksquare$$

The running time of an adversary is the worst case time of the execution of the adversary with the game defining its security, so that the execution time of the called game procedures is included.

General encryption. We introduce and use general encryption schemes, of which both PKE and IBE are special cases. This allows us to avoid repeating similar definitions and proofs. A *general encryption* (GE) scheme is a tuple $\mathcal{GE} = (\mathsf{PG}, \mathsf{KG}, \mathsf{Enc}, \mathsf{Dec})$ of algorithms. The parameter generation algorithm PG takes no input and returns common parameter $pars$ and a master secret key msk. On input $pars, msk, id$, the key generation algorithm KG produces an encryption key ek and decryption key dk. On inputs $pars, ek, M$, the encryption algorithm Enc produces a ciphertext C encrypting plaintext M. On input $pars, ek, dk, C$, the deterministic decryption algorithm Dec returns either a plaintext message M or $\bot$ to indicate that it rejects. We say that $\mathcal{GE}$ is a public-key encryption (PKE) scheme if $msk = \varepsilon$ and KG ignores its id input. To recover the usual syntax we may in this case write the output of PG as $pars$ rather than $(pars, msk)$ and omit msk, id as inputs to KG. We say that $\mathcal{GE}$ is an identity-based encryption

proc Initialize
$(pars, msk) \xleftarrow{\$} \mathsf{PG}$; $U, V \leftarrow \emptyset$
Return $pars$

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

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

proc Dec(C, id)
If $id \notin U$ then return $\bot$
$M \leftarrow \mathsf{Dec}(pars, \mathsf{EK}[id], \mathsf{DK}[id], C)$
Return M

proc Finalize(M, id_0, id_1) // $\mathrm{WROB}_{\mathcal{GE}}$
If $(id_0 \notin U) \vee (id_1 \notin U)$ then return false
If $(id_0 \in V) \vee (id_1 \in V)$ then return false
If $(id_0 = id_1)$ then return false
$M_0 \leftarrow M$; $C \xleftarrow{\$} \mathsf{Enc}(pars, \mathsf{EK}[id_0], M_0)$
$M_1 \leftarrow \mathsf{Dec}(pars, \mathsf{EK}[id_1], \mathsf{DK}[id_1], C)$
Return $(M_0 \neq \bot) \wedge (M_1 \neq \bot)$

proc Finalize(C, id_0, id_1) // $\mathrm{SROB}_{\mathcal{GE}}$
If $(id_0 \notin U) \vee (id_1 \notin U)$ then return false
If $(id_0 \in V) \vee (id_1 \in V)$ then return false
If $(id_0 = id_1)$ then return false
$M_0 \leftarrow \mathsf{Dec}(pars, \mathsf{EK}[id_0], \mathsf{DK}[id_0], C)$
$M_1 \leftarrow \mathsf{Dec}(pars, \mathsf{EK}[id_1], \mathsf{DK}[id_1], C)$
Return $(M_0 \neq \bot) \wedge (M_1 \neq \bot)$

Fig. 3. Games $\mathrm{WROB}_{\mathcal{GE}}$ and $\mathrm{SROB}_{\mathcal{GE}}$ defining WROB-ATK and SROB-ATK security (respectively) of general encryption scheme $\mathcal{GE} = (\mathsf{PG}, \mathsf{KG}, \mathsf{Enc}, \mathsf{Dec})$. The procedures on the left are common to both games, which differ only in their **Finalize** procedures.

(IBE) scheme if $ek = id$, meaning the encryption key created by KG on inputs $pars, msk, id$ always equals id. To recover the usual syntax we may in this case write the output of KG as dk rather than (ek, dk). It is easy to see that in this way we have recovered the usual primitives. But there are general encryption schemes that are neither PKE nor IBE schemes, meaning the primitive is indeed more general.

Correctness. Correctness of a general encryption scheme $\mathcal{GE} = (\mathsf{PG}, \mathsf{KG}, \mathsf{Enc}, \mathsf{Dec})$ requires that, for all $(pars, msk) \in [\mathsf{PG}]$, all plaintexts M in the underlying message space associated to $pars$, all identities id, and all $(ek, dk) \in [\mathsf{KG}(pars, msk, id)]$, we have $\mathsf{Dec}(pars, ek, dk, \mathsf{Enc}(pars, ek, M)) = M$ with probability one, where the probability is taken over the coins of Enc.

AI-ATK security. Historically, definitions of data privacy (IND) [18, 21, 16, 5, 11] and anonymity (ANON) [4, 1] have been separate. We are interested in schemes that achieve both, so rather than use separate definitions we follow [12] and capture both simultaneously via game $\mathrm{AI}_{\mathcal{GE}}$ of Fig. 2. A cpa adversary is one that makes no **Dec** queries, and a cca adversary is one that might make such queries. The ai-advantage of such an adversary, in either case, is

$$\mathbf{Adv}^{\mathsf{ai}}_{\mathcal{GE}}(A) = 2 \cdot \Pr\left[\,\mathrm{AI}^{A}_{\mathcal{GE}}\,\right] - 1.$$

We will assume an ai-adversary makes only one **LR** query, since a hybrid argument shows that making q of them can increase its ai-advantage by a factor of at most q.

Oracle **GetDK** represents the IBE key-extraction oracle [11]. In the PKE case it is superfluous in the sense that removing it results in a definition that is equivalent up to a factor depending on the number of **GetDK** queries. That's probably why the usual definition has no such oracle. But conceptually, if it is there for IBE, it ought to be there for PKE, and it does impact concrete security.

ROBUSTNESS. Associated to general encryption scheme $\mathcal{GE} = (\mathsf{PG}, \mathsf{KG}, \mathsf{Enc}, \mathsf{Dec})$ are games WROB, SROB of Fig. 3. As before, a cpa adversary is one that makes no **Dec** queries, and a cca adversary is one that might make such queries. The wrob and srob advantages of an adversary, in either case, are

$$\mathbf{Adv}^{\mathrm{wrob}}_{\mathcal{GE}}(A) = \Pr\left[\mathrm{WROB}^{A}_{\mathcal{GE}}\right] \qquad \text{and} \qquad \mathbf{Adv}^{\mathrm{srob}}_{\mathcal{GE}}(A) = \Pr\left[\mathrm{SROB}^{A}_{\mathcal{GE}}\right].$$

The difference between WROB and SROB is that in the former the adversary produces a message M, and C is its encryption under the encryption key of one of the given identities, while in the latter it produces C directly, and may not obtain it as an honest encryption. It is worth clarifying that in the PKE case the adversary does *not* get to choose the encryption (public) keys of the identities it is targeting. These are honestly and independently chosen, in real life by the identities themselves and in our formalization by the games.

3 Robustness failures of encryption with redundancy

A natural privacy-and-anonymity-preserving approach to add robustness to an encryption scheme is to add redundancy before encrypting, and upon decryption reject if the redundancy is absent. Here we investigate the effectiveness of this encryption with redundancy approach, justifying the negative results discussed in Section 1 and summarized in the first table of Fig. 1.

REDUNDANCY CODES AND THE TRANSFORM. A redundancy code $\mathcal{RED} = (\mathsf{RKG}, \mathsf{RC}, \mathsf{RV})$ is a triple of algorithms. The redundancy key generation algorithm RKG generates a key K. On input K and data x the redundancy computation algorithm RC returns redundancy r. Given K, x, and claimed redundancy r, the deterministic redundancy verification algorithm RV returns 0 or 1. We say that $\mathcal{RED}$ is unkeyed if the key K output by RKG is always equal to ε, and keyed otherwise. The correctness condition is that for all x we have $\mathsf{RV}(K, x, \mathsf{RC}(K, x)) = 1$ with probability one, where the probability is taken over the coins of RKG and RC. (We stress that the latter is allowed to be randomized.)

Given a general encryption scheme $\mathcal{GE} = (\mathsf{PG}, \mathsf{KG}, \mathsf{Enc}, \mathsf{Dec})$ and a redundancy code $\mathcal{RED} = (\mathsf{RKG}, \mathsf{RC}, \mathsf{RV})$, the *encryption with redundancy transform* associates to them the general encryption scheme $\overline{\mathcal{GE}} = (\overline{\mathsf{PG}}, \overline{\mathsf{KG}}, \overline{\mathsf{Enc}}, \overline{\mathsf{Dec}})$ whose algorithms are shown on the left side of Fig. 5. Note that the transform has the first of our desired properties, namely that it preserves AI-ATK. Also if $\mathcal{GE}$ is a PKE scheme then so is $\overline{\mathcal{GE}}$, and if $\mathcal{GE}$ is an IBE scheme then so is $\overline{\mathcal{GE}}$, which means the results we obtain here apply to both settings.

Fig. 4 shows example redundancy codes for the transform. With the first, $\overline{\mathcal{GE}}$ is identical to $\mathcal{GE}$, so that the counterexample below shows that AI-CCA

RKG	$\mathsf{RC}(K, ek\|M)$	$\mathsf{RV}(K, ek\|M, r)$
Return $K \leftarrow \varepsilon$	Return ε	Return 1
Return $K \leftarrow \varepsilon$	Return 0^k	Return $(r = 0^k)$
Return $K \leftarrow \varepsilon$	Return ek	Return $(r = ek)$
Return $K \leftarrow \varepsilon$	$L \xleftarrow{\$} \{0,1\}^k$; Return $L\|H(L, ek\|M)$	$L\|h \leftarrow r$; Return $(h = H(L, ek\|M))$
Return $K \xleftarrow{\$} \{0,1\}^k$	Return K	Return $(r = K)$
Return $K \xleftarrow{\$} \{0,1\}^k$	Return $H(K, ek\|M)$	Return $(r = H(K, ek\|M))$

Fig. 4. Examples of redundancy codes, where the data x is of the form $ek\|M$. The first four are unkeyed and the last two are keyed.

does not imply WROB-CPA. The second and third rows show redundancy equal to a constant or the encryption key as examples of (unkeyed) redundancy codes. The fourth row shows a code that is randomized but still unkeyed. The hash function H could be a MAC or a collision resistant function. The last two are keyed redundancy codes, the first the simple one that just always returns the key, and the second using a hash function. Obviously, there are many other examples.

SROB FAILURE. We show that encryption with redundancy fails to provide strong robustness for *all* redundancy codes, whether keyed or not. More precisely, we show that for any redundancy code $\mathcal{RED}$ and both ATK $\in \{\mathrm{CPA}, \mathrm{CCA}\}$, there is an AI-ATK encryption scheme $\mathcal{GE}$ such that the scheme $\overline{\mathcal{GE}}$ resulting from the encryption-with-redundancy transform applied to $\mathcal{GE}, \mathcal{RED}$ is not SROB-CPA. We build $\mathcal{GE}$ by modifying a given AI-ATK encryption scheme $\mathcal{GE}^* = (\mathsf{PG}, \mathsf{KG}, \mathsf{Enc}^*, \mathsf{Dec}^*)$. Let l be the number of coins used by RC, and let $\mathsf{RC}(x; \omega)$ denote the result of executing RC on input x with coins $\omega \in \{0,1\}^l$. Let M^* be a function that given *pars* returns a point in the message space associated to *pars* in $\mathcal{GE}^*$. Then $\mathcal{GE} = (\mathsf{PG}, \mathsf{KG}, \mathsf{Enc}, \mathsf{Dec})$ where the new algorithms are shown on the bottom right side of Fig. 5. The reason we used 0^l as coins for RC here is that Dec is required to be deterministic.

Our first claim is that the assumption that $\mathcal{GE}^*$ is AI-ATK implies that $\mathcal{GE}$ is too. Our second claim, that $\overline{\mathcal{GE}}$ is not SROB-CPA, is demonstrated by the following attack. For a pair id_0, id_1 of distinct identities of its choice, the adversary A, on input $(pars, K)$, begins with queries $ek_0 \xleftarrow{\$} \mathbf{GetEK}(id_0)$ and $ek_1 \xleftarrow{\$} \mathbf{GetEK}(id_1)$. It then creates ciphertext $C \leftarrow 0 \,\|\, K$ and returns (id_0, id_1, C). We claim that $\mathbf{Adv}^{\mathrm{srob}}_{\overline{\mathcal{GE}}}(A) = 1$. Letting dk_0, dk_1 denote the decryption keys corresponding to ek_0, ek_1 respectively, the reason is the following. For both $b \in \{0,1\}$, the output of $\mathsf{Dec}(pars, ek_b, dk_b, C)$ is $M^*(pars)\|r_b(pars)$ where $r_b(pars) = \mathsf{RC}(K, ek_b\|M^*(pars); 0^l)$. But the correctness of $\mathcal{RED}$ implies that $\mathsf{RV}(K, ek_b\|M^*(pars), r_b(pars)) = 1$ and hence $\overline{\mathsf{Dec}}((pars, K), ek_b, dk_b, C)$ returns $M^*(pars)$ rather than $\perp$.

WROB FAILURE. We show that encryption with redundancy fails to provide even *weak* robustness for all *unkeyed* redundancy codes. This is still a powerful

Algorithm $\overline{\mathsf{PG}}$
$(pars, msk) \xleftarrow{\$} \mathsf{PG}$; $K \xleftarrow{\$} \mathsf{RKG}$
Return $((pars, K), msk)$

Algorithm $\overline{\mathsf{KG}}((pars, K), msk, id)$
$(ek, dk) \xleftarrow{\$} \mathsf{KG}(pars, msk, id)$
Return ek

Algorithm $\overline{\mathsf{Enc}}((pars, K), ek, M)$
$r \xleftarrow{\$} \mathsf{RC}(K, ek \| M)$
$C \xleftarrow{\$} \mathsf{Enc}(pars, ek, M \| r)$
Return C

Algorithm $\overline{\mathsf{Dec}}((pars, K), ek, dk, C)$
$M \| r \leftarrow \mathsf{Dec}(pars, ek, dk, C)$
If $\mathsf{RV}(K, ek \| M, r) = 1$ then return M
Else return $\perp$

Algorithm $\mathsf{Enc}(pars, ek, M)$
$C \xleftarrow{\$} \mathsf{Enc}^*(pars, ek, M)$
Return C

Algorithm $\mathsf{Dec}(pars, ek, dk, C)$
$M \leftarrow \mathsf{Dec}^*(pars, ek, dk, C)$
If $M = \perp$ then
  $M \leftarrow M^*(pars) \| \mathsf{RC}(\varepsilon, ek \| M^*(pars); 0^l)$
Return M

Algorithm $\mathsf{Enc}(pars, ek, M)$
$C^* \xleftarrow{\$} \mathsf{Enc}^*(pars, ek, M)$
Return $1 \| C^*$

Algorithm $\mathsf{Dec}(pars, ek, dk, C)$
$b \| C^* \leftarrow C$
If $b = 1$ then return $\mathsf{Dec}^*(pars, ek, dk, C^*)$
Else return $M^*(pars) \| \mathsf{RC}(C^*, ek \| M^*(pars); 0^l)$

Fig. 5. Left: Transformed scheme for the encryption with redundancy paradigm. **Top Right**: Counterexample for WROB. **Bottom Right**: Counterexample for SROB.

negative result because many forms of redundancy that might intuitively work, such the first four of Fig. 4, are included. More precisely, we claim that for any unkeyed redundancy code $\mathcal{RED}$ and both ATK $\in \{\mathrm{CPA}, \mathrm{CCA}\}$, there is an AI-ATK encryption scheme $\mathcal{GE}$ such that the scheme $\overline{\mathcal{GE}}$ resulting from the encryption-with-redundancy transform applied to $\mathcal{GE}, \mathcal{RED}$ is not WROB-CPA. We build $\mathcal{GE}$ by modifying a given AI-ATK + WROB-CPA encryption scheme $\mathcal{GE}^* = (\mathsf{PG}, \mathsf{KG}, \mathsf{Enc}^*, \mathsf{Dec}^*)$. With notation as above, the new algorithms for the scheme $\mathcal{GE} = (\mathsf{PG}, \mathsf{KG}, \mathsf{Enc}, \mathsf{Dec})$ are shown on the top right side of Fig. 5.

Our first claim is that the assumption that $\mathcal{GE}^*$ is AI-ATK implies that $\mathcal{GE}$ is too. Our second claim, that $\overline{\mathcal{GE}}$ is not WROB-CPA, is demonstrated by the following attack. For a pair id_0, id_1 of distinct identities of its choice, the adversary A, on input $(pars, \varepsilon)$, makes queries $ek_0 \xleftarrow{\$} \mathbf{GetEK}(id_0)$ and $ek_1 \xleftarrow{\$} \mathbf{GetEK}(id_1)$ and returns $(id_0, id_1, M^*(pars))$. We claim that $\mathbf{Adv}^{\mathrm{wrob}}_{\overline{\mathcal{GE}}}(A)$ is high. Letting dk_1 denote the decryption key for ek_1, the reason is the following. Let $r_0 \xleftarrow{\$} \mathsf{RC}(\varepsilon, ek_0 \| M^*(pars))$ and $C \xleftarrow{\$} \mathsf{Enc}(pars, ek_0, M^*(pars) \| r_0)$. The assumed WROB-CPA security of $\mathcal{GE}^*$ implies that $\mathsf{Dec}(pars, ek_1, dk_1, C)$ is most probably $M^*(pars) \| r_1(pars)$ where $r_1(pars) = \mathsf{RC}(\varepsilon, ek_1 \| M^*(pars); 0^l)$. But the correctness of $\mathcal{RED}$ implies that $\mathsf{RV}(\varepsilon, ek_1 \| M^*(pars), r_1(pars)) = 1$ and hence $\overline{\mathsf{Dec}}((pars, \varepsilon), ek_1, dk_1, C)$ returns $M^*(pars)$ rather than $\perp$.

4 Transforms that work

We present a transform that confers weak robustness and another that confers strong robustness. They preserve privacy and anonymity, work for PKE as well

as IBE, and for CPA as well as CCA. In both cases the security proofs surface some delicate issues. Besides being useful in its own right, the weak robustness transform is a crucial step in obtaining strong robustness, so we begin there.

WEAK ROBUSTNESS TRANSFORM. We saw that encryption-with-redundancy fails to provide even weak robustness if the redundancy code is unkeyed. Here we show that if the redundancy code is keyed, even in the simplest possible way where the redundancy is just the key itself, the transform does provide weak robustness, turning any AI-ATK secure general encryption scheme into an AI-ATK + WROB-ATK one, for both ATK $\in \{\mathrm{CPA}, \mathrm{CCA}\}$.

The transformed scheme encrypts with the message a key K placed in the public parameters. In more detail, the *weak robustness transform* associates to a given general encryption scheme $\mathcal{GE} = (\mathsf{PG}, \mathsf{KG}, \mathsf{Enc}, \mathsf{Dec})$ and integer parameter k, representing the length of K, the general encryption scheme $\overline{\mathcal{GE}} = (\overline{\mathsf{PG}}, \overline{\mathsf{KG}}, \overline{\mathsf{Enc}}, \overline{\mathsf{Dec}})$ whose algorithms are depicted in Fig. 6. Note that if $\mathcal{GE}$ is a PKE scheme then so is $\overline{\mathcal{GE}}$ and if $\mathcal{GE}$ is an IBE scheme then so is $\overline{\mathcal{GE}}$, so that our results, captured by Theorem 2 below, cover both settings.

The intuition for the weak robustness of $\overline{\mathcal{GE}}$ is that the $\mathcal{GE}$ decryption under one key, of an encryption of $\overline{M}\|K$ created under another key, cannot, by the assumed AI-ATK security of $\mathcal{GE}$, reveal K, and hence the check will fail. This is pretty much right for PKE, but the delicate issue is that for IBE, information about K can enter via the identities, which in this case are the encryption keys and are chosen by the adversary as a function of K. The AI-ATK security of $\mathcal{GE}$ is no protection against this. We show however that this can be dealt with by making K sufficiently longer than the identities.

Theorem 2 *Let $\mathcal{GE} = (\mathsf{PG}, \mathsf{KG}, \mathsf{Enc}, \mathsf{Dec})$ be a general encryption scheme with identity space $\{0,1\}^n$, and let $\overline{\mathcal{GE}} = (\overline{\mathsf{PG}}, \overline{\mathsf{KG}}, \overline{\mathsf{Enc}}, \overline{\mathsf{Dec}})$ be the general encryption scheme resulting from applying the weak robustness transform to $\mathcal{GE}$ and integer parameter k. Then*

1. AI-ATK*: Let A be an ai-adversary against $\overline{\mathcal{GE}}$. Then there is an ai-adversary B against $\mathcal{GE}$ such that $\mathbf{Adv}^{\mathrm{ai}}_{\overline{\mathcal{GE}}}(A) = \mathbf{Adv}^{\mathrm{ai}}_{\mathcal{GE}}(B)$. Adversary B inherits the query profile of A and has the same running time as A. If A is a cpa adversary then so is B.*
2. WROB-ATK*: Let A be a wrob adversary against $\overline{\mathcal{GE}}$ with running time t, and let $\ell = 2n + \lceil \log_2(t) \rceil$. Then there is an ai-adversary B against $\mathcal{GE}$ such that $\mathbf{Adv}^{\mathrm{wrob}}_{\overline{\mathcal{GE}}}(A) \leq \mathbf{Adv}^{\mathrm{ai}}_{\mathcal{GE}}(B) + 2^{\ell-k}$. Adversary B inherits the query profile of A and has the same running time as A. If A is a cpa adversary then so is B.* ∎

The first part of the theorem implies that if $\mathcal{GE}$ is AI-ATK then $\overline{\mathcal{GE}}$ is AI-ATK as well. The second part of the theorem implies that if $\mathcal{GE}$ is AI-ATK and k is chosen sufficiently larger than $2n + \lceil \log_2(t) \rceil$ then $\overline{\mathcal{GE}}$ is WROB-ATK. In both cases this is for both ATK $\in \{\mathrm{CPA}, \mathrm{CCA}\}$. The theorem says it directly for CCA, and for CPA by the fact that if A is a cpa adversary then so is B. When we say that B inherits the query profile of A we mean that for every oracle that B has, if A has an oracle of the same name and makes q queries to it, then

Algorithm $\overline{\mathsf{PG}}$
$(pars, msk) \xleftarrow{\$} \mathsf{PG}$
$K \xleftarrow{\$} \{0,1\}^k$
Return $((pars, K), msk)$

Algorithm $\overline{\mathsf{Enc}}((pars, K), ek, \overline{M})$
$C \xleftarrow{\$} \mathsf{Enc}(pars, ek, \overline{M} \| K))$
Return C

Algorithm $\overline{\mathsf{KG}}((pars, K), msk, id)$
$(ek, dk) \xleftarrow{\$} \mathsf{KG}(pars, msk, id)$
Return (ek, dk)

Algorithm $\overline{\mathsf{Dec}}((pars, K), ek, dk, C)$
$M \leftarrow \mathsf{Dec}(pars, ek, dk, C)$
If $M = \perp$ then return $\perp$
$\overline{M} \| K^* \leftarrow M$
If $(K = K^*)$ then return $\overline{M}$
Else Return $\perp$

Fig. 6. General encryption scheme $\overline{\mathcal{GE}} = (\overline{\mathsf{PG}}, \overline{\mathsf{KG}}, \overline{\mathsf{Enc}}, \overline{\mathsf{Dec}})$ resulting from applying our weak-robustness transform to general encryption scheme $\mathcal{GE} = (\mathsf{PG}, \mathsf{KG}, \mathsf{Enc}, \mathsf{Dec})$ and integer parameter k.

this is also the number B makes. The proof of the first part of the theorem is straightforward and is omitted. The proof of the second part is given in [2]. It is well known that collision-resistant hashing of identities preserves AI-ATK and serves to make them of fixed length [7] so the assumption that the identity space is $\{0,1\}^n$ rather than $\{0,1\}^*$ is not really a restriction. In practice we might hash with SHA256 so that $n = 256$, and, assuming $t \leq 2^{128}$, setting $k = 768$ would make $2^{\ell - k} = 2^{-128}$.

Commitment schemes. Our strong robustness transform will use commitments. A commitment scheme is a 3-tuple $\mathcal{CMT} = (\mathsf{CPG}, \mathsf{Com}, \mathsf{Ver})$. The parameter generation algorithm CPG returns public parameters *cpars*. The committal algorithm Com takes *cpars* and data x as input and returns a commitment *com* to x along with a decommittal key *dec*. The deterministic verification algorithm Ver takes $cpars, x, com, dec$ as input and returns 1 to indicate that accepts or 0 to indicate that it rejects. Correctness requires that, for any $x \in \{0,1\}^*$, any $cpars \in [\mathsf{CPG}]$, and any $(com, dec) \in [\mathsf{Com}(cpars, x)]$, we have that $\mathsf{Ver}(cpars, x, com, dec) = 1$ with probability one, where the probability is taken over the coins of Com. We require the scheme to have the *uniqueness* property, which means that for any $x \in \{0,1\}^*$, any $cpars \in [\mathsf{CPG}]$, and any $(com, dec) \in [\mathsf{Com}(cpars, x)]$ it is the case that $\mathsf{Ver}(cpars, x, com^*, dec) = 0$ for all $com^* \neq com$. In most schemes the decommittal key is the randomness used by the committal algorithm and verification is by re-applying the committal function, which ensures uniqueness. The advantage measures $\mathbf{Adv}^{\mathrm{hide}}_{\mathcal{CMT}}(A)$ and $\mathbf{Adv}^{\mathrm{bind}}_{\mathcal{CMT}}(A)$, referring to the standard hiding and binding properties, are recalled in [2]. We refer to the corresponding notions as HIDE and BIND.

The strong robustness transform. The idea is for the ciphertext to include a commitment to the encryption key. The commitment is *not* encrypted, but the decommittal key is. In detail, given a general encryption scheme $\mathcal{GE} = (\mathsf{PG}, \mathsf{KG}, \mathsf{Enc}, \mathsf{Dec})$ and a commitment scheme $\mathcal{CMT} = (\mathsf{CPG}, \mathsf{Com}, \mathsf{Ver})$ the *strong robustness transform* associates to them the general encryption scheme $\overline{\mathcal{GE}} = (\overline{\mathsf{PG}}, \overline{\mathsf{KG}}, \overline{\mathsf{Enc}}, \overline{\mathsf{Dec}})$ whose algorithms are depicted in Fig. 7. Note that if $\mathcal{GE}$ is a

Algorithm $\overline{\mathsf{PG}}$
$(pars, msk) \xleftarrow{\$} \mathsf{PG}$
$cpars \xleftarrow{\$} \mathsf{CPG}$
Return $((pars, cpars), msk)$

Algorithm $\overline{\mathsf{Enc}}((pars, cpars), ek, \overline{M})$
$(com, dec) \xleftarrow{\$} \mathsf{Com}(cpars, ek)$
$C \xleftarrow{\$} \mathsf{Enc}(pars, ek, \overline{M} \| dec))$
Return (C, com)

Algorithm $\overline{\mathsf{KG}}((pars, cpars), msk, id)$
$(ek, dk) \xleftarrow{\$} \mathsf{KG}(pars, msk, id)$
Return (ek, dk)

Algorithm $\overline{\mathsf{Dec}}((pars, cpars), ek, dk, (C, com))$
$M \leftarrow \mathsf{Dec}(pars, ek, dk, C)$
If $M = \perp$ then return $\perp$
$\overline{M} \| dec \leftarrow M$
If $(\mathsf{Ver}(cpars, ek, com, dec) = 1)$ then return $\overline{M}$
Else Return $\perp$

Fig. 7. General encryption scheme $\overline{\mathcal{GE}} = (\overline{\mathsf{PG}}, \overline{\mathsf{KG}}, \overline{\mathsf{Enc}}, \overline{\mathsf{Dec}})$ resulting from applying our strong robustness transform to general encryption scheme $\mathcal{GE} = (\mathsf{PG}, \mathsf{KG}, \mathsf{Enc}, \mathsf{Dec})$ and commitment scheme $\mathcal{CMT} = (\mathsf{CPG}, \mathsf{Com}, \mathsf{Ver})$.

PKE scheme then so is $\overline{\mathcal{GE}}$ and if $\mathcal{GE}$ is an IBE scheme then so is $\overline{\mathcal{GE}}$, so that our results, captured by the Theorem 3, cover both settings.

In this case the delicate issue is not the robustness but the AI-ATK security of $\overline{\mathcal{GE}}$ in the CCA case. Intuitively, the hiding security of the commitment scheme means that a $\overline{\mathcal{GE}}$ ciphertext does not reveal the encryption key. As a result, we would expect AI-ATK security of $\overline{\mathcal{GE}}$ to follow from the commitment hiding security and the assumed AI-ATK security of $\mathcal{GE}$. This turns out not to be true, and demonstrably so, meaning there is a counterexample to this claim. (See below.) What we show is that the claim is true if $\mathcal{GE}$ is additionally WROB-ATK. This property, if not already present, can be conferred by first applying our weak robustness transform.

Theorem 3 *Let $\mathcal{GE} = (\mathsf{PG}, \mathsf{KG}, \mathsf{Enc}, \mathsf{Dec})$ be a general encryption scheme, and let $\overline{\mathcal{GE}} = (\overline{\mathsf{PG}}, \overline{\mathsf{KG}}, \overline{\mathsf{Enc}}, \overline{\mathsf{Dec}})$ be the general encryption scheme resulting from applying the strong robustness transform to $\mathcal{GE}$ and commitment scheme $\mathcal{CMT} = (\mathsf{CPG}, \mathsf{Com}, \mathsf{Ver})$. Then*

1. AI-ATK*: Let A be an ai-adversary against $\overline{\mathcal{GE}}$. Then there is a wrob adversary W against $\mathcal{GE}$, a hiding adversary H against $\mathcal{CMT}$ and an ai-adversary B against $\mathcal{GE}$ such that*

$$\mathbf{Adv}^{\mathrm{ai}}_{\overline{\mathcal{GE}}}(A) \;\le\; 2 \cdot \mathbf{Adv}^{\mathrm{wrob}}_{\mathcal{GE}}(W) + 2 \cdot \mathbf{Adv}^{\mathrm{hide}}_{\mathcal{CMT}}(H) + 3 \cdot \mathbf{Adv}^{\mathrm{ai}}_{\mathcal{GE}}(B) \; .$$

Adversaries W, B inherit the query profile of A, and adversaries W, H, B have the same running time as A. If A is a cpa adversary then so are W, B.

2. SROB-ATK*: Let A be a srob adversary against $\overline{\mathcal{GE}}$ making q **GetEK** queries. Then there is a binding adversary B against $\mathcal{CMT}$ such that*

$$\mathbf{Adv}^{\mathrm{srob}}_{\overline{\mathcal{GE}}}(A) \;\le\; \mathbf{Adv}^{\mathrm{bind}}_{\mathcal{CMT}}(B) + \binom{q}{2} \cdot \mathbf{Coll}_{\mathcal{GE}} \; .$$

Adversary B has the same running time as A. ∎

The first part of the theorem implies that if $\mathcal{GE}$ is AI-ATK and WROB-ATK and $\mathcal{CMT}$ is HIDE then $\overline{\mathcal{GE}}$ is AI-ATK, and the second part of the theorem implies that if $\mathcal{CMT}$ is BIND secure and $\mathcal{GE}$ has low encryption key collision probability then $\overline{\mathcal{GE}}$ is SROB-ATK. In both cases this is for both ATK $\in \{\text{CPA}, \text{CCA}\}$. We remark that the proof shows that in the CPA case the WROB-ATK assumption on $\mathcal{GE}$ in the first part is actually not needed. The encryption key collision probability $\mathbf{Coll}_{\mathcal{GE}}$ of $\mathcal{GE}$ is defined as the maximum probability that $ek_0 = ek_1$ in the experiment where we let $(pars, msk) \xleftarrow{\$} \mathsf{PG}$ and then let $(ek_0, dk_0) \xleftarrow{\$} \mathsf{KG}(pars, msk, id_0)$ and $(ek_1, dk_1) \xleftarrow{\$} \mathsf{KG}(pars, msk, id_1)$, where the maximum is over all distinct identities id_0, id_1. The collision probability is zero in the IBE case since $ek_0 = id_0 \neq id_1 = ek_1$. It is easy to see that $\mathcal{GE}$ being AI implies $\mathbf{Coll}_{\mathcal{GE}}$ is negligible, so asking for low encryption key collision probability is in fact not an extra assumption. (For a general encryption scheme the adversary needs to have hardwired the identities that achieve the maximum, but this is not necessary for PKE because here the probability being maximized is the same for all pairs of distinct identities.) The reason we made the encryption key collision probability explicit is that for most schemes it is unconditionally low. For example, when $\mathcal{GE}$ is the ElGamal PKE scheme, it is $1/|\mathbb{G}|$ where $\mathbb{G}$ is the group being used. Proofs of both parts of the theorem are in [2].

The need for weak-robustness. As we said above, the AI-ATK security of $\overline{\mathcal{GE}}$ won't be implied merely by that of $\mathcal{GE}$. (We had to additionally assume that $\mathcal{GE}$ is WROB-ATK.) Here we justify this somewhat counter-intuitive claim. This discussion is informal but can be turned into a formal counterexample. Imagine that the decryption algorithm of $\mathcal{GE}$ returns a fixed string of the form $(\hat{M}, \hat{dec})$ whenever the wrong key is used to decrypt. Moreover, imagine $\mathcal{CMT}$ is such that it is easy, given $cpars, x, dec$, to find com so that $\mathsf{Ver}(cpars, x, com, dec) = 1$. (This is true for any commitment scheme where dec is the coins used by the Com algorithm.) Consider then the AI-ATK adversary A against the transformed scheme that that receives a challenge ciphertext (C^*, com^*) where $C^* \leftarrow \mathsf{Enc}(pars, \mathsf{EK}[id_b], M^* \| dec^*)$ for hidden bit $b \in \{0, 1\}$. It then creates a commitment $\hat{com}$ of $\mathsf{EK}[id_1]$ with opening information $\hat{dec}$, and queries $(C^*, \hat{com})$ to be decrypted under $\mathsf{DK}[id_0]$. If $b = 0$ this query will probably return $\perp$ because $\mathsf{Ver}(cpars, \mathsf{EK}[id_0], \hat{com}, dec^*)$ is unlikely to be 1, but if $b = 1$ it returns $\hat{M}$, allowing A to determine the value of b. The weak robustness of $\mathcal{GE}$ rules out such anomalies.

5 A SROB-CCA version of Cramer-Shoup

Let $\mathbb{G}$ be a group of prime order p, and H: $\mathsf{Keys}(H) \times \mathbb{G}^3 \to \mathbb{G}$ a family of functions. We assume $\mathbb{G}, p, H$ are fixed and known to all parties. Fig. 8 shows the Cramer-Shoup (CS) scheme and the variant $\mathcal{CS}^*$ scheme where $\mathbf{1}$ denotes the identity element of $\mathbb{G}$. The differences are boxed. Recall that the CS scheme was shown to be IND-CCA in [15] and ANO-CCA in [4]. However, for any message $M \in \mathbb{G}$ the ciphertext $(\mathbf{1}, \mathbf{1}, M, \mathbf{1})$ in the CS scheme decrypts to M under *any*

Algorithm PG
$K \xleftarrow{\$} \mathsf{Keys}(H)$; $g_1 \xleftarrow{\$} \mathbb{G}^*$; $w \xleftarrow{\$} \mathbb{Z}_p^*$; $g_2 \leftarrow g_1^w$; Return (g_1, g_2, K)

Algorithm $\mathsf{KG}(g_1, g_2, K)$
$x_1, x_2, y_1, y_2, z_1, z_2 \xleftarrow{\$} \mathbb{Z}_p$; $e \leftarrow g_1^{x_1} g_2^{x_2}$; $f \leftarrow g_1^{y_1} g_2^{y_2}$; $h \leftarrow g_1^{z_1} g_2^{z_2}$
Return $((e, f, h), (x_1, x_2, y_1, y_2, z_1, z_2))$

Algorithm $\mathsf{Enc}((g_1, g_2, K), (e, f, h), M)$
$u \xleftarrow{\$} \mathbb{Z}_p^{\boxed{*}}$; $a_1 \leftarrow g_1^u$; $a_2 \leftarrow g_2^u$; $b \leftarrow h^u$; $c \leftarrow b \cdot M$; $v \leftarrow H(K, (a_1, a_2, c))$; $d \leftarrow e^u f^{uv}$
Return (a_1, a_2, c, d)

Algorithm $\mathsf{Dec}((g_1, g_2, K), (e, f, h), (x_1, x_2, y_1, y_2, z_1, z_2), C)$
$(a_1, a_2, c, d) \leftarrow C$; $v \leftarrow H(K, (a_1, a_2, c))$; $M \leftarrow c \cdot a_1^{-z_1} a_2^{-z_2}$
If $d \neq a_1^{x_1 + y_1 v} a_2^{x_2 + y_2 v}$ Then $M \leftarrow \perp$
$\boxed{\text{If } a_1 = \mathbf{1} \text{ Then } M \leftarrow \perp}$
Return M

Fig. 8. The original CS scheme [15] does not contain the boxed code while the variant $\mathcal{CS}^*$ does. Although not shown above, the decryption algorithm in both versions always checks to ensure that the ciphertext $C \in \mathbb{G}^4$. The message space is $\mathbb{G}$.

pars, *pk*, and *sk*, meaning in particular that the scheme is not even SROB-CPA. The modified scheme $\mathcal{CS}^*$ —which continues to be IND-CCA and ANO-CCA— removes this pathological case by having Enc choose the randomness u to be non-zero —Enc draws u from $\mathbb{Z}_p^*$ while the CS scheme draws it from $\mathbb{Z}_p$— and then having Dec reject (a_1, a_2, c, d) if $a_1 = \mathbf{1}$. This thwarts the attack, but is there any other attack? We show that there is not by proving that $\mathcal{CS}^*$ is actually SROB-CCA. Our proof of robustness relies only on the security — specifically, pre-image resistance— of the hash family H: it does not make the DDH assumption. Our proof uses ideas from the information-theoretic part of the proof of [15].

We say that a family H: $\mathsf{Keys}(H) \times \mathsf{Dom}(H) \to \mathsf{Rng}(H)$ of functions is *pre-image resistant* if, given a key K and a *random* range element v^*, it is computationally infeasible to find a pre-image of v^* under $H(K, \cdot)$. The notion is captured formally by the following advantage measure for an adversary I:

$$\mathbf{Adv}_H^{\text{pre-img}}(I) = \Pr\left[H(K, x) = v^* \ : \ K \xleftarrow{\$} \mathsf{Keys}(H) \,;\, v^* \xleftarrow{\$} \mathsf{Rng}(H) \,;\, x \xleftarrow{\$} I(K, v^*) \right] .$$

Pre-image resistance is not implied by the standard notion of one-wayness, since in the latter the target v^* is the image under $H(K, \cdot)$ of a random domain point, which may not be a random range point. However, it seems like a fairly mild assumption on a practical cryptographic hash function and is implied by the notion of "everywhere pre-image resistance" of [22], the difference being that, for the latter, the advantage is the maximum probability over all $v^* \in \mathsf{Rng}(H)$. We now claim the following.

Theorem 4 *Let B be an adversary making two **GetEK** queries, no **GetDK** queries and at most $q-1$ **Dec** queries, and having running time t. Then we can construct an adversary I such that*

$$\mathbf{Adv}^{\mathrm{srob}}_{\mathcal{CS}^*}(A) \;\le\; \mathbf{Adv}^{\text{pre-img}}_{H}(I) + \frac{2q+1}{p}\;. \tag{1}$$

Furthermore, the running time of I is $t + q \cdot O(t_{\mathrm{exp}})$ where t_{exp} denotes the time for one exponentiation in $\mathbb{G}$.

Since $\mathcal{CS}^*$ is a PKE scheme, the above automatically implies security even in the presence of multiple **GetEK** and **GetDK** queries as required by game $\mathrm{SROB}_{\mathcal{CS}^*}$. Thus the theorem implies that $\mathcal{CS}^*$ is SROB-CCA if H is pre-image resistant. A detailed proof of Theorem 4 is in [2]. Here we sketch some intuition.

We begin by conveniently modifying the game interface. We replace B with an adversary A that gets input $(g_1, g_2, K), (e_0, f_0, h_0), (e_1, f_1, h_1)$ representing the parameters that would be input to B and the public keys returned in response to B's two **GetEK** queries. Let $(x_{01}, x_{02}, y_{01}, y_{02}, z_{01}, z_{02})$ and $(x_{11}, x_{12}, y_{11}, y_{12}, z_{11}, z_{12})$ be the corresponding secret keys. The decryption oracle takes (only) a ciphertext and returns its decryption under *both* secret keys, setting a WIN flag if these are both non-$\perp$. Adversary A no longer needs an output, since it can win via a **Dec** query.

Suppose A makes a **Dec** query (a_1, a_2, c, d). Then the code of the decryption algorithm Dec from Fig. 8 tells us that, for this to be a winning query, it must be that

$$d \;=\; a_1^{x_{01}+y_{01}v} a_2^{x_{02}+y_{02}v} \;=\; a_1^{x_{11}+y_{11}v} a_2^{x_{12}+y_{12}v}$$

where $v = H(K, (a_1, a_2, c))$. Letting $u_1 = \log_{g_1}(a_1), u_2 = \log_{g_2}(a_2)$ and $s = \log_{g_1}(d)$, we have

$$s \;=\; u_1(x_{01}+y_{01}v) + wu_2(x_{02}+y_{02}v) \;=\; u_1(x_{11}+y_{11}v) + wu_2(x_{12}+y_{12}v) \tag{2}$$

However, even acknowledging that A knows little about $x_{b1}, x_{b2}, y_{b1}, y_{b2}$ ($b \in \{0,1\}$) through its **Dec** queries, it is unclear why Equation (2) is prevented by pre-image resistance —or in fact any property short of being a random oracle— of the hash function H. In particular, there seems no way to "plant" a target v^* as the value v of Equation (2) since the adversary controls u_1 and u_2. However, suppose now that $a_2 = a_1^w$. (We will discuss later why we can assume this.) This implies $wu_2 = wu_1$ or $u_2 = u_1$ since $w \neq 0$. Now from Equation (2) we have

$$u_1(x_{01} + y_{01}v) + wu_1(x_{02} + y_{02}v) - u_1(x_{11} + y_{11}v) - wu_1(x_{12} + y_{12}v) \;=\; 0\;.$$

We now see the value of enforcing $a_1 \neq 1$, since this implies $u_1 \neq 0$. After canceling u_1 and re-arranging terms, we have

$$v(y_{01} + wy_{02} - y_{11} - wy_{12}) + (x_{01} + wx_{02} - x_{11} - wx_{12}) \;=\; 0\;. \tag{3}$$

Given that $x_{b1}, x_{b2}, y_{b1}, y_{b2}$ ($b \in \{0,1\}$) and w are chosen by the game, there is at most one solution v (modulo p) to Equation (3). We would like now to design I

so that on input K, v^* it chooses $x_{b1}, x_{b2}, y_{b1}, y_{b2}$ ($b \in \{0,1\}$) so that the solution v to Equation (3) is v^*. Then (a_1, a_2, c) will be a pre-image of v^* which I can output.

To make all this work, we need to resolve two problems. The first is why we may assume $a_2 = a_1^w$ —which is what enables Equation (3)— given that a_1, a_2 are chosen by A. The second is to properly design I and show that it can simulate A correctly with high probability. To solve these problems, we consider, as in [15], a modified check under which decryption, rather than rejecting when $d \neq a_1^{x_1+y_1 v} a_2^{x_2+y_2 v}$, rejects when $a_2 \neq a_1^w$ or $d \neq a_1^{x+yv}$, where $x = x_1 + w x_2$, $y = y_1 + w y_2$, $v = H(K, (a_1, a_2, c))$ and (a_1, a_2, c, d) is the ciphertext being decrypted. See [2].

Acknowledgments

First and third authors were supported in part by the European Commission through the ICT Program under Contract ICT-2007-216646 ECRYPT II. First author was supported in part by the French ANR-07-SESU-008-01 PAMPA Project. Second author was supported in part by NSF grants CNS-0627779 and CCF-0915675. Third author was supported in part by a Postdoctoral Fellowship from the Research Foundation – Flanders (FWO – Vlaanderen) and by the European Community's Seventh Framework Programme project PrimeLife (grant agreement no. 216483).

We thank Chanathip Namprempre, who declined our invitation to be a co-author, for her participation and contributions in the early stage of this work.

References

1. M. Abdalla, M. Bellare, D. Catalano, E. Kiltz, T. Kohno, T. Lange, J. Malone-Lee, G. Neven, P. Paillier, and H. Shi. Searchable encryption revisited: Consistency properties, relation to anonymous IBE, and extensions. *Journal of Cryptology*, 21(3):350–391, July 2008.
2. M. Abdalla, M. Bellare, and G. Neven. Robust encryption. Cryptology ePrint Archive, 2009. Full version of this paper, `http://eprint.iacr.org/`.
3. M. Abdalla, M. Bellare, and P. Rogaway. The oracle Diffie-Hellman assumptions and an analysis of DHIES. In D. Naccache, editor, *CT-RSA 2001*, volume 2020 of *LNCS*, pages 143–158. Springer, Apr. 2001.
4. M. Bellare, A. Boldyreva, A. Desai, and D. Pointcheval. Key-privacy in public-key encryption. In C. Boyd, editor, *ASIACRYPT 2001*, volume 2248 of *LNCS*, pages 566–582. Springer, Dec. 2001.
5. M. Bellare, A. Desai, D. Pointcheval, and P. Rogaway. Relations among notions of security for public-key encryption schemes. In H. Krawczyk, editor, *CRYPTO'98*, volume 1462 of *LNCS*, pages 26–45. Springer, Aug. 1998.
6. M. Bellare and P. Rogaway. The security of triple encryption and a framework for code-based game-playing proofs. In S. Vaudenay, editor, *EUROCRYPT 2006*, volume 4004 of *LNCS*, pages 409–426. Springer, May / June 2006.

7. D. Boneh and X. Boyen. Efficient selective-ID secure identity based encryption without random oracles. In C. Cachin and J. Camenisch, editors, *EUROCRYPT 2004*, volume 3027 of *LNCS*, pages 223–238. Springer, May 2004.
8. D. Boneh, R. Canetti, S. Halevi, and J. Katz. Chosen-ciphertext security from identity-based encryption. *SIAM Journal on Computing*, 36(5):915–942, 2006.
9. D. Boneh, G. Di Crescenzo, R. Ostrovsky, and G. Persiano. Public key encryption with keyword search. In C. Cachin and J. Camenisch, editors, *EUROCRYPT 2004*, volume 3027 of *LNCS*, pages 506–522. Springer, May 2004.
10. D. Boneh and M. K. Franklin. Identity-based encryption from the Weil pairing. In J. Kilian, editor, *CRYPTO 2001*, volume 2139 of *LNCS*, pages 213–229. Springer, Aug. 2001.
11. D. Boneh and M. K. Franklin. Identity based encryption from the Weil pairing. *SIAM Journal on Computing*, 32(3):586–615, 2003.
12. D. Boneh, C. Gentry, and M. Hamburg. Space-efficient identity based encryption without pairings. In *48th FOCS*, pages 647–657. IEEE Computer Society Press, Oct. 2007.
13. X. Boyen and B. Waters. Anonymous hierarchical identity-based encryption (without random oracles). In C. Dwork, editor, *CRYPTO 2006*, volume 4117 of *LNCS*, pages 290–307. Springer, Aug. 2006.
14. R. Canetti, S. Halevi, and J. Katz. Chosen-ciphertext security from identity-based encryption. In C. Cachin and J. Camenisch, editors, *EUROCRYPT 2004*, volume 3027 of *LNCS*, pages 207–222. Springer, May 2004.
15. R. Cramer and V. Shoup. Design and analysis of practical public-key encryption schemes secure against adaptive chosen ciphertext attack. *SIAM Journal on Computing*, 33(1):167–226, 2003.
16. D. Dolev, C. Dwork, and M. Naor. Nonmalleable cryptography. *SIAM Journal on Computing*, 30(2):391–437, 2000.
17. E. Fujisaki and T. Okamoto. Secure integration of asymmetric and symmetric encryption schemes. In M. J. Wiener, editor, *CRYPTO'99*, volume 1666 of *LNCS*, pages 537–554. Springer, Aug. 1999.
18. S. Goldwasser and S. Micali. Probabilistic encryption. *Journal of Computer and System Sciences*, 28(2):270–299, 1984.
19. D. Hofheinz and E. Weinreb. Searchable encryption with decryption in the standard model. Cryptology ePrint Archive, Report 2008/423, 2008. `http://eprint.iacr.org/`.
20. J. Katz, A. Sahai, and B. Waters. Predicate encryption supporting disjunctions, polynomial equations, and inner products. In N. P. Smart, editor, *EUROCRYPT 2008*, volume 4965 of *LNCS*, pages 146–162. Springer, Apr. 2008.
21. C. Rackoff and D. R. Simon. Non-interactive zero-knowledge proof of knowledge and chosen ciphertext attack. In J. Feigenbaum, editor, *CRYPTO'91*, volume 576 of *LNCS*, pages 433–444. Springer, Aug. 1992.
22. P. Rogaway and T. Shrimpton. Cryptographic hash-function basics: Definitions, implications, and separations for preimage resistance, second-preimage resistance, and collision resistance. In B. K. Roy and W. Meier, editors, *FSE 2004*, volume 3017 of *LNCS*, pages 371–388. Springer, Feb. 2004.
23. K. Sako. An auction protocol which hides bids of losers. In H. Imai and Y. Zheng, editors, *PKC 2000*, volume 1751 of *LNCS*, pages 422–432. Springer, Jan. 2000.