Enhanced Chosen-Ciphertext Security and Applications

Dana Dachman-Soled[1] Georg Fuchsbauer[2] Payman Mohassel[3]
Adam O'Neill[4]

Abstract

We introduce and study a new notion of *enhanced* chosen-ciphertext security (ECCA) for public-key encryption. Loosely speaking, in ECCA, when the decryption oracle returns a plaintext to the adversary, it also provides *coins* under which the returned plaintext encrypts to the queried ciphertext (when they exist). Our results mainly concern the case where such coins can also be recovered efficiently. We provide constructions of ECCA encryption from adaptive trapdoor functions as defined by Kiltz *et al.* (EUROCRYPT 2010), resulting in ECCA encryption from standard number-theoretic assumptions. We then give two applications of ECCA encryption: (1) We use it as a unifying concept in showing equivalence of adaptive trapdoor functions and tag-based adaptive trapdoor functions (namely, we show that both primitives are equivalent to ECCA encryption), resolving a main open question of Kiltz *et al.* (2) We show that ECCA encryption can be used to securely realize an approach to public-key encryption with non-interactive opening (PKENO) suggested by Damgård and Thorbek (EUROCRYPT 2007), resulting in new and practical PKENO schemes quite different from those in prior work. We believe our results indicate that ECCA is an intriguing notion that may prove useful in further work.

[1] Department of Computer Science, Microsoft Research New England, 1 Memorial Drive, Cambridge, MA 02142. Email: `dadachma@microsoft.com`. URL: `http://cs.columbia.edu/dglasner`.

[2] Department of Computer Science, University of Bristol, MVB, Woodland Road, Bristol BS8 1UB, United Kingdom. Email: `georg@cs.bris.ac.uk`. URL: `http://www.di.ens.fr/~fuchsbau`.

[3] Department of Computer Science, University of Calgary, 2500 University Dr. NW, Calgary, AB. Email: `pmohasse@cpsc.ucalgary.ca`. URL: `http://pages.cpsc.ucalgary.ca/~pmohasse`.

[4] Department of Computer Science, Boston University, 111 Cummington St., Boston, MA 02215. Email: `amoneill@bu.edu`. URL: `http://cs-people.bu.edu/amoneill`. Supported in part by NSF grants 0546614, 0831281, 1012910, and 1012798.

Contents

1 Introduction

This paper studies a new notion of security for public-key encryption we call *enhanced* chosen-ciphertext security (ECCA). Recall that in the standard formulation of CCA security [34], the adversary, given a public-key pk, must guess which of the two possible messages its challenge ciphertext c encrypts, while being allowed to query a decryption oracle on any ciphertext c' different from c. Very informally, our "enhancement" is that the decryption oracle, when queried on a ciphertext c', returns not only a decrypted plaintext m' but also coins r' such that m' encrypts to c' under pk using coins r' (in the case that the ciphertext is in the range of the encryption algorithm). ECCA is largely motivated by the related concept of *randomness-recovering encryption* (as highlighted in e.g. [33]), meaning the decryption procedure also recovers such r' efficiently. Interestingly, however, not every randomness-recovering encryption scheme is ECCA secure, because randomness-recovery refers to the behavior of an honest sender whereas an ECCA adversary can make malicious decryption queries. (In fact, in Section 3 we show how to design a randomness-recovering CCA PKE scheme that is *provably insecure* against an ECCA adversary.[1])

Besides being interesting in its own right, we find that ECCA plays a fundamental role in contexts where randomness-recovering encryption is important, in particular the contexts of adaptive trapdoor functions [28] and public-key encryption with non-interactive opening [13]. (We also believe ECCA will find future applications given the importance of randomness-recovering encryption.)

Below we describe our results concerning ECCA in more detail; for a pictorial summary, see Figure 1.

1.1 ECCA-Secure Constructions and Relation to Adaptive Trapdoor Functions

Background. A foundational line of work in cryptography examines the relations between public-key encryption (PKE) and trapdoor functions (TDFs). A specific question studied in recent works [33, 35, 28] is: What is the minimal security assumption on a trapdoor function needed to obtain a construction of CCA-secure PKE? Currently, the minimal assumption (in terms of black-box implications) is that of *adaptivity* [28]. Intuitively, adaptivity is a form of CCA security for TDFs, asking that the TDF remain one-way even when the adversary may query an inversion oracle on points other than its challenge.[2]

Results. We find that the notion of ECCA security for encryption is very natural to consider in this context. Indeed, while [28] showed that ATDFs imply CCA secure encryption, we show that ATDFs are *equivalent* to randomness-recovering ECCA-secure encryption. This helps us better understand the power of ATDFs and gives us constructions of ECCA secure encryption from a variety of standard number-theoretic assumptions as per [28]. We furthermore show that "tag-based" ATDFs as defined in [28] are also equivalent to randomness-recovering ECCA-secure encryption, a corollary being ATDFs and tag-based ATDFs are themselves equivalent, which resolves a foundational question left open by [28]. We next discuss how we obtain these results in more detail.

Technical approach. It is easy to see that from randomness-recovering ECCA-secure encryption we can get an ATDF, simply by viewing the input to the trapdoor function as consisting of the message and the coins for the encryption scheme. The other direction is more challenging. To see why, recall that [28] constructs CCA-secure encryption from a ATDFs in the following manner: first, they construct, a *one-bit* CCA-secure encryption scheme and then apply a transform of Myers and shelat [31] from one-bit to many-bit CCA-secure encryption. But the one-bit scheme of [28] — which works by re-sampling a domain point x until the hardcore bit of x equals the message — is *not* randomness-recovering, since decryption does not recover the "thrown away" x's.[3] Furthermore, even if it was, the construction of many-bit

[1]Extending this idea into a black-box separation between the two notions is an interesting open question.

[2]In subsequent work, Wee [36] showed that a weaker notion of adaptivity for *trapdoor relations* suffices; however, as this is not an assumption on trapdoor *functions* it does not yield randomness-recovering encryption and won't be useful for our results.

[3]It is randomness-recovering in a weaker sense that the decryption algorithm can re-sample coins consistent with the message, but this will in particular not be sufficient for showing equivalence of adaptive trapdoor functions and the tag-

CCA-secure encryption in [31] does not seem to preserve either randomness-recovery or ECCA-security of the one-bit scheme. (In particular, the construction of [31] uses an "inner," "q-bounded" non-malleable encryption scheme, and we do not know how to construct such a scheme that has analogous "enhanced" security.)

We solve both these problems at once via a novel application of *detectable* CCA (DCCA) security, introduced very recently by Hohenberger *et al.* [26]. Informally, DCCA is defined relative to a "detecting" function $\mathcal{F}$ that determines whether two ciphertexts are related; in the DCCA experiment, the adversary is not allowed to ask for decryptions of ciphertexts related to the challenge ciphertext according to $\mathcal{F}$. The work of [26] shows how to construct a CCA-secure encryption from a DCCA-secure one. In particular, bit-by-bit encryption using a 1-bit CCA secure encryption scheme is easily seen to be DCCA secure, is the main application considered in [26]. (It thus encompasses the earlier work of [31] and builds on the techniques of the latter.) Our novelty is that we construct a DCCA secure scheme from ATDFs that also uses bit-by-bit encryption, but where the underlying one-bit encryption scheme is *not* CCA secure—namely, we use a "naïve" one-bit scheme from ATDFs that simply XOR's the message bit with a hardcore bit of the ATDF. The benefit is that this one-bit scheme *is* randomness recovering. We show that in fact the resulting bit-by-bit scheme now satisfies a notion of DCCA with analogous "enhanced" security. Then, we show that (in contrast to [31]) the construction of CCA from DCCA in [26] preserves both this enhanced security as well as randomness recovery — that is, if we start with a randomness-recovering, enhanced DCCA secure scheme, we end up with a randomness-recovering ECCA secure one. We thus get a randomness-recovering ECCA secure scheme from ATDFs as desired. We note that a by-product of our results is to show that, complementing [31], one-bit encryption is also complete for randomness-recovering CCA and ECCA.

We note that it is much easier to construct a randomness-recovering ECCA-secure scheme from a *tag-based* ATDFs; in fact, we show that the construction of CCA-secure encryption from tag-based ATDFs given in [28], which uses the BCHK transform [7], already works. Indeed, the apparent extra power of tag-based ATDFs makes it surprising that they are equivalent to (non tag-based) ATDFs.

We also point out that part of what makes the above construction challenging is that the ATDF only has a single hardcore bit. Given a linear number of hardcore bits (see [28] for some instantiations), one can use the KEM/DEM paradigm to obtain more efficient constructions of ECCA PKE from ATDFs. Surprisingly, the standard assumption that the DEM component is CCA-secure (or ECCA) is no longer sufficient, but we show that it is enough to assume that the DEM is a one-time *authenticated encryption* (See Appendix B).

1.2 Applications to Public-Key Encryption with Non-Interactive Opening

Background. Public-key encryption with non-interactive opening (PKENO), introduced by Damgård an Thorbeck [14] and studied in detail by [13, 17, 18], allows a receiver to non-interactively prove to anyone that a ciphertext c decrypts to a message m. As discussed by the above-mentioned works, PKENO has applications to multiparty computation (*e.g.*, auctions and elections), secure message transmission, group signatures, and more. But despite numerous applications, such schemes have been difficult to realize. Secure constructions of PKENO currently exist from identity-based encryption [13] and robust non-interactive threshold encryption [18].

An alternative approach. We show that ECCA encryption can be used to securely realize an simple and natural approach to PKENO originally suggested by [14] but not made to work in the literature. The basic idea is to use a randomness-recovering encryption scheme and have the receiver provide the recovered coins as the proof. However, several issues need to be addressed for this approach to work. One problem already discussed in [18, Section 4.1] is that there must also be a way for the receiver to prove the claimed behavior of the decryption algorithm on ciphertexts that might not even be an output of the

based variant.

encryption algorithm, and for which no underlying coins necessarily exist. (Note that such ciphertexts may or may not decrypt to $\perp$ in general.) Moreover, we point out that the encryption scheme *must be ECCA secure* (which was not even defined in prior work); standard chosen-ciphertext security is not enough here, because here the adversary in the corresponding PKENO security game has the ability to see random coins underlying ciphertexts of its choosing. We now describe our results in more detail.

PKENO-compatible ECCA encryption. First, we formalize a notion of *PKENO-compatible ECCA-secure encryption*, for which we can overcome the above problems. There are two requirements for such a scheme. The first is that it is what we call *partial-randomness* recovering. This means that there is a partial-randomness recovery algorithm that, informally, given the secret key does not necessarily recover *all* random coins underlying a ciphertext, but just enough to check that the ciphertext is an encryption of a claimed message. However, this should also be true for ciphertexts *outside* the range of the encryption algorithm (but which do not decrypt to $\perp$).[4] The second requirement is that it has what we call *ciphertext verifiability*, meaning one can check without the secret key (but possibly with the help of some partial random coins) whether the decryption of a ciphertext is $\perp$. Note that ECCA security of such schemes is defined with respect to the partial-randomness recovery algorithm. See Section 6.1 for the formal definitions, which are rather delicate.

Given such a PKENO-compatible ECCA scheme, we can safely use the message and the partial randomness as the non-interactive opening of the ciphertext.

PKENO-compatible ECCA encryption from NIZK. We next show a generic construction of PKENO-compatible ECCA secure encryption from any (partial) randomness recovering ECCA encryption scheme, namely by adding a non-interactive zero-knowledge (NIZK) proof to a ciphertext that there exist some underlying message and random coins. (Indeed, the idea of adding such a proof of well-formedness to achieve PKENO] comes from [14, 18], although not in connection with ECCA.) We show that in order for the proof to go through, however, the NIZK needs to be simulation-sound.

We also show that the same idea works to construct an analogous notion of PKENO-compatible ECCA *tag-based* encryption scheme from a (partial) randomness recovering ECCA-secure tag-based encryption scheme. One can then efficiently transform a PKENO-compatible tag-based scheme into a standard PKENO using either of the two transforms from [7]. (Interestingly, for the more-efficient transform based on symmetric-key primitives to work, we crucially exploit that ciphertext verifiability as defined above can use partial random coins; the scheme is not "publicly verifiable.") As explained next, our most efficient instantiations follow the tag-based approach.

Efficient schemes from tag-based ATDFs. While it is possible to construct NIZKs efficiently in the random-oracle model (which we prefer to avoid), the only known efficient instantiation of NIZKs without random oracles to date are Groth-Sahai (GS) proofs [22], which allow to prove a certain class of statements over bilinear groups. (Moreover, as shown in [30], GS proofs can be made simulation-sound at a moderate price.) In order to take advantage of this instantiation we need a scheme for which the statement we need to prove falls in this class. We show that this is the case for a certain tag-based encryption scheme from the decision-linear assumption (DLIN). This scheme is built from a tag-based ATDF, which is itself built by instantiating a construction of [28] using the DLIN-based lossy and all-but-one TDFs of Freeman *et al.* [16]. A novelty of our scheme is that the statement we need to prove is that a range point of the tag-based ATDF has a pre-image which is a fundamentally *weaker* statement than one guaranteeing well-formedness of the ciphertext (as used in our generic construction), but (perhaps surprisingly) still suffices in the PKENO construction. See Section 6.2 for details.

Although our DLIN-based instantiation avoids using generic NIZKs or random oracles, it is probably still not efficient enough for practical use. To obtain such a scheme we consider tag-based ATDFs with an additional property we call *range verifiability*. Intuitively, range verifiability allows us to check the same

[4]For example, consider a randomness-recovering scheme which always outputs ciphertexts whose last bit is "0," but whose decryption algorithm ignores this last bit. Then clearly we can still recover the randomness underlying ciphertexts whose last bit is "1" despite the fact that such ciphertexts are outside the range of the encryption algorithm.

$$\begin{tikzcd}
& \text{PKENO} & \\
\text{ATDF} \arrow[r, shift left] & \boxed{\text{RR-ECCA PKE}} \arrow[l, shift left] \arrow[r, shift left] \arrow[u, "\text{+ ciphertext verifiability}"] \arrow[d, dashed, shift left] & \text{TB-ATDF} \arrow[l, shift left] \\
& \text{CCA PKE} \arrow[u, shift left, "?"] &
\end{tikzcd}$$

Figure 1: Relations between various primitives studied in this paper. Dashed implications are trivial. Implications with question mark are unknown. "RR-CCA PKE" is randomness-recovering chosen-ciphertext secure PKE. "ATDF" is adaptive trapdoor function. "TB-ATDF" is tag-based adaptive trapdoor function. "PKENO" is public-key encryption with non-interactive opening.

property that the Groth-Sahai statement in our above construction is intended to prove, hence eliminating the need for any NIZK proof at all. We show that the tag-based ATDF constructed in [28] from the instance-independent RSA assumption (II-RSA) has this property. Although the assumption is quite strong, the resulting PKENO scheme is extremely practical: it requires only one 512-bit exponentiation to encrypt, whereas the previous DLIN-based scheme of [18] requires several (though it is based on a more standard assumption). We note that the resulting scheme (assuming we use the more-efficient version of the BCHK [7] transform) is also fully randomness recovering, showing that it is in fact possible to achieve fully (rather than partial) randomness recovering PKENO-compatible ECCA encryption.

1.3 Related Work

ECCA is similar in spirit to *coin-revealing selective opening attack* (SOA-C) [8, 15, 2, 6]. In the latter setting, there are say n ciphertexts encrypting related (but unknown) messages under independent random coins, and the adversary requests the plaintexts and random coins corresponding to some subset of them; the question is whether the "unopened" ciphertexts remain secure. However, it seems to us that SOA-C is neither implied by, nor implies, ECCA. On the one hand, in ECCA the adversary does not "open" ciphertexts whose messages are sampled jointly with the challenge message, but, on the other hand, in SOA-C the adversary does not "open" maliciously-formed ciphertexts (this is true even when considering SOA-C in context of CCA security as in [23, 24]). It is an interesting question whether ECCA has any applications in the domain of SOA-C.

An analogue of ECCA has been previously defined for *commitment schemes* by Canetti *et al.* [10], which they call CCA-secure commitments, in the context of secure multiparty computation. These are commitment schemes that remain secure when the adversary has access to an *unbounded decommitment oracle* that it can call on commitments other than the challenge. They are interested in such schemes that are interactive but in the plain model, meaning there are no public keys. Thus, our setting seems incomparable (as we disallow interaction but allow public keys). However, we view their work as supporting the claim that ECCA is a natural notion of security to consider for encryption.

Other variants of CCA-security for encryption considered before include *replayable* CCA security [9], *constrained* CCA security [25], and *detectable* CCA security [26]. Notably, these are all *relaxations* of CCA security, whereas we consider a strengthening. Another strengthening of CCA security previously considered is *plaintext awareness* [5, 1, 4].

2 Preliminaries

Notation and conventions. If A is an algorithm then $y \leftarrow A(x_1, \ldots, x_n; r)$ means we run A on inputs

$x_1, \ldots, x_n$ and coins r and denote the output by y. By $y \leftarrow\!\!\$\; A(x_1, \ldots, x_n)$ we denote the operation of picking r at random and letting $y \leftarrow A(x_1, \ldots, x_n; r)$. Unless otherwise indicated, an algorithm may be randomized. "PPT" stands for "probabilistic polynomial time" and "PT" stands for "polynomial time." The security parameter is denoted $k \in \mathbb{N}$. If we say that an algorithm is efficient we mean that it is PPT (in the security parameter). All algorithms we consider are efficient unless indicated otherwise.

Standard primitives. We recall the definitions of standard primitives such as trapdoor functions and public-key encryption in Appendix A.

3 Enhanced Chosen-Ciphertext Security

Randomness recovery. We start with a definition of *randomness recovery* for public-key encryption. For any public-key encryption scheme $\mathsf{PKE} = (\mathsf{Kg}, \mathsf{Enc}, \mathsf{Dec})$ we specify an additional *randomness recovery* algorithm that takes a secret key sk and ciphertext c to return coins r; that is, we write $\mathsf{PKE} = (\mathsf{Kg}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{Rec})$. To our knowledge, this notion has been discussed informally in the literature (*e.g.* in [33]) but our formalization is novel. Suppose Enc draws its coins from *Coins*. We require that for all messages $m \in \mathit{MsgSp}(1^k)$

$$\Pr[\mathsf{Enc}(pk, m; r') \neq c \;:\; (pk, sk) \leftarrow\!\!\$\; \mathsf{Kg} \;;\; r \leftarrow\!\!\$\; \mathit{Coins}(1^k) \;;\; c \leftarrow \mathsf{Enc}(pk, m; r) \;;\; r' \leftarrow \mathsf{Rec}(sk, c)]$$

is negligible. Note that we do not require $r = r'$; that is, the randomness recovery algorithm need not return the *same* coins used for encryption; indeed, it may not be possible, information theoretically, to determine r from sk and c. We also do *not* require Rec to be efficient. This means that having such Rec is *not* an additional assumption; we can always assume Rec is the canonical, inefficient randomness recovery algorithm that exhaustively searches $\mathit{Coins}(1^k)$ until it finds such r'. But in the special case that Rec is PT we say that PKE is *randomness recovering.* If in addition the forgoing condition on Rec holds for $r = r'$ we say that PKE is *uniquely* randomness recovering. In the general definition that follows randomness recovering PKE is an important special case, but it is not assumed by the definition.

In the case of tag-based public-key encryption, Rec also takes a *tag* as input. In this case we require that for all $m \in \mathit{MsgSp}(1^k)$ and $t \in \mathit{TagSp}(1^k)$

$$\Pr[\mathsf{Enc}(pk, t, m; r') \neq c \;:\; (pk, sk) \leftarrow\!\!\$\; \mathsf{Kg} \;;\; r \leftarrow\!\!\$\; \mathit{Coins}(1^k) \;;\; c \leftarrow \mathsf{Enc}(pk, t, m; r) \;;\; r' \leftarrow \mathsf{Rec}(sk, t, c)]$$

is negligible.

ECCA definition. We are now ready to state our new notion of enhanced chosen-ciphertext security. Let $\mathsf{PKE} = (\mathsf{Kg}, \mathsf{Enc}, \mathsf{Dec})$ be a public-key encryption scheme. We associate to PKE and an adversary $A = (A_1, A_2)$ an *enhanced chosen-ciphertext attack* experiment,

```
Experiment $\mathbf{Exp}^{\text{ind-ecca}}_{\mathsf{PKE},A}(k)$
    $b \leftarrow\!\!\$\; \{0,1\} \;;\; (pk, sk) \leftarrow\!\!\$\; \mathsf{Kg}(1^k)$
    $(m_0, m_1, St) \leftarrow\!\!\$\; A_1^{\mathsf{Dec}^*(sk,\cdot)}(pk)$
    $c \leftarrow\!\!\$\; \mathsf{Enc}(pk, m_b)$
    $d \leftarrow\!\!\$\; A_2^{\mathsf{Dec}^*(sk,\cdot)}(pk, c, St)$
    If $d = b$ then return 1 else return 0

Oracle $\mathsf{Dec}^*(sk, c)$
    $m \leftarrow \mathsf{Dec}(sk, c)$
    $r' \leftarrow \mathsf{Rec}(sk, c)$
    Return $(m, r')$
```

Above we require that the output of A_1 satisfies $|m_0| = |m_1|$ and that A_2 does not query c to its oracle. Define the *ind-ecca advantage* of A against PKE as

$$\mathbf{Adv}^{\text{ind-ecca}}_{\mathsf{PKE},A}(k) = 2 \cdot \Pr\left[\mathbf{Exp}^{\text{ind-ecca}}_{\mathsf{PKE},A}(k) \text{ outputs } 1\right] - 1 \,.$$

We say that PKE is *enhanced chosen-ciphertext secure* (ECCA-secure) if $\mathbf{Adv}^{\text{ind-ecca}}_{\mathsf{PKE},A}(\cdot)$ is negligible for every efficient A.

Note that when PKE is randomness recovering, the ECCA experiment is efficient. Additionally, the case of randomness-recovering ECCA public-key encryption will be important in the applications we consider.

In general, however, one can still ask whether a scheme meets the notion of ECCA even when it is not randomness recovering. Indeed, Canetti *et al.* [10] did so in the context of commitments. In this case, it may still be possible to simulate the ECCA experiment efficiently since in the proof of security we are additionally given the code of the adversary A (and so, for example, the randomness for encryption might be efficiently extractable from the code of A using non-black-box techniques). However, we do not currently have any constructions of ECCA without randomness recoverability or applications of such a scheme.

Is ECCA Stronger than CCA? A natural question to ask is whether ECCA security is a strictly stronger requirement than CCA security or not. In fact, one can ask the same question when the CCA scheme is also randomness-recovering. We answer this question by describing a randomness recovering CCA encryption scheme that provably does not satisfy ECCA security. We note that our example works in the random oracle model. It would be interesting to provide a similar construction in the standard model. It is also an interesting open question to extend these ideas into a full-fledged separation between the two notions of CCA and ECCA security.

Consider a randomness-recovering CCA-secure scheme $\mathsf{PKE} = (\mathsf{Kg}, \mathsf{Enc}, \mathsf{Dec})$, and let $H : \{0,1\}^k \to \{0,1\}^{|sk|}$ be a hash function modeled as a random oracle. We transform PKE to a new scheme $\mathsf{PKE}_{new} = (\mathsf{Kg}_{new}, \mathsf{Enc}_{new}, \mathsf{Dec}_{new})$ thats is not ECCA secure. The idea is to encrypt the sk using the randomness of a ciphertext c and to embed c and the encryption of sk in the public key of the $\mathsf{PKE} + new$. Querying c to the decryption oracle in the ECCA game would allow the attacker to recover sk and break the scheme. the PKE_{new} is constructed as follows:

```
Alg $\mathsf{Kg}_{new}(1^k)$
    $(pk, sk) \leftarrow_\$ \mathsf{Kg}(1^k)$
    $r \leftarrow_\$ \{0,1\}^k$
    Return $((pk, \mathsf{Enc}(pk, 0; r), H(r) \oplus sk), sk)$

Alg $\mathsf{Enc}_{new}(pk, m)$
    Return $\mathsf{Enc}(pk, m)$

Alg $\mathsf{Dec}_{new}(sk, c)$
    Return $\mathsf{Dec}(sk, c)$
```

Claim 3.1 PKE_{new} is a randomness-recovering CCA encryption schemes but is not ECCA secure.

Sketch. It is easy to see that PKE_{new} is not ECCA secure. In particular, consider the following simple attack: the public key consists of $(pk, \mathsf{Enc}(pk, 0; r), H(r) \oplus sk$. Adversary makes a decryption query for $\mathsf{Enc}(pk, 0; r)$ in the ECCA security game and receives $(0, r)$ back. He then computes $H(r)$, and XORs with the third component of the public key to obtain sk.

We now need to show that PKE_{new} remains CCA secure. But this argument easily follows from the CCA security of the PKE and the fact that given a ciphertext $\mathsf{Enc}(pk, 0; r), H(r)$ is uniformly random if H is a random oracle. The latter is the case since the randomness used in encryption needs to remain unpredictable given the ciphertext for the scheme to be secure. We omit the standard hybrid argument that formalizes this intuition.

Tag-based definition. Let $\mathsf{TB\text{-}PKE} = (\mathsf{Kg}, \mathsf{Enc}, \mathsf{Dec})$ be a tag-based public-key encryption scheme with tag-space *TagSp*. We associate to $\mathsf{TB\text{-}PKE}$ and an adversary $A = (A_1, A_2, A_3)$ a *tag-based enhanced chosen-ciphertext attack* experiment,

```
Experiment $\mathbf{Exp}^{\text{ind-tb-ecca}}_{\mathsf{TB\text{-}PKE},A}(k)$
    $b \leftarrow_\$ \{0,1\}$ ; $(pk, sk) \leftarrow_\$ \mathsf{Kg}(1^k)$
    $t \leftarrow_\$ A_1(1^k)$
    $(m_0, m_1, St) \leftarrow_\$ A_2^{\mathsf{Dec}^*(sk,\cdot,\cdot)}(pk, t)$
    $c \leftarrow_\$ \mathsf{Enc}(pk, t, m_b)$
    $d \leftarrow_\$ A_3^{\mathsf{Dec}^*(sk,\cdot,\cdot)}(pk, t, c, St)$
    If $d = b$ then return 1 else return 0

Oracle $\mathsf{Dec}^*(sk, t, c)$
    $m \leftarrow \mathsf{Dec}(sk, t, c)$
    $r' \leftarrow \mathsf{Rec}(sk, t, c)$
    Return $(m, r')$
```

Above we require that the output of A_2 satisfies $|m_0| = |m_1|$ and that A_3 does not make a query of the form $\mathsf{Dec}^*(sk, t, \cdot)$ to its oracle. Define the *ind-tb-ecca advantage* of A against PKE as

$$\mathbf{Adv}^{\text{ind-tb-ecca}}_{\mathsf{PKE},A}(k) = 2 \cdot \Pr\left[\mathbf{Exp}^{\text{ind-tb-ecca}}_{\mathsf{PKE},A}(k) \text{ outputs } 1\right] - 1 .$$

We say that $\mathsf{TB\text{-}PKE}$ is *tag-based enhanced chosen-ciphertext secure* (TB-ECCA-secure) if $\mathbf{Adv}^{\textit{ind-tb-ecca}}_{\mathsf{PKE},A}(\cdot)$ is negligible for every efficient A.

4 Instantiations

We now detail several constructions of ECCA secure encryption. They are based on notions of adaptivity for trapdoor functions introduced in [28] so accordingly we recall those first.

4.1 Adaptivity for Trapdoor Functions

Adaptive trapdoor functions. Let $\mathsf{TDF} = (\mathsf{Tdg}, \mathsf{Eval}, \mathsf{Inv})$ be a trapdoor function family. We associate to TDF and an inverter I an *adaptive one-way* experiment,

```
Experiment $\mathbf{Exp}^{\mathrm{aow}}_{\mathsf{TDF},I}(k)$
    $(ek, td) \leftarrow_\$ \mathsf{Tdg}(1^k)$ ; $x \leftarrow_\$ \{0,1\}^k$
    $y \leftarrow \mathsf{Eval}(ek, x)$
    $x' \leftarrow_\$ I^{\mathsf{Inv}(td,\cdot)}(ek, y)$
    If $x = x'$ then return 1 else return 0
```

Above we require that I does not query y to its oracle. Define the *aow-advantage* of A against TDF as

$$\mathbf{Adv}^{\mathrm{aow}}_{\mathsf{TDF},I}(k) = \Pr\left[\,\mathbf{Exp}^{\mathrm{aow}}_{\mathsf{TDF},I}(k) \text{ outputs } 1\,\right] .$$

We say that TDF is *adaptive one-way* (or is an ATDF) if $\mathbf{Adv}^{\mathrm{aow}}_{\mathsf{TDF},I}(\cdot)$ is negligible for every efficient I.

Tag-based adaptivity. Let $\mathsf{TB\text{-}TDF} = (\mathsf{Tdg}, \mathsf{Eval}, \mathsf{Inv})$ be a tag-based trapdoor function family. We associate to TDF and an inverter $I = (I_1, I_2)$ a *tag-based adaptive one-way* experiment,

```
Experiment $\mathbf{Exp}^{\mathrm{tb\text{-}aow}}_{\mathsf{TB\text{-}TDF},I}(k)$
    $(ek, td) \leftarrow_\$ \mathsf{Tdg}(1^k)$
    $t \leftarrow_\$ I_1(1^k)$ ; $x \leftarrow_\$ \{0,1\}^k$
    $y \leftarrow \mathsf{Eval}(ek, t, x)$
    $x' \leftarrow_\$ I^{\mathsf{Inv}(td,\cdot,\cdot)}(ek, t, y)$
    If $x = x'$ then return 1 else return 0
```

Above we require that I_2 does not make a query of the form $\mathsf{Inv}(td, t, \cdot)$ to its oracle. Define the *tb-aow-advantage* of A against $\mathsf{TB\text{-}TDF}$ as

$$\mathbf{Adv}^{\mathrm{tb\text{-}aow}}_{\mathsf{TB\text{-}TDF},I}(k) = \Pr\left[\,\mathbf{Exp}^{\mathrm{tb\text{-}aow}}_{\mathsf{TB\text{-}TDF},I}(k) \text{ outputs } 1\,\right] .$$

We say that TDF is *tag-based adaptive one-way* (or is a TB-ATDF) if $\mathbf{Adv}^{\mathrm{tb\text{-}aow}}_{\mathsf{TDF},I}(\cdot)$ is negligible for every efficient I.

Realizations. In [28] it is shown that ATDFs and tag-based ATDFs can be realized from lossy TDFs [33] and correlated-product secure TDFs [35], which can be realized from a variety of standard number-theoretic and lattice-based assumptions. Furthermore, tag-based ATDFs were constructed from a strong but non-decisional (*i.e., search*) problem on RSA in [28].

4.2 ECCA Security from Adaptive Trapdoor Functions

Here we construct ECCA-secure public-key encryption from adaptive TDFs. Our construction draws heavily on the recent work of Hohenberger *et al.* [26] and their notion of *detectable* CCA security (DCCA). This should be contrasted with the approach of [28] used to obtain CCA-secure encryption from ATDFs. Our approach allows us to obtain "enhanced" security and also unique randomness recoverabilty. (See the Introduction for further elaboration.) We note that our construction applies to general ATDFs; in the case of ATDFs with a linear number of hardcore bits we obtain a much more efficient construction, see Appendix B for details.

4.2.1 Enhanced DCCA Security

The notion of Detectable Chosen Ciphertext (DCCA) security was recently introduced by [26]. We define here the notion of *enhanced* DCCA (EDCCA) security, which parallels the notion of enhanced CCA security. In our definition, we require that the DCCA scheme be both enhanced and randomness-recovering. This is due to the fact that our application of DCCA requires both properties. However, the more general notion of enhanced DCCA security (with no efficient randomness-recovering property) may also be of interest.

Detectable Encryption Schemes. A *detectable encryption scheme* is a tuple of probabilistic polynomial time algorithms $(\mathsf{Kg}, \mathsf{Enc}, \mathsf{Dec}, \mathcal{F})$ such that: (1) $(\mathsf{Kg}, \mathsf{Enc}, \mathsf{Dec})$ constitute a public-key encryption scheme, and (2) $\mathcal{F}(pk, c', c) \to \{0,1\}$: the detecting function $\mathcal{F}$ takes as input a public key pk and two ciphertexts c', c and outputs a bit.

Additionally, the detecting function $\mathcal{F}$ must have the following property: Informally, given the description of $\mathcal{F}$ and a public key pk, it it should be hard to find a second ciphertext c' that is related to a "challenge" ciphertext c, i.e. such that $\mathcal{F}(pk, c', c) = 1$, *before being given* c. See [26] for the formal definition of the unpredictability experiment.

EDCCA definition. We are now ready to define enhanced, detectable chosen ciphertext security. Let $\mathsf{PKE} = (\mathsf{Kg}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{Rec}, \mathcal{F})$ be a randomness-recovering public-key encryption scheme. We associate to PKE and an adversary $A = (A_1, A_2)$ an *enhanced detectable chosen-ciphertext attack* experiment,

```
Experiment $\mathbf{Exp}^{\text{ind-edcca}}_{\mathsf{PKE},A}(k)$
    $b \leftarrow_{\$} \{0,1\}$ ; $(pk, sk) \leftarrow_{\$} \mathsf{Kg}(1^k)$
    $(m_0, m_1, St) \leftarrow_{\$} A_1^{\mathsf{Dec}^*(sk,\cdot)}(pk)$
    $c^* \leftarrow_{\$} \mathsf{Enc}(pk, m_b)$
    $d \leftarrow_{\$} A_2^{\mathsf{Dec}^*(c^*,sk,\cdot)}(pk, c, St)$
    If $d = b$ then return 1 else return 0

Oracle $\mathsf{Dec}^*(c^*, sk, c)$
    If $\mathcal{F}(pk, c^*, c) = 1$, Return $\perp$
    Else $m \leftarrow \mathsf{Dec}(sk, c)$
    $r' \leftarrow \mathsf{Rec}(sk, c)$
    Return $(m, r')$
```

Above we require that the output of A_1 satisfies $|m_0| = |m_1|$ and that A_2 does not query c to its oracle. Define the *ind-edcca advantage* of A against PKE as

$$\mathbf{Adv}^{\text{ind-edcca}}_{\mathsf{PKE},A}(k) = 2 \cdot \Pr\left[\mathbf{Exp}^{\text{ind-edcca}}_{\mathsf{PKE},A}(k) \text{ outputs } 1\right] - 1 .$$

We say that PKE is *enhanced detectable chosen-ciphertext secure* (EDCCA-secure) if

- Encryptions are indistinguishable: $\mathbf{Adv}^{\text{ind-edcca}}_{\mathsf{PKE},A}(\cdot)$ is negligible for every efficient A AND
- $\mathcal{F}$ is unpredictable: Every efficient adversary A has negligible probability of succeeding in the unpredictability experiment (see [26]).

4.2.2 EDCCA Security from ATDFs

We construct an EDCCA scheme from ATDFs as follows: Let $\mathsf{TDF} = (\mathsf{Tdg}, \mathsf{Eval}, \mathsf{Inv})$ be a trapdoor function with hardcore bit hc, for example the Goldreich-Levin bit [20]. Define the following multi-bit public-key encryption scheme $\mathsf{EDCCA[TDF]} = (\mathsf{Kg_D}, \mathsf{Enc_D}, \mathsf{Dec_D})$:

```
Alg $\mathsf{Kg_D}(1^k)$
    $(ek, td) \leftarrow_{\$} \mathsf{Tdg}(1^k)$
    Return $(ek, td)$

Alg $\mathsf{Enc_D}(ek, m = m_1, \ldots, m_\ell)$
    Choose $x_1, \ldots, x_\ell$ uniformly at random
    Return $C = (\mathsf{Eval}(ek, x_1), \mathsf{hc}(x_1) \oplus m_1,$
        $\ldots, \mathsf{Eval}(ek, x_\ell), \mathsf{hc}(x_\ell) \oplus m_\ell)$

Alg $\mathsf{Dec}(td, C)$
    Parse $C = (y_1, \beta_1, \ldots, y_\ell, \beta_\ell)$
    For $1 \le i \le \ell$
        Compute $m_i = \mathsf{hc}(\mathsf{Inv}(td, y_i)) \oplus \beta_i$
    Return $m_1, \ldots, m_\ell$
```

THE DETECTING FUNCTION $\mathcal{F}_{\mathsf{D}}$: On input $pk, C^* = (y_1^*, \beta_1^*, \dots, y_\ell^*, \beta_\ell^*)$, $C = (y_1, \beta_1, \dots, y_\ell, \beta_\ell)$, we define:

$$\mathcal{F}_{\mathsf{D}}(pk, C^*, C) = \begin{cases} 1 & \text{if for some } i, j \in [\ell], y_i^* = y_j \\ 0 & \text{otherwise} \end{cases}$$

Claim 4.1 Suppose TDF is adaptive one-way. Then $\mathsf{EDCCA}[\mathsf{TDF}]$ defined above is a multi-bit EDCCA-secure encryption scheme.

We give some intuition for why the claim holds. Assume towards contradiction that we have an efficient adversary $A = (A_1, A_2)$ breaking $\mathsf{EDCCA}[\mathsf{TDF}]$. Using a standard hybrid argument, we have that there must be some index $1 \leq i \leq \ell$ such that A successfully distinguishes encryptions of messages m_0, m_1 which differ only in the i-th bit. Now assuming the existence of A, we construct an efficient adversary A' breaking adaptive one-wayness of TDF.

More specifically, A' receives $\tilde{y} = \mathsf{Eval}(ek, \tilde{x})$ externally and uses A to learn $\mathsf{hc}(\mathsf{Inv}(td, y_i))$. A' does this by simulating the EDCCA decryption oracle for $A = (A_1, A_2)$ using its inversion oracle Inv.

First, we consider simulating responses to queries made by A_1. In this case, we have that with overwhelming probability, for every decryption query $C = (y_1, \beta_1, \dots, y_\ell, \beta_\ell)$ made by A_1, it is the case that $y_j \neq \tilde{y}$ for all $1 \leq j \leq \ell$. Thus, A' can decrypt correctly using oracle access to Inv.

At the end of the first phase, A_1 chooses two messages $m^0 = m_1^0, \dots, m_\ell^0$ and $m^1 = m_0^1, \dots, m_\ell^1$ which differ only in the i-th bit. A' prepares the challenge ciphertext C^* by choosing $x_1, \dots, x_{i-1}, x_{i+1}, \dots x_\ell$ and a bit b uniformly at random and setting

$$C^* = (\mathsf{Eval}(ek, x_1), \mathsf{hc}(x_1) \oplus m_1, \dots, \tilde{y}, b, \dots, \mathsf{Eval}(ek, x_\ell), \mathsf{hc}(x_\ell) \oplus m_\ell)\,.$$

Now, to answer EDCCA decryption queries C submitted by A_2, A' checks whether $\mathcal{F}_{\mathsf{D}}(pk, C^*, C) = 1$. If yes, A' perfectly simulates the decryption oracle by returning $\perp$. If not, then this implies in particular that on input $C = (y_1, \beta_1, \dots, y_\ell, \beta_\ell)$, we have that $y_j \neq \tilde{y}$ for all $1 \leq j \leq \ell$. In this case, A' can use its access to the Inv oracle in order to respond correctly to the decryption query.

We omit the technical details of the proof, since they are standard.

Remark 4.2 Scheme $\mathsf{EDCCA}[\mathsf{TDF}]$ defined above is also uniquely randomness-recovering. This will be crucial for our application to adaptive trapdoor functions in Section 5.

We also wish to stress that it gives a novel example of a DCCA secure scheme; our scheme is *not* the concatenation of ciphertexts for a 1-bit CCA-secure scheme. Indeed, a ciphertext of the form $(\mathsf{Eval}(ek, x), \mathsf{hc}(x) \oplus m)$ is trivially malleable by flipping the second component.

4.2.3 From EDCCA to ECCA Security

We next show that the construction of [26] designed to build a CCA-secure scheme from a DCCA secure one allows us to go from EDCCA to ECCA. That is, beyond what was already shown in [26] we show that the construction preserves "enhanced" security; it also preserves (unique) randomness-recoverabiility. Specifically, we instantiate the construction of [26] with the above randomness recovering EDCCA scheme, a CPA-secure scheme with perfect correctness, and a 1-bounded CCA-secure[5] scheme with perfect correctness (note that all these components can be constructed in a black-box manner from ATDFs):

The EDCCA scheme, $\mathsf{EDCCA}[\mathsf{TDF}]$**:** We instantiate the EDCCA scheme with the scheme given in Section 4.2.2. We note that for simplicity, we sometimes refer to the detecting function $\mathcal{F}_{\mathsf{D}}$ as checking for a "quoting" attack on the challenge ciphertext.

[5]By 1-bounded CCA security, we mean an encryption scheme that is secure under an indistinguishability attack when the adversary may make only a single decryption query to its oracle either before or after receiving the challenge ciphertext.

The CPA scheme, $\mathsf{CPA[TDF]}$**:** We instantiate the CPA scheme with the same scheme $\mathsf{EDCCA[TDF]}$ as above. Note that this scheme has perfect correctness since the Inv algorithm of the ATDF is required to invert correctly with probability 1.

The 1-bounded CCA scheme, $\mathsf{1\text{-}CCA[TDF]}$**:** Since we have already observed above that we can construct a multi-bit CPA scheme with perfect correctness from ATDF, we may now use any construction of a multi-bit 1-bounded CCA scheme with perfect correctness from a multi-bit CPA scheme with perfect correctness. This can be done in a black-box manner via the [11] construction. It is not hard to see that the construction of [11] preserves the perfect correctness property.

The Multi-Bit (Uniquely Randomness Recovering) ECCA Scheme. We present a multi-bit, uniquely randomness recovering, ECCA-secure encryption scheme $\mathsf{PKE[TDF]} = (\mathsf{Kg}_{\mathsf{ECCA}}, \mathsf{Enc}_{\mathsf{ECCA}}, \mathsf{Dec}_{\mathsf{ECCA}})$ using the schemes $\mathsf{EDCCA[TDF]} = (\mathsf{Kg}_{\mathsf{D}}, \mathsf{Enc}_{\mathsf{D}}, \mathsf{Dec}_{\mathsf{D}})$, $\mathsf{1\text{-}CCA[TDF]} = (\mathsf{Kg}_{\mathsf{1b}}, \mathsf{Enc}_{\mathsf{1b}}, \mathsf{Dec}_{\mathsf{1b}})$, and $\mathsf{CPA[TDF]} = (\mathsf{Kg}_{\mathsf{CPA}}, \mathsf{Enc}_{\mathsf{CPA}}, \mathsf{Dec}_{\mathsf{CPA}})$ defined above.

```
Alg $\mathsf{Kg}_{\mathsf{ECCA}}(1^\lambda)$
    $(pk_{in}, sk_{in}) \leftarrow_\$ \mathsf{Kg}_{\mathsf{D}}(1^\lambda)$
    $(pk_A, sk_A) \leftarrow_\$ \mathsf{Kg}_{\mathsf{1b}}(1^\lambda)$
    $(pk_B, sk_B) \leftarrow_\$ \mathsf{Kg}_{\mathsf{CPA}}(1^\lambda)$
    $pk \leftarrow (pk_{in}, pk_A, pk_B)$
    $sk \leftarrow (sk_{in}, sk_A, sk_B)$
    Return $(pk, sk)$

Alg $\mathsf{Enc}_{\mathsf{ECCA}}(pk, m)$
    $(r_A, r_B) \leftarrow_\$ \{0,1\}^\lambda$
    $CT_{in} \leftarrow_\$ \mathsf{Enc}_{\mathsf{D}}(pk_{in}, (r_A, r_B, m))$
    $CT_A \leftarrow \mathsf{Enc}_{\mathsf{1b}}(pk_A, CT_{in}; r_A)$
    $CT_B \leftarrow \mathsf{Enc}_{\mathsf{CPA}}(pk_B, CT_{in}; r_B)$
    Return $CT = (CT_A, CT_B)$

Alg $\mathsf{Dec}_{\mathsf{ECCA}}(sk, CT)$
    $CT_{in} \leftarrow_\$ \mathsf{Dec}_{\mathsf{1b}}(sk_A, CT_A)$
    $(r_A, r_B, m) \leftarrow \mathsf{Dec}_{\mathsf{D}}(sk_{in}, CT_{in})$
    $r_{in} \leftarrow \mathsf{Rec}_{\mathsf{D}}(sk_{in}, CT_{in})$
    If $CT_A = \mathsf{Enc}_{\mathsf{1b}}(pk_A, CT_{in}; r_A)$ and
    $CT_B = \mathsf{Enc}_{\mathsf{CPA}}(pk_B, CT_{in}; r_B)$
        return $(r_A, r_B, m, r_{in})$
    Else return $\perp$
```

Theorem 4.3 $\mathsf{PKE[TDF]}$ *is enhanced CCA-secure and uniquely randomness recovering under the assumptions that* $\mathsf{EDCCA[TDF]}$ *is enhanced DCCA-secure and uniquely randomness recovering,* $\mathsf{1\text{-}CCA[TDF]}$ *is 1-bounded CCA secure with perfect correctness, and* $\mathsf{CPA[TDF]}$ *is CPA-secure with perfect correctness.*

Note that Theorem 4.3 implies that there is a black-box construction of multi-bit, uniquely randomness recovering, enhanced CCA-secure encryption from ATDF.

Proof: The proof is based on [26]. We begin by defining a game which is slightly different than the regular enhanced CCA game, but will be useful in our analsysis of $\mathsf{PKE[TDF]}$:

Enhanced Nested Indistinguishability Game for scheme $\mathsf{PKE[TDF]}$: We associate to the scheme $\mathsf{PKE[TDF]}$ and to an adversary $A = (A_1, A_2)$ an *enhanced nested indistinguishability under chosen ciphertext attack* experiment,

```
Experiment $\mathbf{Exp}^{\text{nested-ind-ecca}}_{\mathsf{PKE[TDF]},A}(k)$
    $b, z \leftarrow_\$ \{0,1\}$ ; $(pk, sk) \leftarrow_\$ \mathsf{Kg}_{\mathsf{ECCA}}(1^k)$
    $(m_0, m_1, St) \leftarrow_\$ A_1^{\mathsf{Dec}^*_{\mathsf{ECCA}}(sk,\cdot)}(pk)$
    $r_A, r_B \leftarrow_\$ \{0,1\}^\lambda$
    If $z = 0$
        $CT^*_{in} \leftarrow_\$ \mathsf{Enc}_{\mathsf{D}}(pk_{in}, (r_A, r_B, m_b))$
    Else if $z = 1$
        $CT^*_{in} \leftarrow_\$ \mathsf{Enc}_{\mathsf{D}}(pk_{in}, 0^{|r_A|+|r_A|+|m_b|})$
    $CT^*_A \leftarrow \mathsf{Enc}_{\mathsf{1b}}(pk_A, CT^*_{in}; r_A)$
    $CT^*_B \leftarrow \mathsf{Enc}_{\mathsf{CPA}}(pk_B, CT^*_{in}; r_B)$
    $CT^* \leftarrow (CT^*_A, CT^*_B)$
    $z' \leftarrow_\$ A_2^{\mathsf{Dec}^*_{\mathsf{ECCA}}(sk,\cdot)}(pk, CT^*, St)$
    If $z' = z$ then return 1 else return 0

Oracle $\mathsf{Dec}^*_{\mathsf{ECCA}}(sk, c)$
    $m \leftarrow \mathsf{Dec}_{\mathsf{ECCA}}(sk, c)$
    $r' \leftarrow \mathsf{Rec}_{\mathsf{ECCA}}(sk, c)$
    Return $(m, r')$
```

Above we require that the output of A_1 satisfies $|m_0| = |m_1|$ and that A_2 does not query CT^* to its oracle. In the following, we refer to decryption queries made by A_1 as "Phase 1 queries" and to decryption queries made by A_2 as "Phase 2 queries."

Define the *nested-ind-ecca advantage* of A against $\mathsf{PKE[TDF]}$ as

$$\mathbf{Adv}^{\text{nested-ind-ecca}}_{\mathsf{PKE[TDF]},A}(k) = 2 \cdot \Pr\left[\mathbf{Exp}^{\text{nested-ind-ecca}}_{\mathsf{PKE[TDF]},A}(k) \text{ outputs } 1\right] - 1 \,.$$

We say that $\mathsf{PKE[TDF]}$ has *enhanced nested indistinguishable encryptions under a chosen ciphertext attack* if $\mathbf{Adv}^{\text{nested-ind-ecca}}_{\mathsf{PKE[TDF]},A}(\cdot)$ is negligible for every efficient A.

It should be clear that enhanced nested indistinguishability of $\mathsf{PKE[TDF]}$ under a chosen ciphertext attack implies enhanced CCA security of $\mathsf{PKE[TDF]}$ (via a simple hybrid argument).

Consider the following event:

Definition 4.4 (The Bad Query Event) *We say that a* bad query event *has occurred during an execution of this experiment if in Phase 2, the adversary A makes a decryption query of the form $CT = (CT_A, CT_B)$ such that*

- *(Quoting attack on inner ciphertext:)* $\mathcal{F}_{\mathsf{D}}(pk_{in}, CT^*_{in}, \mathsf{Dec_{1b}}(sk_A, CT_A)) = 1$ *AND*
- *(Query ciphertext differs from challenge ciphertext in first half:)* $CT^*_A \neq CT_A$.

We will show that Bad Query Event occurs with at most negligible probability when $z = 1$ and when $z = 0$. Once we have shown this, Nested Indistinguishability of $\mathsf{PKE[TDF]}$ will follow in a straightforward manner.

Lemma 4.5 *Bad Query Event occurs with negligible probability when $z = 1$.*

We prove Lemma 4.5 via a sequence of hybrids:

Hybrid H_0: Proceeds exactly as the nested indistinguishability game for the case where $z = 1$.

Hybrid H_1: Proceeds exactly like H_0 except that CT^*_B is set to be: $CT^*_B = \mathsf{Enc_{CPA}}(pk_B, 1^k; r_B)$.

Claim 4.6 The probability of a Bad Query Event in H_1 and H_0 differs by a negligible amount.

Since sk_B is never used by the decryption oracle and since the decryption oracle can detect all Bad Query Events, the claim follows immediately by a reduction to the semantic security of $\mathsf{CPA[TDF]}$.

Hybrid H_2: Proceeds exactly like H_1 except CT^*_A is set to be $CT^*_A = \mathsf{Enc_{1b}}(pk_A, 1^k; r_A)$.

Claim 4.7 The probability of Bad Query Event in H_2 is negligible.

The claim follows due to the fact that the challenge ciphertext in H_2 contains no information about CT^*_{in} and since the detecting function $\mathcal{F}_{\mathsf{D}}$ is unpredictable.

Claim 4.8 The probability of a Bad Query Event in H_2 and H_1 differs by a negligible amount.

Intuitively, Claim 4.8 will reduce to the 1-bounded CCA security of 1-CCA[TDF].

Proof: Assume towards contradiction that there is some efficient adversary A which causes Bad Query Event to occur with negligible probability in H_2 and non-negligible probability in H_1. We denote by q the (polynomial) number of Phase 2 queries made by A. By standard hybrid argument, there must be some index $i \in [q]$ such that the i-th Phase 2 query made by A in H_1 causes Bad Query Event to occur with non-negligible probability. On the other hand, for every index i, the i-th Phase 2 query made by A in H_2 causes Bad Query Event to occur with at most negligible probability.

Fix such i. We construct an adversary B which breaks the security of 1-CCA[TDF]. B will receive a challenge ciphertext that is either an encryption of CT^*_{in} or of 1^n. In case the challenge ciphertext was an encryption of CT^*_{in}, B will perfectly simulate the adversary's view in H_1. In case the challenge ciphertext was an encryption of 1^n, B will perfectly simulate the adversary's view in H_2.

Moreover, B will be able to detect whether Bad Query Event occurred in the i-th Phase 2 query of the experiment. Thus, if Bad Query Event occurs in the i-th query with non-negligible probability in H_1 and negligible probability in H_2, then B will be able to break security of 1-CCA[TDF].

Formally, consider the following Simulated Decryption Oracle:

Simulated Decryption Oracle:

- Decrypt CT_B using sk_B to retreive CT_{in}.
- Decrypt CT_{in} using sk_{in} to retrieve (r_A, r_B, m).
- Use the randomness recovering algorithm Rec_D and sk_{in} to retrieve $r_{in} = \mathsf{Rec}_D(sk_{in}, CT_{in})$.
- Check that CT_A and CT_B were formed correctly with respect to (r_A, r_B, CT_{in}). If not, output $\perp$. Otherwise, output (m, r_A, r_B, r_{in}).

Note that for every possible string CT submitted to the oracle, the output of the Simulated Decryption Oracle and the real decryption oracle is identical since 1-CCA[TDF] and CPA[TDF] have perfect correctness.

Now, B does the following: B receives pk_A from its external 1-bounded CCA challenger and generates $(sk_B, pk_B), (sk_{in}, pk_{in})$ honestly.

B and A interact in Phase 1 (while B uses the Simulated Decryption Oracle to respond to decryption queries). At some point A outputs m_0 and m_1. B computes $CT^*_{in} = \mathsf{Enc}_\mathsf{D}(pk_{in}, 0^\ell)$ and $CT^*_B = \mathsf{Enc}_\mathsf{CPA}(pk_B, 1^k)$. It outputs CT^*_{in} and 1^k as messages m_0, m_1 to its external 1-bounded CCA challenger and receives a ciphertext CT^*_A. B then outputs $CT^* = (CT^*_A, CT^*_B)$ to A. B continues interacting with A during Phase 2, while using the Simulated Decryption Oracle as above. When B receives the i-th Phase 2 query of A, denoted by (CT^i_A, CT^i_B), B checks for the Bad Query Event by doing the following:

- B checks if $CT^i_A \neq CT^*_A$.
- If so, B submits CT^i_A to its external 1-bounded CCA decryption oracle and receives CT^i_{in} in response.
- B checks whether $\mathcal{F}_\mathsf{D}(pk_{in}, CT^*_{in}, CT^i_{in}) = 1$ (i.e. whether a quoting attack occurred). If yes, B outputs 0. Otherwise, B outputs 1.

Since by assumption we have that Bad Query Event occurs with non-negligible probability at the i-th Phase 2 query in H_1 and occurs with negligible probability at the i-th Phase 2 query in H_2, we have that B achieves non-negligible advantage in the external 1-bounded CCA security game. This is a contradiction to the security of $\mathsf{1\text{-}CCA[TDF]}$ and so the claim is proved. ▌

Lemma 4.5 follows immediately from Claims 4.6, 4.7 and 4.8.

We now turn to the case where $z = 0$:

Lemma 4.9 *Bad Query Event occurs with negligible probability when* $z = 0$.

Intuitively, Lemma 4.9 will reduce to the enhanced detectable CCA security of $\mathsf{EDCCA[TDF]}$.

Proof: We have already shown that when $z = 1$, Bad Query Event occurs with negligible probability. We will now show that if there is an efficient adversary A causing Bad Query Event to occur with non-negligible probability when $z = 0$, then there is a ppt adversary B breaking the security of $\mathsf{EDCCA[TDF]}$.

Assume towards contradiction that there is an efficient adversary A which causes Bad Query Event to occur with non-negligible probability when $z = 0$. Consider the following efficient adversary B which interacts with A in a run of the nested indistinguishability experiment, while externally participating in an enhanced DCCA indistinguishability experiment. B does the following:

Setup: B receives pk_{in} externally from the EDCCA experiment. B honestly generates $(sk_A, pk_A), (sk_B, pk_B)$.

Phase 1: B simulates the honest (enhanced) decryption oracle using sk_A and the decryption oracle in the external enhanced DCCA experiment. At the end of this phase A will output m_0, m_1.

Challenge: Choose random $\beta \in \{0, 1\}$ and $r_A, r_B \in \{0, 1\}^\lambda$. Send to the external enhanced DCCA challenger $M_0 = (r_A, r_B, m_\beta)$ and $M_1 = 0^{|M_0|}$ and obtain the ciphertext CT^*_{in}. Compute CT^*_A and CT^*_B honestly, given CT^*_{in}, r_A, r_B. Return $CT^* = (CT^*_A, CT^*_B)$ to A.

Phase 2: When A queries the decryption oracle on $CT = (CT_A, CT_B)$, compute $CT_{in} = \mathsf{Dec_{1b}}(sk_A, CT_A)$.

Case 1 (a bad query event): $CT_A \neq CT^*_A$ and yet $\mathcal{F}_{\mathsf{D}}(pk_{in}, CT^*_{in}, CT_{in}) = 1$ (i.e. a quoting attack occurred), then abort and output the bit 0.

Case 2 (partial match with challenge): $CT_A = CT^*_A$, then return $\perp$ to A.

Otherwise, query the external EDCCA decryption oracle to decrypt CT_{in} and return its randomness. Check that CT_A and CT_B are consistent with the response. If not, return $\perp$. Otherwise, return the message and all randomness.

Output: When A outputs a bit, B outputs 0 or 1 with probability 1/2.

We argue that B correctly answers all decryption queries except when it aborts. This will follow immediately once we establish that B always answers correctly by returning $\perp$ when a Case 2 query occurs. We next show that this is indeed the case.

Since a decryption query on the challenge is forbidden by the experiment, if $CT_A = CT^*_A$, then $CT_B \neq CT^*_B$. However, in this case CT must be an invalid ciphertext. We see this as follows: Since decryption is deterministic, we have that $CT_{in} = \mathsf{Dec_{1b}}(sk_A, CT_A) = \mathsf{Dec_{1b}}(sk_A, CT^*_A)$ and $(r_A, r_B, m) = \mathsf{Dec_D}(sk_{in}, CT_{in})$. But this means that there is only one possible ciphertext CT_B that matches $CT_A = CT^*_A$. Since the challenge CT^* is a valid ciphertext, CT^*_B must be this value and so $CT = (CT^*_A, CT_B)$ must be invalid.

Now, if a Case 1 query occurs, B cannot decrypt using its EDCCA decryption oracle and must abort the experiment. But in this case, B can already guess that $z = 0$ since when $z = 1$, Case 1 occurs with negligible probability.

More specifically, when B aborts, it causes the external EDCCA experiment to output 1 with high probability. Moreover, when B does not abort, it causes the external EDCCA experiment to output 1 with probability 1/2. Since B aborts with non-negligible probability when $z = 0$, B causes the experiment's output to be 1 with probability non-negligibly greater than 1/2. This is a contradiction to the security of EDCCA[TDF] and so the claim is proved. ▌

Finally, assuming that Bad Query Event occurs with negligible probability both when $z = 0$ and $z = 1$, we show that Nested Indistinguishability under chosen ciphertext attacks holds. This is straightforward via a reduction to the enhanced DCCA security of EDCCA[TDF]. ▌

4.3 ECCA Security from Tag-Based Adaptive Trapdoor Functions

We next give constructions of ECCA-secure public-key encryption from *tag-based* adaptive trapdoor functions introduced by Kiltz *et al.* [28].

From tag-based ATDF to tag-based ECCA-secure PKE. It is straightforward to construct a multi-bit tag-based ECCA-secure PKE scheme from a tag-based ATDF, as follows. Let TB-TDF = (Tdg, Eval, Inv) be a tag-based adaptive trapdoor function family with tag-space *TagSp* and hardcore bit hc. Define the following tag-based public-key encryption scheme TB-PKE[TB-TDF] = (Kg, Enc, Dec) with tag-space *TagSp* and message-space $\{0,1\}^\ell$:

```
Alg Kg(1^k)
    (ek, td) <-$ Tdg(1^k)
    Return (ek, td)

Alg Enc(ek, t, m)
    For i = 1 to ℓ do:
        x_i <-$ {0,1}^k
        c_{i,1} <- Eval(ek, t, x_i)
        c_{i,2} <- hc(x) ⊕ m[i]
    c <- ((c_{1,1}, c_{1,2}), ..., (c_{ℓ,1}, c_{ℓ,2}))
    Return c

Alg Dec(td, t, c)
    ((c_{1,1}, c_{1,2}), ..., (c_{ℓ,1}, c_{ℓ,2})) <- c
    For i = 1 to ℓ do:
        x_i <- Inv(td, c_{i,1})
        m[i] <- hc(x_i) ⊕ c_{i,2}
    Return m
```

Remark 4.10 Scheme TB-PKE[TB-TDF] defined above is uniquely randomness recovering.

Proposition 4.11 *Suppose* TB-TDF *is adaptive one-way. Then* TB-PKE[TB-TDF] *defined above is ECCA-secure.*

We omit the proof, which is routine.

From ECCA-secure tag-based PKE to ECCA-secure PKE. Note that Kiltz *et al.* [28] show a construction of CCA-secure PKE from any CCA-secure tag-based PKE using a strongly one-time unforgeable signature scheme. However, this construction does not preserve the randomness-recovering property or the ECCA security of the tag-based PKE. To get around this issue, and to construct ECCA-secure PKE from ECCA-secure tag-based PKE we employ a transformation of Boneh *et al.* [7], instead. Let TB-PKE = $\mathsf{Kg}_{tag}, \mathsf{Enc}_{tag}, \mathsf{Dec}_{tag}$) be a tag-based public-key encryption scheme, H, g be hash functions, and MAC = (tag, ver) be a message-authentication code. Define PKE[TB-PKE, H, g, MAC] = (Kg, Enc, Dec):

```
Alg Kg(1^k)
    (pk, sk) <-$ Kg_tag(1^k)
    Return (pk, sk)

Alg Enc(pk, m)
    x <-$ {0,1}^k ; c_1 <- H(x)
    c_2 <-$ Enc_tag(pk, c_1, m||x)
    c_3 <- tag(g(x), c_2)
    Return (c_1, c_2, c_3)

Alg Dec(sk, (c_1, c_2, c_3))
    m||x <- Dec_tag(sk, c_1, c_2)
    If H(x) ≠ c_1 then return ⊥
    If ver(g(x), c_2) = 1 then return m
    Else return ⊥
```

Remark 4.12 If $\mathsf{TB\text{-}PKE}$ is (uniquely) randomness recovering, so is $\mathsf{PKE}[\mathsf{TB\text{-}PKE}, H, g, \mathsf{MAC}]$. In particular, this is the case when $\mathsf{TB\text{-}PKE} = \mathsf{TB\text{-}PKE}[\mathsf{TB\text{-}TDF}]$ as defined above. Thus, we obtain a uniquely randomness-recovering ECCA-secure PKE scheme from any tag-based ATDF.

Proposition 4.13 *Suppose* $\mathsf{TB\text{-}PKE}$ *is ECCA-secure,* H *is target collision-resistant,* g *is pairwise-independent, and* MAC *is strongly unforgeable. Then* $\mathsf{PKE}[\mathsf{TB\text{-}PKE}, H, g, \mathsf{MAC}]$ *is ECCA-secure.*

Proof: We prove security using a sequence of hybrids. Our proof follows that of [7], and uses a deferred analysis technique originating from [19].

Hybrid H_0: The first game is the *ind-cca* game for $\mathsf{PKE}[\mathsf{TB\text{-}PKE}, H, g, \mathsf{MAC}]$ as defined earlier. Let $c^* = (c_1^*, c_2^*, c_3^*)$ be the challenge ciphertext, and x^* be random input used as input to the H when computing the challenge ciphertext.

Hybrid H_1: H_1 is the same as H_0 except that on decryption queries of the form (c_1^*, c_2, c_3) we always return $\perp$. Let *valid* be the event that this ciphertext is indeed valid (it has a valid decryption).

Obviously we have that $|\mathbf{Adv}_A^{H_1}(k) - \mathbf{Adv}_A^{H_0}(k)| \le \Pr_1[\mathit{valid}]$. Let *coll* be the event that c_2 is valid and correctly decrypts to a message $m||x$ and at the same time we have $g(x) \ne g(x^*)$. Furthermore, let $forge$ be the event that $c_3 \leftarrow \mathsf{tag}(g(x^*), c_2)$. It is easy to see that $\Pr_1[\mathit{valid}] < \Pr_1[\mathit{coll}] + \Pr_1[forge]$. Note that $\Pr_1[\mathit{coll}]$ is negligible given the collision resistance of H. In particular, $g(x) \ne g(x^*)$ implies that $x \ne x^*$, which makes the pair (x, x^*) a collision for H. We note that this argument works in presence of an ECCA oracle as well. In particular, note that decryption queries of the form (c_1, c_2, c_3) are only answered if $c_1 \ne c_1^*$, and in this case, one can query c_2 to the decryption oracle for the tag-based PKE and recover both the underlying message and all the randomness used in generating the ciphertext (note that c_3 is deterministic given c_1 and c_2). Analyzing he bound on $\Pr_1[forge]$ is deferred to a later hybrid.

Hybrid H_2: H_2 is the same as H_1 except that when computing the challenge ciphertext we compute $c_2^* = \mathsf{Enc}_{tag}(pk, c_1, 0^{|m_0|}||0^k)$.

Note that $\mathbf{Adv}_A^{H_2}(k) = 1/2$ for any PPT adversary A. It is also straightforward to see that $|\mathbf{Adv}_A^{H_2}(k) - \mathbf{Adv}_A^{H_1}| < \mathbf{Adv}_{\mathsf{TB\text{-}PKE},A}^{\mathsf{ind\text{-}ecca}}(k)$. For the latter, once again, decryption queries are handled as discussed above. The same bound is true for $\Pr_2[forge] - \Pr_1[forge]$.

Hybrid H_3: H_3 is the same as H_2 except that the key for the MAC in the challenge ciphertext (i.e. for computing c_3^*) is generated uniformly at random as opposed being set to $g(x^*)$.

We note that $|\mathbf{Adv}_A^{H_3}(k) - \mathbf{Adv}_A^{H_2}|$ is statistically bounded since the only information available about x^* is $H(x^*)$. But since H is compressing and g is pairwise-independent hash function, it operates as an extractor of the remaining randomness in x^* and outputs a uniformly random key for the MAC. Distinguishing a uniformly random key from $g(x^*)$ is therefore negligible and bounded by this statistical bound (see [7] for a complete argument). Decryption queries are handled as before. The same argument implies that $\Pr_3[forge] - \Pr_2[forge]$ is also negligible.

It remains for us show that $\Pr_3[forge]$ is also negligible but this automatically follows from the unforgeability of the MAC. ▌

5 Application to Adaptive Trapdoor Functions

In this section, we establish that the notions of adaptive trapdoor functions and tag-based adaptive trapdoor functions introduced by Kiltz *et al.* [28] are *equivalent* (via fully black-box reductions), resolving a main open question of [28]. In light of Section 4, we just need to show that reverse implications hold; namely, that from a randomness-recovering ECCA-secure PKE scheme we can get both ATDFs and tag-based ATDFs.

5.1 From ECCA Security to Adaptivity

To construct an adaptive trapdoor function from a uniquely randomness-recovering ECCA-secure PKE scheme, we use part of the input to the former as coins for the latter used to encrypt the other part. Namely, let $\mathsf{PKE} = (\mathsf{Kg}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{Rec})$ be a uniquely randomness-recovering public-key encryption scheme with message-space *MsgSp* and coin-space *Coins*. Define a trapdoor function family $\mathsf{TDF}[\mathsf{PKE}] = (\mathsf{Tdg}, \mathsf{Eval}, \mathsf{Inv})$ on domain $\mathit{MsgSp} \times \mathit{Coins}$ as follows:

```
Alg Tdg(1^k)
    $(pk, sk) \leftarrow_\$ \mathsf{Kg}(1^k)$
    Return $(pk, sk)$

Alg Eval$(pk, x)$
    $(m, r) \leftarrow x$
    $c \leftarrow \mathsf{Enc}(pk, m; r)$
    Return $c$

Alg Inv$(sk, c)$
    $m \leftarrow \mathsf{Dec}(sk, c)$
    $r \leftarrow \mathsf{Rec}(sk, c)$
    Return $(m, r)$
```

Proposition 5.1 *Suppose* PKE *is uniquely randomness recovering and ECCA-secure. Then* TDF[PKE] *defined above is adaptive one-way.*

Proof: Given an AOW-adversary I against TDF[PKE], we can easily construct an ECCA-adversary $A = (A_1, A_2)$ against PKE, as follows:

```
Adversary $A_1^{\mathsf{Dec}^*(sk,\cdot)}(pk)$
    $m_0, m_1 \leftarrow_\$ \mathit{MsgSp}(1^k)$
    Return $(m_0, m_1, m_0 \| m_1)$

Adversary $A_2^{\mathsf{Dec}^*(sk,\cdot)}(pk, c, St)$
    $m_0 \| m_1 \leftarrow St$
    Run $I$ on inputs $pk, c$:
    When $I$ makes query $y$ do:
        $(m, r) \leftarrow \mathsf{Dec}^*(sk, y)$
        Return $(m, r)$
    Let $(m^*, r^*)$ be the output of $I$
    If $m_0 = m^*$ then return 0
    Else return 1
```

It is clear by construction that $\mathbf{Adv}^{\mathrm{aow}}_{\mathsf{TDF},I}(\cdot) \leq \mathbf{Adv}^{\mathrm{ind\text{-}ecca}}_{\mathsf{PKE},A}(\cdot)$ which proves the claim. ∎

5.2 From ECCA Security to Tag-Based Adaptivity

To construct a tag-based adaptive trapdoor function from a uniquely randomness-recovering ECCA-secure PKE scheme, we can use an analogous construction of [29, Section 4.4]. Namely, let $\mathsf{PKE} = (\mathsf{Kg}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{Rec})$ be a uniquely randomness-recovering public-key encryption scheme. Define a tag-based trapdoor function family $\mathsf{TB\text{-}TDF}[\mathsf{PKE}] = (\mathsf{Tdg}, \mathsf{Eval}, \mathsf{Inv})$ as follows:

```
Alg Tdg(1^k)
    $(pk, sk) \leftarrow_\$ \mathsf{Kg}(1^k)$
    Return $(pk, sk)$

Alg Eval$(pk, t, x)$
    $m \| r \leftarrow x$
    $c \leftarrow \mathsf{Enc}(pk, t \| m; r)$
    Return $c$

Alg Inv$(sk, t, c)$
    $t' \| m \leftarrow \mathsf{Dec}(sk, c)$
    $r \leftarrow \mathsf{Rec}(sk, c)$
    If $t = t'$ then return $m \| r$
    Else return $\perp$
```

Proposition 5.2 *Suppose* PKE *is uniquely randomness-recovering and ECCA-secure. Then* TB-TDF[PKE] *is tag-based adaptive one-way.*

Proof: Given an TB-AOW-adversary I against TB-TDF[PKE], we construct an ECCA-adversary $A = (A_1, A_2, A_3)$ against PKE, as follows:

```
Adversary $A_1(pk)$
    $t \leftarrow\$ I_1(pk)$
```

```
Adversary $A_1^{\mathsf{Dec}^*(sk,\cdot)}(pk,t)$
    $m_0, m_1 \leftarrow\$ MsgSp(1^k)$
    Return $(m_0, m_1, m_0\|m_1)$
```

```
Adversary $A_2^{\mathsf{Dec}^*(sk,\cdot)}(pk,t,c,St)$
    $m_0\|m_1 \leftarrow St$
    Run $I$ on inputs $pk, t, c$:
    When $I$ makes query $y, t'$ do:
        $(m,r) \leftarrow \mathsf{Dec}^*(sk,y,t')$
        Return $(m,r)$
    Let $(m^*, r^*)$ be the output of $I$
    If $m_0 = m^*$ then return 0
    Else return 1
```

We claim that $\mathbf{Adv}^{\text{tb-aow}}_{\mathsf{TB\text{-}TDF},I}(\cdot) \le \mathbf{Adv}^{\text{ind-ecca}}_{\mathsf{PKE},A}(\cdot)$. To see this, note that I_1, I_2 does make a query of the form $t' = t$, which by consistency of PKE means that A does not query its challenge ciphertext. ▌

6 Application to PKE with Non-Interactive Opening

In this section, we show that ECCA-secure encryption is a natural building-block for *public key encryption with non-interactive opening* (PKENO) [14, 13, 17, 18]. PKENO allows the receiver to non-interactively prove that a given ciphertext decrypts to a claimed message. Our constructions yield new and practical PKENO schemes. As discussed in the introduction, PKENO has applications to multiparty computation (*e.g.*, auctions and elections), secure message transmission, group signatures, and more.

PKENO. Public-key encryption with non-interactive opening (PKENO) extends a scheme $\mathsf{PKE} = (\mathsf{Kg}, \mathsf{Enc}, \mathsf{Dec})$ for public-key encryption by the following algorithms: Prove takes as input a secret key sk and a ciphertext c, and outputs a proof π. Ver takes as input a public key pk, a ciphertext c, a plaintext m and a proof π, and outputs 0 or 1.

In addition to decryption correctness, stating that $\Pr[\mathsf{Dec}(sk, \mathsf{Enc}(pk,m)) \neq m \,:\, (pk, sk) \leftarrow\$ \mathsf{Kg}(1^k)]$ is negligible for all $k \in \mathbb{N}$ and $m \in MsgSp(1^k)$, we require *proof correctness*: i.e. for all ciphertexts (i.e. strings) c:

$$\Pr[\mathsf{Ver}(pk, c, \mathsf{Dec}(sk,c), \mathsf{Prove}(sk,c)) \neq 1 \,:\, (pk, sk) \leftarrow\$ \mathsf{Kg}(1^k)]$$

is negligible.

In [13, 17], security of PKENO is defined by *indistinguishability under chosen-ciphertext and -proof attacks* (IND-CCPA) and *proof soundness.* The former guarantees that a ciphertext hides the plaintext even when the adversary can see decryptions of and proofs for other ciphertexts; the latter formalizes that no adversary should be able to produce a proof for a ciphertext and a message which is not encrypted by that ciphertext.

Indistinguishability. Let $\mathsf{PKENO} = (\mathsf{Kg}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{Prove}, \mathsf{Ver})$ be a public-key encryption scheme with non-interactive opening. We associate to PKENO and an adversary $A = (A_1, A_2)$ the *chosen-ciphertext and -proof attack* experiment given on the left in Figure 2. We require that the output of A_1 satisfies $|m_0| = |m_1|$ and that A_2 does not query c to any of its oracles. We say that PKENO is *chosen-ciphertext and -proof-attack secure* (CCPA-secure) if $\mathbf{Adv}^{\text{ind-ccpa}}_{\mathsf{PKENO},A}(k) := 2 \cdot \Pr\left[\mathbf{Exp}^{\text{ind-ccpa}}_{\mathsf{PKENO},A}(k) \text{ outputs } 1\right] - 1$ is negligible for every efficient A.

Proof soundness. We associate to a scheme $\mathsf{PKENO} = (\mathsf{Kg}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{Prove}, \mathsf{Ver})$ and an adversary $A = (A_1, A_2)$ a *proof-soundness* experiment, given on the right in Figure 2. We say that PKENO is *proof-sound* if $\mathbf{Adv}^{\text{proof-snd}}_{\mathsf{PKENO},A}(k) := \Pr\left[\mathbf{Exp}^{\text{proof-snd}}_{\mathsf{PKENO},A}(k) \text{ outputs } 1\right]$ is negligible for every efficient A.

We note that as compared to [13, 17] our definition of proof soundness also considers adversarially-produced ciphertexts, which not even be a valid output of the encryption algorithm. Note that it is already required by proof correctness that the PKENO correctly proves decryption of such ciphertexts (which in general may or may not decrypt to $\perp$), so it would seem that constructions should achieve this stronger notion of proof soundness anyway.

```
Experiment $\mathbf{Exp}^{\text{ind-ccpa}}_{\mathsf{PKENO},A}(k)$
    $b \leftarrow_{\$} \{0,1\}$ ; $(pk, sk) \leftarrow_{\$} \mathsf{Kg}(1^k)$
    $(m_0, m_1, St) \leftarrow_{\$} A_1^{\mathsf{Dec}(sk,\cdot),\mathsf{Prove}(sk,\cdot)}(pk)$
    $c \leftarrow_{\$} \mathsf{Enc}(pk, m_b)$
    $d \leftarrow_{\$} A_2^{\mathsf{Dec}(sk,\cdot),\mathsf{Prove}(sk,\cdot)}(pk, c, St)$
    If $d = b$ then return 1 else return 0
```

```
Experiment $\mathbf{Exp}^{\text{proof-snd}}_{\mathsf{PKENO},A}(k)$
    $(pk, sk) \leftarrow_{\$} \mathsf{Kg}(1^k)$
    $(m', \pi', c') \leftarrow_{\$} A(pk, sk)$
    $m \leftarrow \mathsf{Dec}(sk, c')$
    If $\mathsf{Ver}(pk, c', m', \pi') = 1$ and $m \neq m'$
        then return 1 ; else return 0
```

Figure 2: Security experiments for PKENO.

Strong proof soundness. An even stronger notion of proof soundness is defined in [18], which also handles maliciously chosen public key (*i.e.*, security for senders against a malicious receiver). Such a notion is quite challenging to achieve, and hence we mostly focus on the above formulation of proof soundness in the paper. However, in Appendix D we define notions of strong proof soundness and discuss how our constructions can be adapted to to meet them.

6.1 PKENO-Compatible ECCA-Secure Encryption

A natural approach to building PKENO suggested by [14] is to use a randomness-recovering encryption scheme and have the receiver provide the recovered coins as the proof. A moment's reflection reveals that for this approach to work, the encryption scheme must be ECCA secure. In addition, as discussed in [14, 13, 18], we also need a way for the receiver to prove correct decryption of ciphertexts not in the range of the encryption algorithm, in which case such coins may not be defined. Our goal in this section is to define a notion of *PKENO-compatible ECCA-secure encryption* for which we can do this.

Partial-randomness recovery. It turns out that for such schemes we do not always achieve, nor need, the notion of full randomness recovery, so we first define a natural generalization we call *partial-randomness recovery*, which (loosely) says that enough of the random coins are recovered to uniquely identify the underlying message. (Such a notion is alluded to in [33], who note that their CCA-secure encryption seem is not actually fully randomness-recovering because they use a one-time signature, and is generally useful whenever we use a "publicly verifiable" but randomized component such as a one-time signature or NIZK.) However, in order to deal with the case that some ciphertexts outside the range of the encryption algorithm may not decrypt to $\perp$, we also *strengthen* what we get from randomness-recovering encryption in some respect; see the discussion following the definition.

Formally, Suppose Enc draws its coins from *Coins*. We say that a public-key encryption scheme $\mathsf{PKE} = (\mathsf{Kg}, \mathsf{Enc}, \mathsf{Dec})$ has *partial-randomness recovery* if it also has a *partial-randomness recovering algorithm* pRec and a *message-consistency checking algorithm* Cons, which with the help of partial randomness can check whether a ciphertext c encrypts a message m. Namely:

- (Completeness) For all $(pk, sk) \leftarrow_{\$} \mathsf{Kg}$ and all $c \in \{0,1\}^*, m \in MsgSp(1^k) \cup \{\perp\}$, let $s \leftarrow_{\$} \mathsf{pRec}(sk, c)$. Then
$$\mathsf{Dec}(sk, c) = m \wedge m \neq \perp \;\Rightarrow \mathsf{Cons}(pk, c, m, s) = 1 \ .$$
- (Soundness) For all $(pk, sk) \leftarrow_{\$} \mathsf{Kg}$ and all $c \in \{0,1\}^*, m \in MsgSp(1^k), s \in \{0,1\}^*$:
$$\mathsf{Cons}(pk, c, m, s) = 1 \;\Rightarrow\; \mathsf{Dec}(sk, c) = m \ .$$

Relation to randomness recovery. It turns out that a fully randomness recovering scheme is not necessarily partial-randomness recovering as we have defined it. This is because, the completeness condition requires that even for *invalid* ciphertexts c (*i.e.*, those that are never output by the encryption algorithm) that do not decrypt to $\perp$ but rather some $m \neq \perp$, enough partial randomness can still be recovered from c to check that it decrypts to m.

Ciphertext verifiability. We next define a notion of *ciphertext verifiability*, which intuitively means a verifier can check (with the help of some partial random coins) whether the decryption algorithm returns $\perp$ on a given ciphertext. Let $\mathsf{PKE} = (\mathsf{Kg}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{pRec}, \mathsf{Cons})$ be a public-key encryption scheme with partial-randomness recovery. We say that PKE has *ciphertext-verifiability* if it also has a *ciphertext-invalidity checking algorithm* Inval such that

- (Completeness) For all $(pk, sk) \leftarrow\$ \mathsf{Kg}$ and all $c \in \{0,1\}^*$, let $s \leftarrow\$ \mathsf{pRec}(sk, c)$. Then
$$\mathsf{Dec}(sk, c) = \perp \ \Rightarrow \ \mathsf{Inval}(pk, c, s) = 1 \ .$$
- (Soundness) For all $(pk, sk) \leftarrow\$ \mathsf{Kg}$ and all $c \in \{0,1\}^*, s \in \{0,1\}^*$:
$$\mathsf{Inval}(pk, c, s) = 1 \ \Rightarrow \ \mathsf{Dec}(sk, c) = \perp \ .$$

We write $\mathsf{PKE} = (\mathsf{Kg}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{pRec}, \mathsf{Cons}, \mathsf{Inval})$.

We note that the notion of *public* ciphertext verifiability has been discussed informally in the literature and formalized and studied concurrently to our work by [32]. In a publicly verifiable scheme, the Inval algorithm ignores the input s. In fact, all our constructions except those that use the "more-efficient" BCHK transform [7] (the one using symmetric primitives rather than one-time signatures) are publicly verifiable.

We say that PKE is *ECCA-secure* if it is ECCA-secure as defined in Section 3 but where the Rec algorithm there is replaced with pRec.

PKENO-COMPATIBLE ECCA ENCRYPTION AND PKENO FROM IT. We call a partial-randomness recovering ECCA-secure PKE scheme with ciphertext verifiability *PKENO-compatible ECCA encryption.* Consider a PKENO-compatible ECCA encryption scheme $\mathsf{PKE} = (\mathsf{Kg}_{\mathrm{pke}}, \mathsf{Enc}_{\mathrm{pke}}, \mathsf{Dec}_{\mathrm{pke}}, \mathsf{pRec}_{\mathrm{pke}}, \mathsf{Cons}_{\mathrm{pke}}, \mathsf{Inval}_{\mathrm{pke}})$ and define the following PKENO $\mathsf{PKENO} = (\mathsf{Kg}_{\mathrm{pke}}, \mathsf{Enc}_{\mathrm{pke}}, \mathsf{Dec}_{\mathrm{pke}}, \mathsf{Prove}, \mathsf{Ver})$ with:

- $\mathsf{Prove}(sk, c)$: Output $s \leftarrow\$ \mathsf{pRec}_{\mathrm{pke}}(sk, c)$.
- $\mathsf{Ver}(pk, c, m, \pi = s)$: If $m \neq \perp$ then return 1 iff $\mathsf{Inval}_{\mathrm{pke}}(pk, c) = 0$ and $\mathsf{Cons}_{\mathrm{pke}}(pk, c, m, s) = 1$;
If $m = \perp$ then return 1 iff $\mathsf{Inval}_{\mathrm{pke}}(pk, c) = 1$.

Proposition 6.1 *Suppose* PKE *is a PKENO-compatible ECCA-secure encryption scheme. Then the construction* $\mathsf{PKENO}[\mathsf{PKE}]$ *above is a PKE scheme with non-interactive opening which is CCPA-rsecure and has proof soundness.*

Proof: For the case of proof correctness, we show that for $(pk, sk) \leftarrow\$ \mathsf{Kg}(1^k)$ and for all strings c,
$$\mathsf{Ver}(pk, c, \mathsf{Dec}(sk, c), \mathsf{Prove}(sk, c)) = 1 \ .$$
There are two cases to consider $\mathsf{Dec}(sk, c) = \perp$ and $\mathsf{Dec}(sk, c) \neq \perp$. If $\mathsf{Dec}(sk, c) = \perp$ then by completeness of ciphertext verifiability we have $\mathsf{Inval}(pk, c, s) = 1$ hence $\mathsf{Ver}(pk, c, \mathsf{Dec}(sk, c), \mathsf{Prove}(sk, c)) = 1$. On the other hand, if $m = \mathsf{Dec}(sk, c) \neq \perp$ then by soundness of ciphertext verifiability we know that if $\mathsf{Inval}(pk, c, s) = 1$ then $\mathsf{Dec}(sk, c) = \perp$, so $\mathsf{Inval}(pk, c, s) = 0$. Now by completeness of partial-randomness recovery if $\mathsf{Cons}(pk, c, m, s) = 0$ then $m = \perp$, so we know $\mathsf{Cons}(pk, c, m, s) = 1$ and hence $\mathsf{Ver}(pk, c, \mathsf{Dec}(sk, c), \mathsf{Prove}(sk, c)) = 1$.

For the case of proof soundness, in the relevant experiment in Figure 2, we show that
$$m \neq m' \Rightarrow \mathsf{Ver}(pk, c', m', \pi') = 0 \ .$$
Again we consider two cases, $m = \mathsf{Dec}(sk, c') = \perp$ and $m = \mathsf{Dec}(sk, c') \neq \perp$. If $m = \perp$ and $m' \neq \perp$, then if $\mathsf{Ver}(pk, c', m', \pi') = 1$ we would have $\mathsf{Cons}(pk, c', m', \pi') = 1$ which by soundness of partial-randomness recovery means $\mathsf{Dec}(sk, c') = m'$, a contradiction. On the other hand, if $m \neq \perp$, we again split into two cases: $m' = \perp$ and $m' \neq \perp$. If $m' = \perp$ then if $\mathsf{Inval}(pk, c', \pi') = 1$ then by soundness of ciphertext verifiability we would have $\mathsf{Dec}(sk, c') = m = \perp$, so $\mathsf{Inval}(pk, c', \pi') = 0$ and thus verification fails. Finally, if both $m \neq \perp$ and $m' \neq \perp$ but $m \neq m'$ then if $\mathsf{Ver}(pk, c', m', \pi') = 1$ we have $\mathsf{Cons}(pk, c', m', \pi')$ which by soundness of partial-randomness recovery means $\mathsf{Dec}(sk, c') = m'$, a contradiction.

IND-CCPA follows immediately from ECCA w.r.t. $\mathsf{pRec}_{\mathrm{pke}}$. The simulator can use its $\mathsf{pRec}_{\mathrm{pke}}$ oracle to simulate the Prove oracle. To simulate the Dec oracle on a query c, the simulator queries $\mathsf{pRec}_{\mathrm{pke}}$ for c to get s and outputs $\perp$ if $\mathsf{Val}_{\mathrm{pke}}(pk, c, s) = 0$ and otherwise forwards a $\mathsf{Dec}_{\mathrm{pke}}$ oracle response. ▌

THE TAG-BASED CASE. We define *PKENO-compatible tag-based ECCA encryption* $\mathsf{TB\text{-}PKE} = (\mathsf{Kg}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{pRec}, \mathsf{Cons}, \mathsf{Inval})$ analogously, where all algorithms except Kg now take an additional input t called the *tag*. The completeness and soundness definitions quantify over all tags t, and the scheme is required to satisfy the tag-based ECCA definition where Rec is replaced with pRec. One can convert any PKENO-compatible tag-based ECCA encryption scheme into a PEKNO-compatible (non-tag-based) ECCA encryption scheme by using BCHK transform [7], which preserves both partial-randomness recovery and ciphertext verifiability. Either version of the BCHK transform works, though it is interesting to that that ciphertext verifiability when using the "more-efficient" version based on symmetric-key primitives uses the fact that the Inval algorithm take as input partial randomness (since unlike the version using one-time signatures the symmetric-key primitives are not publicly verifiable). Details are in Appendix C. Hence, as per the above, PKENO-compatible tag-based ECCA encryption also yields PKENO. Looking ahead, using tag-based constructions will lead to our most efficient PKENO schemes.

6.2 A Generic Construction using Non-Interactive Zero-Knowledge

We show how to obtain PKENO-compatible ECCA encryption by combining partially randomness-recovering (pRR) ECCA secure encryption with a non-interactive zero-knowledge proof (NIZK) to add ciphertext verifiability. (Indeed, the approach of adding a proof of well-formedness originates from [13, 18], although not with respect to ECCA secure encryption.)

CONSTRUCTION. Consider a pRR ECCA-secure PKE scheme $\mathsf{PKE} = (\mathsf{Kg}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{pRec}, \mathsf{Cons})$ and a simulation-sound NIZK proof system $\mathsf{NIZK} = (\mathsf{Setup}, \mathsf{Prv}, \mathsf{Vrf})$ for the language $\mathcal{L} := \{(pk, c) \,|\, \exists (m, r) : c = \mathsf{Enc}(pk, m; r)\}$. We define a partial-randomness-recovering scheme $\mathsf{PKE}_{\rm prr} = (\mathsf{Kg}_{\rm prr}, \mathsf{Enc}_{\rm prr}, \mathsf{Dec}_{\rm prr}, \mathsf{Cons}_{\rm prr}, \mathsf{Inval}_{\rm prr})$ as follows:

- $\mathsf{Kg}_{\rm prr}(1^k)$: Run $crs \leftarrow_{\$} \mathsf{Setup}(1^k)$; $(pk, sk) \leftarrow_{\$} \mathsf{Kg}(1^k)$. Output $\overline{pk} := (crs, pk)$ and $\overline{sk} := (crs, pk, sk)$.
- $\mathsf{Enc}_{\rm prr}((crs, pk), m; r)$: Set $c := \mathsf{Enc}(pk, m; r)$ and $\tau \leftarrow_{\$} \mathsf{Prv}(crs, (pk, c), (m, r))$; output $\overline{c} := (c, \tau)$.
- $\mathsf{Dec}_{\rm prr}((crs, pk, sk), (c, \tau))$: If $\mathsf{Vrf}(crs, (pk, c), \tau) = 0$ then output $\perp$; else output $\mathsf{Dec}(sk, c)$.
- $\mathsf{pRec}_{\rm prr}((crs, pk, sk), (c, \tau))$: If $\mathsf{Vrf}(crs, (pk, c), \tau) = 0$ then output $\perp$; else output $\mathsf{pRec}(sk, c)$.
- $\mathsf{Cons}_{\rm prr}((crs, pk), (c, \tau), m, s)$: Return 1 iff $\mathsf{Vrf}(crs, (pk, c), \tau) = 1$ and $\mathsf{Cons}(pk, c, m, s) = 1$.
- $\mathsf{Inval}_{\rm prr}((crs, pk), (c, \tau), s)$: Return 1 iff $\mathsf{Vrf}(crs, (pk, c), \tau) = 0$.

Proposition 6.2 *Suppose* PKE *is partial-randomness-recovering and ECCA-secure and* NIZK *is simulation-sound and zero-knowledge. Then* $\mathsf{PKE}_{\rm prr}[\mathsf{PKE}, \mathsf{NIZK}]$*, defined above, is an ECCA-secure PKE scheme with partial randomness-recovery and public verifiability (i.e., it is PKENO-compatible).*

Proof: We first show that the scheme $\mathsf{PKE}_{\rm prr}$ is still ECCA-secure (with respect to the partial randomness recovery function pRec). Let A be an adversary against ECCA security of $\mathsf{PKE}_{\rm prr}$. We use A to break ECCA security of the underlying scheme PKE. Upon receiving our challenge c^*, we simulate a NIZK proof τ^* to create a challenge (c^*, τ^*) for A. Consider A making an ECCA-query Dec^* for (c, τ): if τ is invalid on c, we return $\perp$; else we forward c to our own oracle and give A the reply. Note that simulation soundness of the NIZK implies that A cannot produce a query (c^*, τ) for a $\tau \neq \tau^*$ which is valid for c^*. (This is formally proven analogously to the proof of full anonymity of the group signature construction in [3], where a group signature is defined as a CCA-secure ciphertext and a simulation-sound NIZK proof of well-formedness of the ciphertext.) This means that every query A makes can be forwarded to our own oracle.

Finally, we need to show completeness and soundness of both partial randomness recovery and ciphertext verifiability, as defined in Section 6.1. For all these notions, let $(\overline{pk} = (crs, pk), \overline{sk})$ be the output of $\mathsf{Kg}_{\rm prr}$.

Completeness of partial-randomness recovery: Let $c \in \{0,1\}^*$ and let $s \leftarrow_{\$} \mathsf{pRec}_{\rm prr}(sk, c)$. Suppose $\mathsf{Dec}_{\rm prr}(sk, c) = m$ with $m \neq \perp$. By the definition of $\mathsf{Dec}_{\rm prr}$ we get $\mathsf{Vrf}(crs, (pk, c), \tau) = 1$ $(*)$. By the definition of $\mathsf{pRec}_{\text{prr}}$, this means s is the output of $\mathsf{pRec}(sk, c)$. By completeness of Cons, we have $\mathsf{Cons}(pk, c, m, s) = 1$, and thus together with $(*)$: $\mathsf{Cons}_{\text{prr}}(\overline{pk}, (c, \tau), m, s) = 1$.

Soundness of partial-randomness recovery: Let $c \in \{0,1\}^*, m \in MsgSp(1^k)$ and $s \in \{0,1\}^*$. Suppose $\mathsf{Cons}_{\text{prr}}(\overline{pk}, (c, \tau), m, s) = 1$, that is $\mathsf{Vrf}(crs, (pk, c), \tau) = 1$ $(*)$ and $\mathsf{Cons}(pk, c, m, s) = 1$. By soundness of Cons we have $\mathsf{Dec}(sk, c) = m$, which together with $(*)$ yields: $\mathsf{Dec}_{\text{prr}}(\overline{sk}, (c, \tau)) = m$.

Completeness of ciphertext verifiability: Let $c \in \{0,1\}^*$ and let $s \leftarrow\!\$\ \mathsf{pRec}_{\text{prr}}(sk, c)$. Suppose $\mathsf{Dec}_{\text{prr}}(sk, (c, \tau)) = \perp$. Then we have either (1) $\mathsf{Vrf}(crs, (pk, c), \tau) = 0$ or (2) $\mathsf{Dec}(sk, c) = \perp$. We show (2) implies (1): If Vrf output 1 then by soundness of NIZK there exist $(m, r) : c = \mathsf{Enc}(pk, m; r)$. By correctness of PKE we have $\mathsf{Dec}(sk, c) = m$ which contradicts (1). In either case we have therefore $\mathsf{Vrf}(crs, (pk, c), \tau) = 0$ and thus $\mathsf{Inval}_{\text{prr}}(\overline{pk}, (c, \tau), s) = 1$.

Soundness of ciphertext verifiability: Let $c \in \{0,1\}^*$ and $s \in \{0,1\}^*$ and assume $\mathsf{Inval}_{\text{prr}}(\overline{pk}, (c, \tau), s) = 1$, that is, $\mathsf{Vrf}(crs, (pk, c), \tau) = 0$. Then by definition of $\mathsf{Dec}_{\text{prr}}$, we have $\mathsf{Dec}_{\text{prr}}(\overline{sk}, (c, \tau)) = \perp$. ▌

Constructing PKENO-compatible tag-based PKE using NIZK. An analogous construction works to construct PKENO-compatible ECCA-secure tag-based PKE from partial-randomness recovering ECCA-secure tag-based PKE. The latter can be converted into PKENO-compatible ECCA encryption using one-time signatures or symmetric primitives as explained in Appendix C.

6.3 Efficient CCA-secure PKENO-Compatible Tag-Based PKE using Groth-Sahai

It is possible to construct NIZKs efficiently in the random-oracle model, but the only known efficient instantiation of NIZKs without random oracles to date are Groth-Sahai (GS) proofs [22], which allow to prove a certain class of statements over bilinear groups. Moreover, as shown in [30], GS proofs can be made simulation-sound (SS) at a moderate price. We show how to build an efficient DLIN-based partial-randomness recovering tag-based PKE scheme such that the statements we are proving fall in this class.

Essentially, the idea is to build a decision-linear (DLIN) based tag-based ATDF for which proving that a claimed image point has a preimage is GS statement. We then build from it a PKENO-compatible tag-based PKE scheme as per Appendix C. Note that the GS statement here is *weaker* than a proof of well-formedness of the ciphertext for the CCA scheme (as used in our generic construction above), so it is somewhat surprising that it suffices for a PKENO construction. (The intuition is that, if the proof verifies, the adversary is obliged to *provide* the preimage, which allows to verify the rest of the ciphertext if needed.)

Specifically, we use the lossy trapdoor function (LTDF) and an all-but-one (ABO) TDF from DLIN by Freeman *et al.* [16], which can be combined to a tag-based ATDF using the transformation of [28]. Note that we use the the DLIN-based instantiations rather than the original DDH-based ones of [33] because Groth-Sahai proofs require bilinear groups in which DDH is false.

We first describe the ABO-TDF. In a group $\mathbb{G}$ of order p, generated by $g \in \mathbb{G}$, the scheme is defined as follows. To sample a function with a lossy branch b^*, choose a matrix $A \leftarrow\!\$\ \mathbb{Z}_p^{n \times n}$ with rank 1 and define $M := A - b^* I_n$, where I_n is the identity matrix. Define $\mathbf{S} \in \mathbb{G}^{n \times n}$ as the matrix with components $S_{ij} := g^{m_{ij}}$, where m_{ij} are the components of M. The function indexed by $\mathbf{S}$ is evaluated on a branch b on input $\vec{x} \in \{0,1\}^n$ as follows: $f_{\mathbf{S},b}(\vec{x}) \in \mathbb{G}^n$ is the vector whose i-th component is defined as $\prod S_{ij}^{x_j} \cdot g^{b \cdot x_i}$. In the language of Groth-Sahai proofs, interpreting $f_i = \prod S_{ij}^{x_j} \cdot g^{b \cdot x_i}$ as an equation over variables $x_1, \ldots, x_n$, this is a *multi-exponentiation equation* (called multi-scalar multiplication equation when using additive notation, as in [22]). For a set of such equations over the same variables, Groth-Sahai proofs provide an efficient proof of satisfiability; thus, a proof that a value has a preimage under $f_{\mathbf{S},b}$.

The same holds for the LTDF, which is a special case of the above for $b = 0$.

In our construction we combine these into a TB-ATDF using a target-collision-resistant hash function T as per [28] as follows. A public key is a pair of keys $(ek_{\text{ltf}}, ek_{\text{abo}})$ for LF and $\mathsf{ABO\text{-}F}$, respectively. The evaluation of ATDF on input $x \in \{0,1\}^k$ and tag t under key $(ek_{\text{ltf}}, ek_{\text{abo}})$ is defined as (y_1, y_2) with

$$y_1 := \mathsf{LF}(ek_{\text{ltf}}, x) \qquad y_2 := \mathsf{ABO\text{-}F}(T(t), ek_{\text{abo}}, x)$$

Adding Groth-Sahai proofs that for both y_1 and y_2 there exists a preimage x (which is the *same* for both), this gives us a TB-ATDF with *public verifiability of preimage existence.*

Finally, using the construction of ECCA PKE from tag-based ECCA PKE in Section 4.3, we get a PKENO-compatible ECCA-secure tag-based PKE scheme.

6.4 PKENO-Compatible ECCA Encryption from Range-Verifiable ATDFs

Although our DLIN-based instantiation avoids using generic NIZKs or random oracles, it is probably not efficient enough for practical use. Here we show that using a tag-based ATDF with an additional property we call *range verifiability*, we can directly obtain a PKENO-compatible ECCA-secure tag-based PKE scheme *without using NIZK at all.* Intuitively, this property allows us to check the property that the Groth-Sahai statement in our above construciton is proving, so that there is no need for a separate proof. We show that the tag-based ATDF constructed in [28] from the instance-independent RSA assumption (II-RSA) has this property.

Range verifiable tag-based TDFs. We call a tag-based TDF *range verifiable* if it can be augmented by an efficient algorithm RVrf such that or any key pair (ek, sk) and any tuple (t, c), $\mathsf{RVrf}(ek, t, c)$ returns 1 if and only if there exists x such that $c = \mathsf{Eval}(ek, t, x)$.

Proposition 6.3 *The* $\mathsf{TB\text{-}PKE[TB\text{-}TDF]}$ *construction from Section 4.3 is PKENO-compatible if the underlying* $\mathsf{TB\text{-}TDF}$ *is a range-verifiable tag-based ATDF.*

Proof sketch. The proof is quite straightforward. We simply need to show that given an efficient RVrf for the $\mathsf{TB\text{-}TDF}$, it is possible to design an efficient RVrf' for the PKE. We define RVrf' such that on pk, t, and the ciphertext $((c_{1,1}, c_{1,2}), \ldots, (c_{\ell,1}, c_{\ell,2}))$ as in Section 4.3, it runs RVrf on $pk, t, c_{i,1}$ for $1 \le i \le \ell$ and returns 1 if and only if all invocations of RVrf return 1.

Range verifiable tag-based ATDF from II-RSA. The tag-based ATDF of [28] based on the instance-independent RSA (II-RSA) assumption is a one example of such TDFs. The domain and range of this tag-based ATDF (which is in fact a permutation) is $\mathbb{Z}_N^*$, i.e., the group of multiplicatively invertible elements modulo $N = pq$. In this case the associated range verifying function $\mathsf{RVrf}(ek, t, c)$ returns 1 if and only if $c < N$. Note that we need to relax our definitions to allow completeness and soundness to consider PPT generated inputs and fail with negligible probability, since the adversary may produce a c in $\mathbb{Z}_N \setminus \mathbb{Z}_N^*$ with negliglible probability assuming factoring N is hard. (For simplicitly, we do not make these relaxations in our formal definitions.)

The resulting PKENO scheme based on II-RSA given by Proposition 6.3 is quite efficient, requiring only one 512-bit exponentiation to encrypt. Indeed, it is exactly the same RSA-based CCA PKE scheme from [28], which was already of note for its efficiency even without showing that it is also a PKENO. In terms of efficiency this compares favorably to the previous DLIN-based construction of [18] which requires multiple exponentiations (but is secure under a more standard assumption).

References

[1] 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. 4

[2] M. Bellare, D. Hofheinz, and S. Yilek. Possibility and impossibility results for encryption and commitment secure under selective opening. In A. Joux, editor, *EUROCRYPT 2009*, volume 5479 of *LNCS*, pages 1–35. Springer, Apr. 2009. 4

[3] M. Bellare, D. Micciancio, and B. Warinschi. Foundations of group signatures: Formal definitions, simplified requirements, and a construction based on general assumptions. In E. Biham, editor, *EUROCRYPT 2003*, volume 2656 of *LNCS*, pages 614–629. Springer, May 2003. 20

[4] M. Bellare and A. Palacio. Towards plaintext-aware public-key encryption without random oracles. In P. J. Lee, editor, *ASIACRYPT 2004*, volume 3329 of *LNCS*, pages 48–62. Springer, Dec. 2004. 4

[5] M. Bellare and P. Rogaway. Optimal asymmetric encryption. In A. D. Santis, editor, *EUROCRYPT'94*, volume 950 of *LNCS*, pages 92–111. Springer, May 1994. 4

[6] M. Bellare and S. Yilek. Encryption schemes secure under selective opening attack. Cryptology ePrint Archive, Report 2009/101, 2009. `http://eprint.iacr.org/`. 4

[7] D. Boneh, R. Canetti, S. Halevi, and J. Katz. Chosen-ciphertext security from identity-based encryption. *SIAM Journal on Computing*, 36(5):1301–1328, 2007. 2, 3, 4, 14, 15, 19, 20, 27

[8] R. Canetti, U. Feige, O. Goldreich, and M. Naor. Adaptively secure multi-party computation. In *28th ACM STOC*, pages 639–648. ACM Press, May 1996. 4

[9] R. Canetti, H. Krawczyk, and J. B. Nielsen. Relaxing chosen-ciphertext security. In D. Boneh, editor, *CRYPTO 2003*, volume 2729 of *LNCS*, pages 565–582. Springer, Aug. 2003. 4

[10] R. Canetti, H. Lin, and R. Pass. Adaptive hardness and composable security in the plain model from standard assumptions. In *51st FOCS*, pages 541–550. IEEE Computer Society Press, Oct. 2010. 4, 6

[11] S. G. Choi, D. Dachman-Soled, T. Malkin, and H. Wee. Black-box construction of a non-malleable encryption scheme from any semantically secure one. In R. Canetti, editor, *TCC 2008*, volume 4948 of *LNCS*, pages 427–444. Springer, Mar. 2008. 10

[12] 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. 25, 26

[13] I. Damgård, D. Hofheinz, E. Kiltz, and R. Thorbek. Public-key encryption with non-interactive opening. In T. Malkin, editor, *CT-RSA 2008*, volume 4964 of *LNCS*, pages 239–255. Springer, Apr. 2008. 1, 2, 17, 18, 20

[14] I. Damgård and R. Thorbek. Non-interactive proofs for integer multiplication. In M. Naor, editor, *EUROCRYPT 2007*, volume 4515 of *LNCS*, pages 412–429. Springer, May 2007. 2, 3, 17, 18

[15] C. Dwork, M. Naor, O. Reingold, and L. J. Stockmeyer. Magic functions. *Journal of the ACM*, 50(6):852–921, 2003. 4

[16] D. M. Freeman, O. Goldreich, E. Kiltz, A. Rosen, and G. Segev. More constructions of lossy and correlation-secure trapdoor functions. In P. Q. Nguyen and D. Pointcheval, editors, *PKC 2010*, volume 6056 of *LNCS*, pages 279–295. Springer, May 2010. 3, 21

[17] D. Galindo. Breaking and repairing Damgård et al. public key encryption scheme with non-interactive opening. In M. Fischlin, editor, *CT-RSA 2009*, volume 5473 of *LNCS*, pages 389–398. Springer, Apr. 2009. 2, 17

[18] D. Galindo, B. Libert, M. Fischlin, G. Fuchsbauer, A. Lehmann, M. Manulis, and D. Schröder. Public-key encryption with non-interactive opening: New constructions and stronger definitions. In D. J. Bernstein and T. Lange, editors, *AFRICACRYPT 10*, volume 6055 of *LNCS*, pages 333–350. Springer, May 2010. 2, 3, 4, 17, 18, 20, 22, 28, 29

[19] R. Gennaro and V. Shoup. A note on an encryption scheme of Kurosawa and Desmedt. Cryptology ePrint Archive, Report 2004/194, 2004. `http://eprint.iacr.org/`. 15

[20] O. Goldreich and L. A. Levin. A hard-core predicate for all one-way functions. In *21st ACM STOC*, pages 25–32. ACM Press, May 1989. 8

[21] S. Goldwasser and S. Micali. Probabilistic encryption. *Journal of Computer and System Sciences*, 28(2):270–299, 1984. 24

[22] J. Groth and A. Sahai. Efficient non-interactive proof systems for bilinear groups. In N. P. Smart, editor, *EUROCRYPT 2008*, volume 4965 of *LNCS*, pages 415–432. Springer, Apr. 2008. 3, 21

[23] B. Hemenway, B. Libert, R. Ostrovsky, and D. Vergnaud. Lossy encryption: Constructions from general assumptions and efficient selective opening chosen ciphertext security. In D. H. Lee and X. Wang, editors, *ASIACRYPT 2011*, volume 7073 of *LNCS*, pages 70–88. Springer, Dec. 2011. 4

[24] D. Hofheinz. All-but-many lossy trapdoor functions. In *EUROCRYPT*, pages 209–227, 2012. 4

[25] D. Hofheinz and E. Kiltz. Secure hybrid encryption from weakened key encapsulation. In A. Menezes, editor, *CRYPTO 2007*, volume 4622 of *LNCS*, pages 553–571. Springer, Aug. 2007. 4, 25, 26

[26] S. Hohenberger, A. B. Lewko, and B. Waters. Detecting dangerous queries: A new approach for chosen ciphertext security. page 663. 2, 4, 7, 8, 9, 10

[27] S. A. Kakvi, A. May, and E. Kiltz. Certifying rsa. 29

[28] E. Kiltz, P. Mohassel, and A. O'Neill. Adaptive trapdoor functions and chosen-ciphertext security. In H. Gilbert, editor, *EUROCRYPT 2010*, volume 6110 of *LNCS*, pages 673–692. Springer, May 2010. 1, 2, 3, 4, 7, 14, 15, 21, 22, 24, 25, 29

[29] P. D. MacKenzie, M. K. Reiter, and K. Yang. Definitions, constructions, and applications (extended abstract). In M. Naor, editor, *TCC 2004*, volume 2951 of *LNCS*, pages 171–190. Springer, Feb. 2004. 16, 25

[30] H. K. Maji, M. Prabhakaran, and M. Rosulek. Attribute-based signatures. In A. Kiayias, editor, *CT-RSA 2011*, volume 6558 of *LNCS*, pages 376–392. Springer, Feb. 2011. 3, 21

[31] S. Myers and A. Shelat. Bit encryption is complete. In *50th FOCS*, pages 607–616. IEEE Computer Society Press, Oct. 2009. 1, 2

[32] J. M. G. Nieto, M. Manulis, B. Poettering, J. Rangasamy, and D. Stebila. Publicly verifiable ciphertexts. Cryptology ePrint Archive, Report 2012/357, 2012. `http://eprint.iacr.org/`. 19

[33] C. Peikert and B. Waters. Lossy trapdoor functions and their applications. In R. E. Ladner and C. Dwork, editors, *40th ACM STOC*, pages 187–196. ACM Press, May 2008. 1, 5, 7, 18, 21

[34] 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. 1

[35] A. Rosen and G. Segev. Chosen-ciphertext security via correlated products. In O. Reingold, editor, *TCC 2009*, volume 5444 of *LNCS*, pages 419–436. Springer, Mar. 2009. 1, 7

[36] H. Wee. Efficient chosen-ciphertext security via extractable hash proofs. In T. Rabin, editor, *CRYPTO 2010*, volume 6223 of *LNCS*, pages 314–332. Springer, Aug. 2010. 1

[37] A. C. Yao. Theory and applications of trapdoor functions. In *23rd FOCS*, pages 80–91. IEEE Computer Society Press, Nov. 1982. 24

A Standard Primitives

Trapdoor functions. A *trapdoor function family* [37] is a triple of algorithms $\mathsf{TDF} = (\mathsf{Tdg}, \mathsf{Eval}, \mathsf{Inv})$. The key-generation algorithm Tdg returns an evaluation key ek and matching trapdoor td. The deterministic evaluation algorithm Eval takes ek and $x \in \{0,1\}^k$ to return an image y. The deterministic inversion algorithm Inv takes td and y to return a point x. We require that for all $x \in \{0,1\}^k$,

$$\Pr[\mathsf{Inv}(td, \mathsf{Eval}(ek, x)) = x \,:\, (ek, td) \leftarrow{\$}\, \mathsf{Tdg}(1^k)]$$

with probability 1. We say that TDF is *tag-based* [28] with tag-space TagSp if $\mathsf{Eval}, \mathsf{Inv}$ take an additional input $t \in \mathit{TagSp}$ called the *tag* and for all $x \in \{0,1\}^k$ and $t \in \mathit{TagSp}(1^k)$,

$$\Pr[\mathsf{Inv}(td, t, \mathsf{Eval}(ek, t, x)) = x \,:\, (ek, td) \leftarrow{\$}\, \mathsf{Tdg}(1^k)]$$

with probability 1.

Public-key encryption. A *public-key encryption scheme* [21] with message-space MsgSp is a triple of algorithms $\mathsf{PKE} = (\mathsf{Kg}, \mathsf{Enc}, \mathsf{Dec})$. The key-generation algorithm Kg returns a public key pk and matching secret key sk. The encryption algorithm Enc takes pk and a plaintext m to return a ciphertext. The deterministic decryption algorithm Dec takes sk and a ciphertext c to return a plaintext. We require that for all $k \in \mathbb{N}$ and $m \in \mathit{MsgSp}(1^k)$,

$$\Pr[\mathsf{Dec}(sk, \mathsf{Enc}(pk, m)) \neq m \,:\, (pk, sk) \leftarrow{\$}\, \mathsf{Kg}(1^k)]$$

is negligible. We say that PKE is *tag-based* [29] with tag-space $TagSp$ if $\mathsf{Enc}, \mathsf{Dec}$ take an additional input $t \in TagSp(1^k)$ called the *tag* and for all $k \in \mathbb{N}$, $m \in MsgSp(1^k)$, and $t \in TagSp(1^k)$,

$$\Pr[\mathsf{Dec}(sk, t, \mathsf{Enc}(pk, t, m)) = m \,:\, (pk, sk) \leftarrow\!\$\, \mathsf{Kg}(1^k)]$$

with probability 1.

Authenticated Encryption. We use the definition of [25] for authenticated encryption. An authenticated symmetric encryption (AE) scheme $\mathsf{AE} = (\mathsf{AE.Enc}, \mathsf{AE.Dec})$ is specified by its encryption algorithm $\mathsf{AE.Enc}$ (encrypting $m \in MsgSp(k)$ with a key $K \in K(k)$) and decryption algorithm $\mathsf{AE.Dec}$ (returning $m \in MsgSp(k)$ or $\perp$). Here we restrict ourselves to deterministic $\mathsf{AE.Enc}$ and $\mathsf{AE.Dec}$. The AE scheme needs to provide privacy (indistinguishability against one-time attacks) and authenticity (ciphertext authenticity against one-time attacks). This is simultaneously captured by defining the *ae-ot-advantage* of an adversary A as:

$$\mathbf{Adv}^{\text{ot-ae}}_{A,\mathsf{AE}}(k) = 2 \cdot \Pr[b = b' \,:\, K \leftarrow\!\$\, K(k) \;;\; b \leftarrow\!\$\, \{0,1\} \;;\; b' \leftarrow\!\$\, A^{\mathsf{LoR}_b(\cdot,\cdot),\mathsf{DoR}_b(\cdot)}(1^k)] - 1 \,.$$

Here, $\mathsf{LoR}_b(m_0, m_1)$ returns $\mathsf{AE.Enc}(K, m_b)$, and A is allowed only one query to this left-or-right encryption oracle, with a pair of equal length messages. Furthermore, the decrypt-or-reject oracle $\mathsf{DoR}_0(C)$ returns $m \leftarrow \mathsf{AE.Dec}(K, C)$ and $\mathsf{DoR}_1(C)$ always returns $\perp$ (reject). A is allowed only one query to this oracle which must be different from the output of the left-or-right oracle. An encryption scheme is a *one-time authenticated encryption* if $\mathbf{Adv}^{\text{ot-ae}}_{A,\mathsf{AE}}(k)$ is negligible for any PPT adversary A.

Key Encapsulation Mechanisms. The KEM/DEM paradigm was first formalized in [12]. We borrow our formal definitions from [25]. A key-encapsulation mechanism $\mathsf{KEM} = (\mathsf{KEM.kg}, \mathsf{KEM.enc}, \mathsf{KEM.enc})$ with key-space $K(k)$ consists of three polynomial-time algorithms. Via $(pk, sk) \leftarrow\!\$\, \mathsf{KEM.kg}(1^k)$ the randomized key-generation algorithm produces public/secret keys for security parameter k; via $(K, C) \leftarrow\!\$\, \mathsf{KEM.enc}(pk)$, the randomized encapsulation algorithm creates a uniformly distributed symmetric key $K \in K(k)$ together with a ciphertext C; via $K \leftarrow \mathsf{KEM.dec}(sk, C)$ the possessor of secret key sk decrypts ciphertext C to get back a key K which is an element in K or a special rejection symbol $\perp$. For consistency, we require that for all all $(K, C) \leftarrow\!\$\, \mathsf{KEM.enc}(pk)$ we have $\Pr[\mathsf{KEM.dec}(sk, C) = K] = 1$, where the probability is taken over the choice of $(pk, sk) \leftarrow\!\$\, \mathsf{KEM.kg}(1^k)$, and the coins of all the algorithms in the expression above. Here we only consider KEMs that produce perfectly uniformly distributed keys (i.e., we require that for all public keys pk that can be output by $\mathsf{KEM.kg}$, the first component of $\mathsf{KEM.enc}(pk)$ has uniform distribution).

```
Experiment $\mathbf{Exp}^{\text{kem-cca}}_{\mathsf{KEM},A}(k)$
    $b \leftarrow\!\$\, \{0,1\}$ ; $(pk, sk) \leftarrow\!\$\, \mathsf{KEM.kg}(1^k)$
    $K^*_0 \leftarrow\!\$\, K(k)$; $(K^*_1, C^*) \leftarrow\!\$\, \mathsf{KEM.enc}(pk)$
    $d \leftarrow\!\$\, A_1^{\mathsf{KEM.dec}^*(sk,\cdot)}(pk, K^*_b, C^*)$
    If $d = b$ then return 1 else return 0

Oracle $\mathsf{KEM.dec}^*(sk, C)$
    $K \leftarrow \mathsf{KEM.dec}(sk, C)$
    Return $K$
```

Above we require that A does not query c^* to its oracle. Define the *kem-cca advantage* of A against KEM as

$$\mathbf{Adv}^{\text{ekm-cca}}_{\mathsf{KEM},A}(k) = 2 \cdot \Pr\left[\mathbf{Exp}^{\text{kem-cca}}_{\mathsf{KEM},A}(k) \text{ outputs } 1\right] - 1 \,.$$

We say that KEM is *chosen-ciphertext secure* if $\mathbf{Adv}^{\text{kem-cca}}_{\mathsf{KEM},A}(\cdot)$ is negligible for every efficient A.

B From ATDF to ECCA via KEM/DEM

The above constructions assume that the ATDF/tb-ATDF only provides us with a single hardcore bit. But, if a linear number of hardcore bits are available (e.g. based on Lossy TDFs as discussed in [28]), one can design significantly more efficient ECCA PKE constructions. The basic idea is simple: we use the ATDF/tb-ATDF with linear hardcore bits to encrypt a one-time secret key k via a key encapsulation mechanism (KEM), and use k to encrypt the message via a data encapsulation mechanism (DEM). The standard KEM/DEM paradigm guarantees that the resulting hybrid PKE scheme is CCA secure if the KEM component and the DEM component are both CCA secure[6] (e.g. see [12]).

In our case, however, we need the hybrid PKE to be ECCA secure and randomness recovering as well. One can construct a KEM based on a ATDF by simply using the hardcore bits as the one-time key. It is easy to see that this construction is both ECCA secure and randomness recovering. The natural next step is to use a CCA (or ECCA) DEM component to obtain a hybrid PKE with the desired properties.

CCA security of DEM is not sufficient. Surpringly, this does not work: hybrid encryption does *not* preserve the ECCA security of the KEM. Roughly speaking, the subtlety in the proof arises when the simulator needs to answer decryption queries for ciphertexts that have the same KEM component as the challenge ciphertext but a different DEM component. In the standard proof, such decryption queries are answered by decrypting the DEM component and returning the message (without having to decrypt the KEM component). But, to achieve ECCA security, we need to return all the randomness to adversary, including those used in the KEM component. To solve this we instead use a a one-time *authenticated encryption* scheme as the DEM, in which case we show the resulting hybrid PKE is ECCA secure and randomness recovering.

An ECCA KEM/DEM Construction from ATDFs. Consider the following construction based on any ATDF with linear hardcore bits. Let $\mathsf{TDF} = (\mathsf{Tdg}, \mathsf{Eval}, \mathsf{Inv})$ be a trapdoor function with a hardcore function hc, and $\mathsf{AE} = (\mathsf{AE.Kg}, \mathsf{AE.Enc}, \mathsf{AE.Dec})$ be a deterministic authenticated encryption scheme. Define the following multi-bit public-key encryption scheme $\mathsf{PKE}[\mathsf{TDF}] = (\mathsf{Kg}, \mathsf{Enc}, \mathsf{Dec})$:

```
Alg Kg($1^k$)
    $(ek, td) \leftarrow\$ \mathsf{Tdg}(1^k)$
    Return $(ek, td)$

Alg Enc($ek, m$)
    $x \leftarrow\$ \{0,1\}^k$
    $y_1 \leftarrow \mathsf{Eval}(ek, x)$
    $y_2 \leftarrow \mathsf{AE.Enc}(\mathsf{hc}(x), m)$
    Return $(y_1, y_2)$

Alg Dec($td, ((y_1, y_2), \mathsf{flag})$)
    If $\mathsf{flag} = 1$ then return $(y_1, y_2)$
    Else $K \leftarrow \mathsf{hc}(\mathsf{Inv}(td, y_1))$
    Return $(x, \mathsf{AE.Dec}(k, y_2))$
```

Proposition B.1 *Suppose* TDF *is adaptive one-way and* AE *is a one-time authenticated encryption. Then* $\mathsf{PKE}[\mathsf{TDF}]$ *defined above is ECCA-secure and randomness-recovering.*

Proof: It is easy to see that the above construction is randomness-recovering. In particular, the decryption algorithm recovers $x = \mathsf{Inv}(td, y_1)$, computes $K = \mathsf{hc}(x)$ and then uses K to recover the message encrypted in y_2. Since the AE is deterministic, there is no additional randomness to recover.

Next, we show that $\mathsf{PKE}[\mathsf{TDF}]$ is also ECCA. We take advantage of the deferred analysis technique in this proof as well. Consider the following sequence of games:

Hybrid H_0: The first game is the *ind-cca* experiment for $\mathsf{PKE}[\mathsf{TDF}]$. Denote the challenge ciphertext by $c^* = (c_1^*, c_2^*)$, and the secret key used to encrypt c_2^* by k^*.

Hybrid H_1: H_1 is the same as H_0 except that on decryption queries of the form $c = (c_1^*, c_2)$ by the adversary we return $\perp$.

Note that $|\mathbf{Adv}_A^{H_0}(k) - \mathbf{Adv}_A^{H_1}(k)| < \Pr_1[valid]$, where *valid* is the event that in game H_1, for some decryption query $c \neq c^*$ where $c_1 = c_1^*$, c is a valid ciphertext. Assuming no decryption error, this probability is bounded by $\Pr_1[forge]$ where $forge$ is the event that c_2 is valid ciphertext for the AE scheme. We postpone the analysis of the bound on $\Pr_1[forge]$ until the final hybrid.

Hybrid H_2: H_2 is the same as H_1, except that in the challenge ciphertext, we use a uniformly random key K' for the AE scheme.

[6]It is possible to relax the security requirement for the KEM component but the CCA security of the DEM seems necessary in order to obtain a CCA secure PKE (see [25]).

Lets denote by KEM, the KEM component of the above construction. It is easy to see that $|\mathbf{Adv}_A^{H_2}(k) - \mathbf{Adv}_A^{H_1}(k)| \leq \mathbf{Adv}_{B,\mathsf{KEM}}^{\text{kem-cca}}$. Note that the same bound is true for $|\Pr_2[forge] - \Pr_1[forge]|$.

Hybrid H_3: H_3 is the same as H_2, except that instead of encrypting m_b for the challenge ciphertext (in the DEM component), we encrypt $0^{|m_0|}$.

Obviously, $\mathbf{Adv}_A^{H_3}(k) = 1/2$. It is also not hard to see that $|\mathbf{Adv}_A^{H_3}(k) - \mathbf{Adv}_A^{H_2}(k)| \leq \mathbf{Adv}_{C,\mathsf{AE}}^{\text{ot-ae}}(k)$. The same bound is true for $|\Pr_3[forge] - \Pr_2[forge]|$ as well.

The last thing we need to show is that $\Pr_3[forge]$ is negligible, but now the key K' for the AE is generated at random, we have that $\Pr_3[forge] < \mathbf{Adv}_{C,\mathsf{AE}}^{\text{ot-ae}}(k)$, hence concluding the proof.

∎

An ECCA KEM/DEM Construction from tb-ATDFs. The above construction can also be used to yield an ECCA tag-based PKE from any tb-ATDF with essentially an identical proof. Then we can apply the transformation of Section 4.3 based on [7], to turn this into a standard ECCA PKE scheme (see proposition 4.13).

C From PKENO-Compatible ECCA tag-based PKE to PKENO-Compatible ECCA PKE

We show that starting with a PKENO-compatible ECCA tag-based PKE, the construction of Section 4.3 based on the BCHK transformation [7] yields a PKENO-compatible ECCA encryption scheme. We focus here on the "more-efficient" version of the BCHK transform that uses symmetric primitives; the simpler version with one-time signatures also works, however.

Let $\mathsf{TB\text{-}PKE} = (\mathsf{Kg}_{tag}, \mathsf{Enc}_{tag}, \mathsf{Dec}_{tag}, \mathsf{pRec}_{tag}, \mathsf{Inval}_{tag})$ be a partially randomness-recovering ciphertext-verifiable tag-based public-key encryption scheme, H, g be hash functions as in Section 4.3, and $\mathsf{MAC} = (\mathsf{tag}, \mathsf{ver})$ be a message-authentication code. We define a partial-randomness-recovering scheme $\mathsf{PKE}_{\text{pce}} = (\mathsf{Kg}_{\text{pce}}, \mathsf{Enc}_{\text{pce}}, \mathsf{Dec}_{\text{pce}}, \mathsf{Cons}_{\text{pce}}, \mathsf{Inval}_{\text{pce}})$ as follows:

- $\mathsf{Kg}_{\text{pce}}(1^k)$: $(pk, sk) \leftarrow\!\!\$\, \mathsf{Kg}_{tag}(1^k)$; output (pk, sk).
- $\mathsf{Enc}_{\text{pce}}(pk, m; r\|x)$: $c_1 \leftarrow H(x)$; $c_2 \leftarrow\!\!\$\, \mathsf{Enc}_{tag}(pk, c_1, m\|x; r)$; $c_3 \leftarrow \mathsf{tag}(g(x), c_2)$; output (c_1, c_2, c_3).
- $\mathsf{Dec}_{\text{pce}}(sk, (c_1, c_2, c_3))$: $m\|x \leftarrow \mathsf{Dec}_{tag}(sk, c_1, c_2)$; if $H(x) \neq c_1$ or $\mathsf{ver}(g(x), c_2, c_3) = 0$ then output $\perp$; else output m.
- $\mathsf{pRec}_{\text{pce}}((pk, sk), (c_1, c_2, c_3))$: $m\|x \leftarrow\!\!\$\, \mathsf{Dec}_{tag}(sk, c_2)$; $r \leftarrow\!\!\$\, \mathsf{pRec}_{tag}(sk, c_1, c_2)$; output (x, r).
- $\mathsf{Cons}_{\text{pce}}(pk, (c_1, c_2, c_3), m, (x, r))$: Return 1 iff $H(x) = c_1$, $\mathsf{ver}(g(x), c_2, c_3) = 1$ and $\mathsf{Cons}_{tag}(pk, c_1, c_2, m\|x, r) = 1$.
- $\mathsf{Inval}_{\text{pce}}(pk, (c_1, c_2, c_3), (x, r))$: Return 1 iff $H(x) \neq c_1$ or $\mathsf{ver}(g(x), c_2, c_3)) \neq 1$, or $\mathsf{Inval}_{tag}(pk, c_1, c_2, r) = 1$.

Proposition C.1 *If* $\mathsf{TB\text{-}PKE}$ *is a PKENO-compatible ECCA tag-based PKE scheme then* $\mathsf{PKE}_{\text{pce}}$ *defined above is a PKENO-compatible ECCA encryption scheme.*

Proof: We have already proven the ECCA security of the scheme in Proposition 4.13. In order to prove that $\mathsf{PKE}_{\text{pce}}$ is a PKENO-compatible ECCA encryption scheme we must show that it satisfies partial-randomness recovery and ciphertext verifiability.

We begin with partial-randomness recovery (pRR). Recall that completeness for pRR is defined as follows: For all $(pk, sk) \leftarrow\!\!\$\, \mathsf{Kg}_{\text{pce}}$ and all $c = (c_1, c_2, c_3) \in \{0,1\}^*, m \in MsgSp(1^k) \cup \{\perp\}$, let $(x, r) \leftarrow\!\!\$\, \mathsf{pRec}_{\text{pce}}(sk, (c_1, c_2, c_3))$. Then

$$\mathsf{Dec}_{\text{pce}}(sk, c) = m \wedge m \neq \perp \;\Rightarrow \mathsf{Cons}_{\text{pce}}(pk, c, m, s) = 1 \ .$$

Let $m\|x \leftarrow\$ \mathsf{Dec}_{tag}(sk, c_2)$. Then by definition of $\mathsf{Dec}_{\mathrm{pce}}$ that when $\mathsf{Dec}_{\mathrm{pce}}$ returns $m \neq \bot$ we have that $H(x) = c_1$ and $\mathsf{ver}(g(x), c_2, c_3) = 1$. Additionally, by pRR completeness of $\mathsf{TB\text{-}PKE}$, we have that r is returned by $\mathsf{pRec}_{tag}(sk, c_1, c_2)$, and thus by pRR completeness of $\mathsf{TB\text{-}PKE}$ that $\mathsf{Cons}_{tag}(pk, c_1, c_2, m\|x, r) = 1$. Together we have that $\mathsf{Cons}_{\mathrm{pce}}(pk, (c_1, c_2, c_3), m, (x, r))$ outputs 1.

Next, recall that soundness for partial randomness recovery (pRR) is defined as follows: For all $(pk, sk) \leftarrow\$ \mathsf{Kg}_{\mathrm{pce}}$ and all $c = (c_1, c_2, c_3) \in \{0,1\}^*, m \in MsgSp(1^k), s = (x, r) \in \{0,1\}^*$:

$$\mathsf{Cons}_{\mathrm{pce}}(pk, c, m, s) = 1 \;\Rightarrow\; \mathsf{Dec}_{\mathrm{pce}}(sk, c) = m \enspace .$$

Soundness for partial randomness recovery follows immediately from the same notion for $\mathsf{TB\text{-}PKE}$ by the definitions of $\mathsf{Cons}_{\mathrm{pce}}$ and $\mathsf{Dec}_{\mathrm{pce}}$.

We now move onto showing the ciphertext verifiability property. Recall that completeness for ciphertext verifiability is defined as follows: For all $(pk, sk) \leftarrow\$ \mathsf{Kg}_{\mathrm{pce}}$ and all $c = (c_1, c_2, c_3) \in \{0,1\}^*$, let $(x, r) \leftarrow\$ \mathsf{pRec}_{\mathrm{pce}}(sk, c)$. Then

$$\mathsf{Dec}_{\mathrm{pce}}(sk, c) = \bot \;\Rightarrow\; \mathsf{Inval}_{\mathrm{pce}}(pk, c, (x, r)) = 1 \enspace .$$

If $\mathsf{Dec}_{\mathrm{pce}}(sk, c) = \bot$ then we must have that $H(x) \neq c_1$ or $\mathsf{Ver}_{tag}(pk, c_1, c_2) \neq 1$, or $\mathsf{Dec}_{tag}(sk, c_1, c_2) = \bot$. Completeness for ciphertext verifiability of $\mathsf{TB\text{-}PKE}$ implies that in the last case $\mathsf{Inval}_{tag}(pk, c_1, c_2, r) = 1$. Together this implies that $\mathsf{Inval}_{\mathrm{pce}}$ returns 1.

Recall that soundness for ciphertext verifiability is defined as follows: For all $(pk, sk) \leftarrow\$ \mathsf{Kg}_{\mathrm{pce}}$ and all $c = (c_1, c_2, c_3) \in \{0,1\}^*, s = (x, r) \in \{0,1\}^*$:

$$\mathsf{Inval}_{\mathrm{pce}}(pk, (c_1, c_2, c_3), (x, r)) = 1 \;\Rightarrow\; \mathsf{Dec}_{\mathrm{pce}}(sk, (c_1, c_2, c_3)) = \bot \enspace .$$

Note that if $\mathsf{Inval}_{\mathrm{pce}}(pk, c, s) = 1$ we have that $H(x) \neq c_1$ or $\mathsf{ver}(g(x), c_2, c_3)) \neq 1$ or $\mathsf{Inval}_{tag}(pk, c_1, c_2, r) = 1$. In the last case, by soundness of $\mathsf{TB\text{-}PKE}$, we have $\mathsf{Dec}_{tag}(sk, c_1, c_2) = \bot$. Together this implies that $\mathsf{Dec}_{\mathrm{pce}}(sk, (c_1, c_2, c_3)) = \bot$. ▌

D Achieving Strong Proof Soundness

The following two notions, given in [18], strengthen proof soundness. Consider the following two games:

```
Experiment $\mathbf{Exp}^{\text{s-proof-snd}}_{\mathsf{PKENO},A}(k)$
    $(pk, m, St) \leftarrow\$ A_1(1^k)$
    $c \leftarrow\$ \mathsf{Enc}(pk, m)$
    $(m', \pi') \leftarrow\$ A_2(pk, c, St)$
    If $m \in MsgSp$, $\mathsf{Ver}(pk, c, m', \pi') = 1$ and $m \neq m'$
        then return 1 ; else return 0
```

```
Experiment $\mathbf{Exp}^{\text{s-comm}}_{\mathsf{PKENO},A}(k)$
    $(pk, c, m, \pi, m', \pi') \leftarrow\$ A(1^k)$
    If $\mathsf{Ver}(pk, c, m, \pi) = 1$ and
        $\mathsf{Ver}(pk, c, m', \pi') = 1$ and
        $m \neq m'$ then return 1
    Else return 0
```

We say that PKENO is *strongly proof-sound* if for every efficient $A = (A_1, A_2)$ the probability of $\mathbf{Exp}^{\text{s-proof-snd}}_{\mathsf{PKENO},A}$ outputting 1 is negligible. Moreover, PKENO is *strongly committing* if for every efficient A the probability of $\mathbf{Exp}^{\text{s-comm}}_{\mathsf{PKENO},A}$ outputting 1 is negligible.

These notions of proof soundness strengthen the original ones in that they let the adversary choose the public key. We next discuss whether our constructions can achieve them/

Generic construction using NIZK. If we are to show that our generic construction satisfies these notions, we cannot base it on correctness of decryption, as for an adversarially chosen key there might not even exist a decryption key. We define thus the following notion, which formalizes the encryption scheme being a binding commitment scheme when the public key is chosen by the adversary:

BACK defintion. A public-key encryption scheme is *binding under adversarially chosen keys* (BACK) if no efficient adversary can output a tuple (pk, m, r, m', r') such that pk is in the public-key space, $m \neq m'$ and $\mathsf{Enc}(pk, m; r) = \mathsf{Enc}(pk, m'; r')$.

We now show that if PKE is a randomness-recovering ECCA-secure scheme which satisfies BACK and public verifiability (i.e., Inval has only inputs pk and c) then the construction $\mathsf{PKENO}[\mathsf{PKE}]$ from Section 6.1, where for RR schemes we define $\mathsf{Cons}(pk, c, m, s) :\equiv (\mathsf{Enc}(pk, m; s) = c)$, satisfies the two strengthenings of proof soundness from [18].

Proposition D.1 *Suppose* PKE *is randomness-recovering, has public verifiability, and satisfies ECCA-security and BACK. Then* $\mathsf{PKENO}[\mathsf{PKE}]$*, defined in Section 6.1 is a PKE scheme with non-interactive opening which is CCPA-secure and satisfies strong proof soundness and the strong committing property.*

Proof: *Strong proof soundness.* Let A be a successful adversary: she outputs (pk, m), the challenger chooses randomness r, gives $c = \mathsf{Enc}(pk, m; r)$ to A, who outputs (m', π'). The adversary wins if $m \neq m'$ and $\mathsf{Ver}(pk, c, m', \pi') = 1$. We distinguish two cases: (1) If $m' \neq \bot$, thus since Ver outputs 1, we have $\mathsf{Cons}(pk, c, m', \pi') = 1$, that is $\mathsf{Enc}(pk, m'; \pi') = c$. The tuple (pk, m, r, m', π') thus breaks BACK. (2) If $m' = \bot$, this means $\mathsf{Inval}(pk, c) = 1$. But $c = \mathsf{Enc}(pk, m; r)$, so this contradicts completeness of Inval.

Committing property. Assume an efficient adversary that outputs $(pk, c, m, \pi, m', \pi')$. We distinguish two cases: (1) $m \neq \bot$ and $m' \neq \bot$; (2) (w.l.o.g.) $m \neq \bot$ and $m' = \bot$; note that since m and m' must be different, they cannot be both $\bot$.

(1) Since both triples (c, m, π) and (c, m', π') satisfy Ver and by the definition of Cons we have:

$$\mathsf{Enc}(pk, m, \pi) = c = \mathsf{Enc}(pk, m', \pi') \ .$$

Such an adversary would thus immediately break BACK.

(2) Again we have $\mathsf{Enc}(pk, m, \pi) = c$ and by $\mathsf{Ver}(pk, c, m', \pi') = 1$ we have $\mathsf{Inval}(pk, c) = 1$. Together this contradicts soundness of Inval.

▌

Efficient scheme based on II-RSA. For our efficient PKENO scheme based on II-RSA in Section 6.4, we can achieve strong proof soundness by having the verifier check that the RSA function defined by a ciphertext is a permutation. Essentially, this yields a tag-based ECCA-secure encryption scheme satisfying BACK, which can be turned into an ECCA-secure encryption scheme satisfying BACK with public verifiability by using one-time signatures. That is, $\mathsf{Ver}(pk, c', m', \pi')$ first parses pk and $N = pq$ and c' as (t, y, vk, σ) where t is the tag, $y \in \mathbb{Z}_N^*$ and vk, σ are for the one-time signature. Then it checks that $(\cdot)^{H(t)} \bmod N$ defines a permutation where H is the hash-to-primes algorithm described in [28]. This can be done efficiency using a recent result of Kakvi, May, and Kiltz [27] (previous methods required $e = H(t) > n$, whereas theirs requires only $e \geq N^{1/4+\varepsilon}$ so can be incorporated without increasing the size of e versus the version of the scheme without strong proof soundness (it is 512 bits in either case).