Publicly Verifiable Ciphertexts*

Juan Manuel González Nieto[1], Mark Manulis[2], Bertram Poettering[3], Jothi Rangasamy[1], and Douglas Stebila[1]

[1] Queensland University of Technology, Brisbane, Australia
{j.gonzaleznieto | j.rangasamy | stebila}@qut.edu.au
[2] University of Surrey, Guildford, United Kingdom
mark@manulis.eu
[3] Royal Holloway, University of London, United Kingdom
bertram.poettering@rhul.ac.uk

Abstract. In many applications, where encrypted traffic flows from an open (public) domain to a protected (private) domain, there exists a gateway that bridges the two domains and faithfully forwards the incoming traffic to the receiver. We observe that indistinguishability against (adaptive) chosen-ciphertext attacks (IND-CCA), which is a mandatory goal in face of active attacks in a public domain, can be essentially relaxed to indistinguishability against chosen-plaintext attacks (IND-CPA) for ciphertexts once they pass the gateway that acts as an IND-CCA/CPA filter by first checking the validity of an incoming IND-CCA ciphertext, then transforming it (if valid) into an IND-CPA ciphertext, and forwarding the latter to the recipient in the private domain. "Non-trivial filtering" can result in reduced decryption costs on the receivers' side.
We identify a class of encryption schemes with *publicly verifiable ciphertexts* that admit generic constructions of (non-trivial) IND-CCA/CPA filters. These schemes are characterized by existence of public algorithms that can distinguish between valid and invalid ciphertexts. To this end, we formally define (non-trivial) public verifiability of ciphertexts for general encryption schemes, key encapsulation mechanisms, and hybrid encryption schemes, encompassing public-key, identity-based, and tag-based encryption flavours. We further analyze the security impact of public verifiability and discuss generic transformations and concrete constructions that enjoy this property.

1 Introduction

Transmission of sensitive information over public networks necessitates the use of cryptographic protection. Modern cryptography offers various techniques, including public key encryption (PKE) and identity-based encryption (IBE), by which the sender can use public information to encrypt a message only the intended receiver can decrypt. These two encryption flavours can be combined into a common syntax, called *general encryption* (GE) [1], and for longer messages,

* This is full version of the paper that appears in Proceedings of the 8th International Conference on Security and Cryptography for Networks (SCN 2012).

hybrid encryption schemes based on key and data encapsulation techniques, i.e. the KEM/DEM approach [10], are often more efficient.

The most standard security notion for encryption schemes is *indistinguishability* (IND) — a ciphertext may not leak any information about the encrypted message (except possibly its length) — whose definitions consider different types of attacks. The strongest is an *adaptive chosen-ciphertext attack* (CCA), in which an attacker can ask for decryption of ciphertexts of her own choice (other than the target ciphertext). IND-CCA-security hence protects encrypted messages of honest senders despite the threat that receivers may also have to decrypt ciphertexts constructed by the adversary. More generally, such threat exists if the network is susceptible to active attacks. In contrast, if senders are trustworthy and their messages are delivered over a network that protects authenticity, then security against *chosen-plaintext attacks* (CPA) would already provide sufficient confidentiality guarantees, possibly resulting in better performance.

IND-CCA/CPA Filtering and Its Applications. Consider an intermediate party, called a *gateway*, and assume that encrypted sender's messages are transmitted over a public network until they reach the gateway and are then forwarded by the gateway over a private network to the receiver, with the gateway being trusted by the receiver to forward faithfully.

By the above reasoning, IND-CCA security would be required for the encrypted traffic from (possibly malicious) senders towards the gateway. But for messages on the internal network — including from the gateway to the receiver — IND-CPA security would be sufficient in practice to preserve confidentiality. If the gateway just forwards all (IND-CCA) ciphertexts from the outside world without modification, all security goals remain satisfied, but perhaps we can improve efficiency for the receiver by having the gateway do some processing on ciphertexts before forwarding them.

An often observed difference between IND-CPA and IND-CCA schemes is that IND-CPA schemes successfully decrypt every given ciphertext, whereas the majority of IND-CCA schemes typically check ciphertexts for consistency and decrypt only those that are "well-formed" [8,10,16,17,18]. For such schemes the gateway could act as a filter that would sort out inconsistent IND-CCA ciphertexts. There exist few IND-CCA schemes [4,24,25], that decrypt every ciphertext to a possibly meaningless (random) message. Such schemes are not well-suited to filtering since the gateway would need to know the receiver's private key to decide whether the message is meaningful, which would in general be unacceptable since it requires trusting gateways for confidentiality, not just integrity.

In this paper, we are interested in solutions that allow an honest-but-curious gateway to transform IND-CCA-secure traffic from a public network into IND-CPA-secure traffic for a private network at low cost and while fully preserving confidentiality of encrypted messages; the key step is that the gateway is trusted to correctly perform a "validity check" of traffic from the public network before forwarding it on to the private network. Recipient devices on the private network can then use a more efficient decryption procedure.

Many real applications could benefit from this "sender-gateway-receiver" scenario: for example, sensor networks often consist of many low-powered nodes that communicate with each other locally and which use a single more powerful gateway device to communicate with the Internet. To protect their local communications, nodes may have shared keys with the sink which they use in highly efficient symmetric key algorithms, only needing to resort to more expensive asymmetric algorithms when communicating with the outside world. In our paradigm, the gateway could take IND-CCA-secure traffic from senders in the outside world, check it for validity, and convert it to a simpler (IND-CPA-secure) format to reduce the processing costs for receiving sensor nodes. As a second example, mail servers (MTAs) generally receive emails over unprotected networks, whereas email recipients typically contact these servers to access their emails after having established an end-to-end authenticated (and possibly encrypted) channel with them. Hence, for encrypted emails or attachments, the mail server could perform a "sanity check" and filter out inconsistent ciphertexts, saving the client from their local processing.

Ciphertext Consistency Checks. IND-CCA-secure schemes where inconsistent ciphertexts can be filtered out based on consistency checks seem very suitable for our purposes. Consistency checks can be either private or public: the check is private if it requires at least partial knowledge of the private key (e.g. in [10]), while public checks do not require any secrets (e.g. in [8,16]).

We will focus on IND-CCA-secure cryptosystems with *publicly verifiable ciphertexts*. Interestingly, public verifiability has been treated so-far in a rather folklore manner, e.g. as a property of concrete schemes, e.g. [8,16,18,17]. To make use of this property in general, for example to enable "black-box" constructions of higher-level security protocols from publicly verifiable encryption schemes, a more formal and thorough characterization of public verifiability is merited. We also note that public verifiability has been extensively addressed in a different context, namely with regard to *threshold encryption*, where as observed initially in [19] and then provably realized in [31,8,7,21], this property is useful to make the threshold decryption process of an IND-CCA-secure threshold encryption scheme non-interactive and robust.

In our applications, public verifiability can immediately be used to detect and filter out invalid IND-CCA ciphertexts, i.e. by trusting the gateway to perform the check. This filtering could also be performed for IND-CCA schemes with private consistency checks, as long as these checks need only parts of the private key that are by themselves not sufficient to break IND-CPA security. Existence of such IND-CCA schemes has been demonstrated by Persiano [26] through his concept of *trapdoor cryptosystems*. For instance, he proved that a trapdoor containing private-key components in Cramer-Shoup PKE [10] that are used in the consistency check cannot be used for an IND-CPA attack (although their disclosure allows malleability attacks). Being concerned about IND-CCA-security, Persiano argued that existence of such trapdoors is a drawback. Taking a look at trapdoor cryptosystems in [26] from the perspective of our work, we observe that the gateway could indeed be given trapdoor information to check IND-CCA

consistency *without* losing IND-CPA security. However, this approach would offer somewhat weaker guarantees in contrast to publicly verifiable schemes: if the delegated trapdoor keys are ever leaked, then IND-CCA security can never be recovered. This contrasts with our approach, in which the receiver always has the potential to obtain IND-CCA security at any particular time simply by performing more operations.

Contributions. We formalize the property of *publicly verifiable ciphertexts* for general encryption, general KEMs, and general KEM/DEM hybrid schemes. Our definitions emphasize the role of public ciphertext consistency checks within the decryption procedure. In our approach, decryption algorithms of publicly verifiable schemes follow a strictly modular design where the consistency check can be performed independently of the remaining "lightweight" decryption procedure. Success or failure of the entire decryption procedure is indicated by the consistency check, which can be performed by any third party without access to any secret information. The only exception is the KEM/DEM approach, where we relax these conditions to account for decryption failures in the DEM part. Our definitions employ the syntax of generalized encryption by Abdalla *et al.* [1] which we extend to also capture *tag-based encryption* (TBE) [3,16] and to address KEMs and the KEM/DEM framework.

With these definitions, we first prove the very general statement that any IND-CCA-secure scheme with publicly verifiable ciphertexts remains at least IND-CPA-secure if the underlying consistency check is outsourced from the decryption procedure. In some sense, this gives us the trivial and well-known result that any IND-CCA-secure ciphertext can be publicly converted into a ciphertext that still guarantees basic IND-CPA protection (since every IND-CCA-secure scheme is also IND-CPA-secure). However, the notion of public verifiability is particular interesting in the case where the verification algorithm is *strictly non-trivial* — the public consistency check fails exactly when the IND-CCA-secure scheme's decryption algorithm fails — as such publicly verifiable schemes can readily be used to build the aforementioned IND-CCA/CPA filters.

We provide several constructions (general and concrete) of IND-CCA-secure schemes with strictly non-trivial publicly verifiability. In addition to existing schemes, e.g. [8,16,18,17], for which public verifiability was discussed informally, we first show that two well-known general ways for obtaining IND-CCA secure schemes offer public verifiability (although not strictly non-trivial public verifiability), namely the Canetti-Halevi-Katz (CHK) transform [9] and the NIZK-based transform [28,22]. The result on CHK contrasts with the related transform by Boneh and Katz [6] that uses a message authentication code (MAC) and does not offer public verifiability. We present a concrete PKE scheme, obtained through a tweak on the KEM of Kiltz [17], that offers an especially lightweight decryption procedure for ciphertexts that passed its strictly non-trivial public verification. In addition to PKE we consider KEMs and give examples of public key-based, identity-based, and tag-based KEMs with strictly non-trivial public verification. Finally, we look into the KEM/DEM paradigm and show that strictly non-trivial public verification of the KEM partially carries over to the

hybrid scheme — namely, we define a *somewhat non-trivial* public verification for hybrid encryption schemes by linking a failure in the hybrid decryption process to a verification failure in either the KEM or the DEM, and show that by outsourcing KEM consistency check the hybrid construction remains at least IND-CPA-secure.

2 Publicly Verifiable Ciphertexts in General Encryption

2.1 Definition: General Encryption

A *general encryption* (GE) scheme $\mathsf{GE} = (\mathsf{PG}, \mathsf{KG}, \mathsf{Enc}, \mathsf{Dec})$ consists of four algorithms:

$\mathsf{PG}(1^k)$**:** The parameter generation algorithm PG takes input a security parameter 1^k, $k \geq 0$, and returns public parameters par and a master secret key msk. Public parameters include a description of the identity space IDSp, the message space MsgSp, and the tag space TagSp.

$\mathsf{KG}(\mathsf{par}, \mathsf{msk}, \mathsf{id})$**:** On input par, msk, and $\mathsf{id} \in \mathsf{IDSp}$, the key generation algorithm KG produces an encryption key ek and decryption key dk.

$\mathsf{Enc}(\mathsf{par}, \mathsf{ek}, M, t)$**:** On input par, ek, a message $M \in \mathsf{MsgSp}$, and a tag $t \in \mathsf{TagSp}$, the encryption algorithm Enc outputs a ciphertext C.

$\mathsf{Dec}(\mathsf{par}, \mathsf{ek}, \mathsf{dk}, C, t)$**:** On input par, ek, dk, C, and a tag t, the deterministic decryption algorithm Dec returns either a plaintext message M or $\perp$ to indicate that it rejects.

This GE formalism encompasses public-key, identity-based, and tag-based encryption schemes:

PKE: Set $\mathsf{msk} = \epsilon$ and assume that IDSp and TagSp contain a single fixed element that can be omitted as implicit input to the algorithms.

IBE: Consider KG that on input id outputs $\mathsf{ek} = \mathsf{id}$ and assume that TagSp contains again a single fixed element that can be omitted as implicit input to the algorithms.

TBE: Set $\mathsf{msk} = \epsilon$ and assume that IDSp contains again a single fixed element that can be omitted as implicit input to the algorithms.

Correctness. A general encryption scheme $\mathsf{GE} = (\mathsf{PG}, \mathsf{KG}, \mathsf{Enc}, \mathsf{Dec})$ is *correct* if, for all $(\mathsf{par}, \mathsf{msk}) \in [\mathsf{PG}]$, all plaintexts $M \in \mathsf{MsgSp}$, all identities $\mathsf{id} \in \mathsf{IDSp}$, all $(\mathsf{ek}, \mathsf{dk}) \in [\mathsf{KG}(\mathsf{par}, \mathsf{msk}, \mathsf{id})]$, and all tags $t \in \mathsf{TagSp}$, we have $\mathsf{Dec}(\mathsf{par}, \mathsf{ek}, \mathsf{dk}, \mathsf{Enc}(\mathsf{par}, \mathsf{ek}, M, t), t) = M$ with probability one, where the probability is taken over the coins of Enc.

Indistinguishability. The IND-CCA/CPA security games between a challenger and an adversary $\mathcal{A}$ are defined by the experiments in Figure 1 (left column). The advantage of $\mathcal{A}$ in those games is defined as

$$\mathbf{Adv}^{\mathsf{IND\text{-}xxx}}_{\mathcal{A},\mathsf{GE}}(k) = \left| \Pr\left(\mathbf{Exp}^{\mathsf{IND\text{-}xxx},0}_{\mathcal{A},\mathsf{GE}}(k) = 1\right) - \Pr\left(\mathbf{Exp}^{\mathsf{IND\text{-}xxx},1}_{\mathcal{A},\mathsf{GE}}(k) = 1\right) \right|,$$

where $\mathsf{xxx} \in \{\mathsf{CPA}, \mathsf{CCA}\}$. A GE scheme is IND-xxx-secure if the advantage of any PPT adversary $\mathcal{A}$ in the corresponding game is negligible in the security parameter k.

$\mathbf{Exp}^{\mathsf{IND\text{-}xxx},b}_{\mathcal{A},\mathsf{GE}}(k)$:

1. $(\mathsf{par}, \mathsf{msk}) \xleftarrow{\$} \mathsf{PG}(1^k)$
2. $U, V, KList, DList \leftarrow \emptyset$
3. $(st, M_0, M_1, \mathsf{id}^*, t^*) \xleftarrow{\$} \mathcal{A}_1^{\mathcal{O}_{\mathsf{EK}}, \mathcal{O}_{\mathsf{DK}}[, \mathcal{O}_{\mathsf{Dec}}]}(\mathsf{par})$
   If $\mathcal{A}$ queries $\mathcal{O}_{\mathsf{EK}}(\mathsf{id})$:
   (a) If $\mathsf{id} \in U$ then return $\perp$
   (b) $U \leftarrow U \cup \{\mathsf{id}\}$
   (c) $(\mathsf{ek}[\mathsf{id}], \mathsf{dk}[\mathsf{id}]) \xleftarrow{\$} \mathsf{KG}(\mathsf{par}, \mathsf{msk}, \mathsf{id})$
   (d) Append $(\mathsf{id}, \mathsf{ek}[\mathsf{id}], \mathsf{dk}[\mathsf{id}])$ to $KList$
   (e) Answer $\mathcal{A}$ with $\mathsf{ek}[\mathsf{id}]$
   If $\mathcal{A}$ queries $\mathcal{O}_{\mathsf{DK}}(\mathsf{id})$:
   (a) If $\mathsf{id} \notin U$ then return $\perp$
   (b) $V \leftarrow V \cup \{\mathsf{id}\}$
   (c) Answer $\mathcal{A}$ with $\mathsf{dk}[\mathsf{id}]$ from $KList$
   If $\mathcal{A}$ queries $\mathcal{O}_{\mathsf{Dec}}(C, \mathsf{id}, t)$ (if $\mathsf{xxx} = \mathsf{CCA}$):
   (a) If $\mathsf{id} \notin U$ then return $\perp$
   (b) $M \leftarrow \mathsf{Dec}(\mathsf{par}, \mathsf{ek}[\mathsf{id}], \mathsf{dk}[\mathsf{id}], C, t)$
   (c) Append (C, id, t) to $DList$
   (d) Answer $\mathcal{A}$ with M
4. If $M_0 = M_1$ or $|M_0| \neq |M_1|$ then return $\perp$
5. If $\mathsf{id}^* \notin U$ then return $\perp$
6. $C^* \xleftarrow{\$} \mathsf{Enc}(\mathsf{par}, \mathsf{ek}[\mathsf{id}^*], M_b, t^*)$
7. $b' \xleftarrow{\$} \mathcal{A}_2^{\mathcal{O}_{\mathsf{EK}}, \mathcal{O}_{\mathsf{DK}}[, \mathcal{O}_{\mathsf{Dec}}]}(st, C^*)$
   Answer queries as above
8. If $\mathsf{id}^* \in V$ then return $\perp$
9. If $(C^*, \mathsf{id}^*, t^*) \in DList$ then return $\perp$
10. Return 1 if $b' = b$, else return 0.

$\mathbf{Exp}^{\mathsf{IND\text{-}xxx},b}_{\mathcal{A},\mathsf{GKEM}}(k)$:

1. $(\mathsf{par}, \mathsf{msk}) \xleftarrow{\$} \mathsf{PG}(1^k)$
2. $U, V, KList, DList \leftarrow \emptyset$
3. $(st, \mathsf{id}^*, t^*) \xleftarrow{\$} \mathcal{A}_1^{\mathcal{O}_{\mathsf{EK}}, \mathcal{O}_{\mathsf{DK}}[, \mathcal{O}_{\mathsf{Dec}}]}(\mathsf{par})$
   If $\mathcal{A}$ queries $\mathcal{O}_{\mathsf{EK}}(\mathsf{id})$:
   (a) If $\mathsf{id} \in U$ then return $\perp$
   (b) $U \leftarrow U \cup \{\mathsf{id}\}$
   (c) $(\mathsf{ek}[\mathsf{id}], \mathsf{dk}[\mathsf{id}]) \xleftarrow{\$} \mathsf{KG}(\mathsf{par}, \mathsf{msk}, \mathsf{id})$
   (d) Append $(\mathsf{id}, \mathsf{ek}[\mathsf{id}], \mathsf{dk}[\mathsf{id}])$ to $KList$
   (e) Answer $\mathcal{A}$ with $\mathsf{ek}[\mathsf{id}]$
   If $\mathcal{A}$ queries $\mathcal{O}_{\mathsf{DK}}(\mathsf{id})$:
   (a) If $\mathsf{id} \notin U$ then return $\perp$
   (b) $V \leftarrow V \cup \{\mathsf{id}\}$
   (c) Answer $\mathcal{A}$ with $\mathsf{dk}[\mathsf{id}]$ from $KList$
   If $\mathcal{A}$ queries $\mathcal{O}_{\mathsf{Dec}}(C, \mathsf{id}, t)$ (if $\mathsf{xxx} = \mathsf{CCA}$):
   (a) If $\mathsf{id} \notin U$ then return $\perp$
   (b) $K \leftarrow \mathsf{Decap}(\mathsf{par}, \mathsf{ek}[\mathsf{id}], \mathsf{dk}[\mathsf{id}], C, t)$
   (c) Append (C, id, t) to $DList$
   (d) Answer $\mathcal{A}$ with K
4. If $\mathsf{id}^* \notin U$ then return $\perp$
5. $(C^*, K_0^*) \xleftarrow{\$} \mathsf{Encap}(\mathsf{par}, \mathsf{ek}[\mathsf{id}^*], t^*)$
6. $K_1^* \xleftarrow{\$} \mathsf{KeySp}(k)$
7. $b' \xleftarrow{\$} \mathcal{A}_2^{\mathcal{O}_{\mathsf{EK}}, \mathcal{O}_{\mathsf{DK}}[, \mathcal{O}_{\mathsf{Dec}}]}(st, C^*, K_b^*)$
   Answer queries as above.
8. If $\mathsf{id}^* \in V$ then return $\perp$
9. If $(C^*, \mathsf{id}^*, t^*) \in DList$ then return $\perp$
10. Return 1 if $b' = b$, else return 0.

Fig. 1. IND-CCA/CPA security experiments for General Encryption (left) and General Key Encapsulation (right)

2.2 General Encryption with Publicly Verifiable Ciphertexts

In our definition of general encryption with publicly verifiable ciphertexts we require existence of a separate algorithm for ciphertext validation and that the scheme's original decryption procedure can be logically divided into this public validation check followed by a lightweight decryption algorithm.

Definition 1 (Publicly Verifiable GE). *A general encryption scheme* $\mathsf{GE} = (\mathsf{PG}, \mathsf{KG}, \mathsf{Enc}, \mathsf{Dec})$ *is said to be* publicly verifiable *with respect to auxiliary algorithms* Ver *and* Dec' *if* $\mathsf{Dec}(\mathsf{par}, \mathsf{ek}, \mathsf{dk}, C, t)$ *has the same input/output behavior as the following sequence of operations:*

1. $C' \leftarrow \mathsf{Ver}(\mathsf{par}, \mathsf{ek}, C, t)$
2. *If* $C' = \perp$*, then return* $\perp$
3. $M \leftarrow \mathsf{Dec}'(\mathsf{par}, \mathsf{ek}, \mathsf{dk}, C', t)$
4. *Return* M

where Ver *and* Dec′ *satisfy the following:*

$\mathsf{Ver}(\mathsf{par}, \mathsf{ek}, C, t)$**:** *Given public parameters* par*, the encryption key* ek*, a ciphertext* C*, and a tag* t*, this algorithm outputs either* $\perp$ *if the ciphertext fails the validation or a (transformed) ciphertext* C'*. Note that* Ver *does not take any secrets as input.*

$\mathsf{Dec}'(\mathsf{par}, \mathsf{ek}, \mathsf{dk}, C', t)$**:** *This deterministic algorithm takes input public parameters* par*, encryption and decryption keys* $(\mathsf{ek}, \mathsf{dk})$*, a ciphertext* C'*, and a tag* t*, and outputs a message* M *or* $\perp$*.*

Hereafter, when we say $\mathsf{Dec} = \mathsf{Dec}' \circ \mathsf{Ver}$ we mean that Dec can be decomposed into two algorithms Ver and Dec′ according to the above construction. Note that all IND-CCA-secure general encryption schemes trivially achieve public verifiability with respect to $\mathsf{Ver}(\mathsf{par}, \mathsf{ek}, C, t) := C$ and $\mathsf{Dec}' := \mathsf{Dec}$. We are often interested in the case where something non-trivial is occurring in Ver, i.e. where the consistency check is essential for successful decryption. Note that this separation does not formally ensure that Dec′ is more efficient than Dec, though in practice we are of course interested primarily in such schemes.

Definition 2 (Strictly Non-Trivial Public Verification). *Let* $\mathsf{GE} = (\mathsf{PG}, \mathsf{KG}, \mathsf{Enc}, \mathsf{Dec})$ *be a general encryption scheme that is publicly verifiable with respect to auxiliary algorithms* Ver *and* Dec′*. Let* $(\mathsf{par}, \mathsf{msk}) \leftarrow \mathsf{PG}(1^k)$*.* Ver *is said to be* strictly non-trivial *if, for all* $\mathsf{id} \in \mathsf{IDSp}$*, all* $t \in \mathsf{TagSp}$*, and* $(\mathsf{ek}, \mathsf{dk}) \leftarrow \mathsf{KG}(\mathsf{par}, \mathsf{msk}, \mathsf{id})$*,*

1. $\mathsf{Ver}(\mathsf{par}, \mathsf{ek}, C, t) = \perp \Leftrightarrow \mathsf{Dec}(\mathsf{par}, \mathsf{ek}, \mathsf{dk}, C, t) = \perp$ *for all* C*, and*
2. *there exists a ciphertext* C *for which* $\mathsf{Dec}(\mathsf{par}, \mathsf{ek}, \mathsf{dk}, C, t) = \perp$*.*

Condition 1 requires that successful public verification is both necessary and sufficient for the decryption algorithm not to fail. Condition 2 formally excludes IND-CCA-secure schemes where Dec never outputs $\perp$ (e.g. [24,25,4] where modified (challenge) ciphertexts decrypt to random messages) to capture the intuition that in order to determine whether C carries some meaningful message one must have at least partial knowledge of the private key (which contradicts the goals of strictly non-trivial public verification).

Theorem 1 (proven in Appendix A) shows that any IND-CCA-secure GE scheme with publicly verifiable ciphertexts remains at least IND-CPA-secure if its decryption algorithm Dec is replaced with Dec′. In the original decryption procedure a strictly non-trivial verification process may syntactically modify the ciphertext. For syntactical reasons we must ensure that ciphertexts output by the encryption algorithm of the new scheme can be processed with Dec′. This is achieved via post-processing of original ciphertexts using Ver and by viewing this step as part of the new encryption algorithm.

Theorem 1. *Let* $\mathsf{GE} = (\mathsf{PG}, \mathsf{KG}, \mathsf{Enc}, \mathsf{Dec})$ *be an* IND-CCA*-secure general encryption scheme that is publicly verifiable with respect to* Ver *and* Dec'*. Let* $\mathsf{Enc}' := \mathsf{Ver} \circ \mathsf{Enc}$ *(where* $\circ$ *denotes the obvious composition) and let* $\mathsf{GE}' := (\mathsf{PG}, \mathsf{KG}, \mathsf{Enc}', \mathsf{Dec}')$*. For every* IND-CPA *adversary* $\mathcal{A}$ *against* GE' *there exists an* IND-CCA *adversary* $\mathcal{B}$ *against* GE *such that, for all* $k \geq 0$*,* $\mathbf{Adv}^{\mathsf{IND\text{-}CPA}}_{\mathcal{A},\mathsf{GE}'}(k) \leq \mathbf{Adv}^{\mathsf{IND\text{-}CCA}}_{\mathcal{B},\mathsf{GE}}(k)$*, where* $\mathcal{B}$ *has (asymptotically) the same running time as* $\mathcal{A}$*.*

2.3 Publicly Verifiable Ciphertexts through CHK Transformation

Canetti, Halevi, and Katz [9] described an method for constructing an IND-CCA-secure public key encryption scheme PKE from any IND-CPA-secure identity-based encryption scheme IBE with identity-space $\{0,1\}^{\ell_s(k)}$ and any strongly unforgeable *one-time signature* scheme $\mathsf{OTS} = (\mathsf{KG}, \mathsf{Sign}, \mathsf{Vrfy})$ with the verification key space $\{0,1\}^{\ell_s(k)}$ (see [9] for the syntax of OTS and the details of the original CHK transform; note that one-time signature schemes can be constructed from any one-way function [27]). Later, Kiltz [16] showed that CHK transform works also if the IND-CPA-secure IBE scheme is replaced by a weakly IND-CCA-secure tag-based encryption scheme TBE with tag-space $\{0,1\}^{\ell_s(k)}$.

Figure 2 (which uses GE notation) shows that in both cases, the resulting PKE is public verification with respect to PKE.Ver and $\mathsf{PKE.Dec}'$, but importantly the public verification is *not* strictly non-trivial: the IBE or TBE decryption operation may still fail. In the IBE-based case $\mathsf{ek} = \epsilon$ remains empty while $\mathsf{dk} = \mathsf{msk}$ and par are output by IBE.PG. In the TBE-based case ek and dk are output by TBE.KG using par generated by TBE.PG. Original IBE-based transform from [9] and its TBE-based version from [16] are obtained using $\mathsf{PKE.Dec} = \mathsf{PKE.Dec}' \circ \mathsf{PKE.Ver}$.

$\mathsf{PKE.Enc}(\mathsf{par}, \mathsf{ek}, M)$:

1. $(vk, sigk) \xleftarrow{\$} \mathsf{OTS.KG}(1^k)$
2. If IBE-based:
   $c \leftarrow \mathsf{IBE.Enc}(\mathsf{par}, vk, M)$
   If TBE-based:
   $c \leftarrow \mathsf{TBE.Enc}(\mathsf{par}, \mathsf{ek}, M, vk)$
3. $\sigma \leftarrow \mathsf{OTS.Sign}(sigk, c)$
4. Return $C = (c, \sigma, vk)$

$\mathsf{PKE.Ver}(\mathsf{par}, \mathsf{ek}, C)$:

1. $(c, \sigma, vk) \leftarrow C$
2. If $\mathsf{OTS.Vrfy}(c, \sigma, vk) = \perp$ then return $\perp$
3. Return $C' = (c, vk)$

$\mathsf{PKE.Dec}'(\mathsf{par}, \mathsf{ek}, \mathsf{dk}, C')$:

1. $(c, vk) \leftarrow C'$
2. If IBE-based:
   $usk_{vk} \leftarrow \mathsf{IBE.KG}(\mathsf{par}, \mathsf{dk}, vk)$
   $M \leftarrow \mathsf{IBE.Dec}(\mathsf{par}, vk, usk_{vk}, c)$
   If TBE-based:
   $M \leftarrow \mathsf{TBE.Dec}(\mathsf{par}, \mathsf{ek}, \mathsf{dk}, c, vk)$
3. Return M

Fig. 2. PKE with Publicly Verifiable Ciphertexts from CHK Transformation

2.4 Publicly Verifiable Ciphertexts using NIZKs

An IND-CPA-secure public key encryption scheme $\mathsf{PKE}' = (\mathsf{PG}, \mathsf{KG}, \mathsf{Enc}, \mathsf{Dec})$ can be converted into an IND-CCA-secure one using a non-interactive zero-knowledge (NIZK) proof (P, V) with simulation soundness, as proven by Sahai [28] based on the Naor-Yung approach [22]. The private/public key pair of the resulting scheme

PKE is given by $(\mathsf{dk}, \mathsf{ek}) = ((\mathsf{dk}_1, \mathsf{dk}_2), (\mathsf{ek}_1, \mathsf{ek}_2, \rho))$ where $(\mathsf{dk}_i, \mathsf{ek}_i)$, $i \in \{1, 2\}$, are obtained from two independent runs of $\mathsf{PKE}'.\mathsf{KG}$ and ρ is the common reference string of the NIZK proof system for languages of the form $(c_1, c_2, \mathsf{ek}_1, \mathsf{ek}_2)$ satisfying $c_1 = \mathsf{PKE}'.\mathsf{Enc}(\mathsf{par}, \mathsf{ek}_1, M) \wedge c_2 = \mathsf{PKE}'.\mathsf{Enc}(\mathsf{par}, \mathsf{ek}_2, M)$ where M (and implicitly random coins used in the encryption process) play the role of the witness. As demonstrated in Figure 3, IND-CCA schemes output by this transformation offer public verifiability, though not strictly non-trivial public verifiability as the $\mathsf{PKE}'.\mathsf{Dec}$ operation in $\mathsf{PKE}.\mathsf{Dec}'$ may output $\perp$. This reasoning also applies to the NIZK-based constructions from [12] and to the first IND-CCA-secure PKE scheme by Dolev, Dwork, and Naor [11] that uses NIZK-proofs in a slightly different way. Although NIZK-based schemes are regarded as not efficient, we notice that their lightweight decryption procedure Dec' (if the scheme is viewed from the public verifiability perspective) is as efficient as that of CHK-based schemes in Figure 2.

$\mathsf{PKE}.\mathsf{Enc}(\mathsf{par}, \mathsf{ek}, M)$:

1. $(\mathsf{ek}_1, \mathsf{ek}_2, \rho) \leftarrow \mathsf{ek}$
2. $c_1 \leftarrow \mathsf{PKE}'.\mathsf{Enc}(\mathsf{par}, \mathsf{ek}_1, M)$
3. $c_2 \leftarrow \mathsf{PKE}'.\mathsf{Enc}(\mathsf{par}, \mathsf{ek}_2, M)$
4. $\pi \leftarrow P(M, (c_1, c_2, \mathsf{ek}_1, \mathsf{ek}_2), \rho)$
5. Return $C = (c_1, c_2, \pi)$

$\mathsf{PKE}.\mathsf{Ver}(\mathsf{par}, \mathsf{ek}, C)$:

1. $(\mathsf{ek}_1, \mathsf{ek}_2, \rho) \leftarrow \mathsf{ek}$
2. $(c_1, c_2, \pi) \leftarrow C$
3. If $V(\rho, (c_1, c_2, \mathsf{ek}_1, \mathsf{ek}_2), \pi) = \perp$ then return $\perp$
4. Return $C' = c_1$

$\mathsf{PKE}.\mathsf{Dec}'(\mathsf{par}, \mathsf{ek}, \mathsf{dk}, C')$:

1. $(\mathsf{dk}_1, \mathsf{dk}_2) \leftarrow \mathsf{dk}$
2. $c_1 \leftarrow C'$
3. $M \leftarrow \mathsf{PKE}'.\mathsf{Dec}(\mathsf{par}, \mathsf{ek}_1, \mathsf{dk}_1, c_1)$
4. Return M

Fig. 3. PKE with Publicly Verifiable Ciphertexts from NIZK-based Transformation

2.5 Our PKE Scheme with Strictly Non-Trivial Publicly Verifiable Ciphertexts

In this section, we propose a practical IND-CCA-secure PKE scheme, whose public verification is strictly non-trivial and is well-suited for IND-CCA/CPA filters described in the introduction due to an especially light algorithm Dec'. Our construction is inspired by the IND-CCA public-key KEM of Kiltz [17], which when plugged into a KEM/DEM framework would yield an IND-CCA-secure PKE scheme (but loose strictly non-trivial public verification as discussed in Section 4). In contrast, we obtain strictly non-trivial publicly verifiable PKE in a more direct way, by using the encapsulated key in [17] as a one-time pad for the message and by linking the resulting ciphertext components together with a one-time signature, whose verification key is in turn bound to the KEM ciphertext part through a tweak on the original scheme from [17]. Our scheme provides strictly non-trivial public verifiability, unlike the schemes presented in Sections 2.3 and 2.4 based on the CHK and NIZK transformations.

The scheme. Our PG algorithm is similar to [17] except that it uses gap groups: $\mathsf{PG}(1^k)$ outputs public parameters $\mathsf{par} = (\mathbb{G}, p, g, \mathsf{DDH}, \mathsf{H})$ where $\mathbb{G} = \langle g \rangle$ is a multiplicative cyclic group of prime order p, $2^k < p < 2^{k+1}$, DDH is an efficient

algorithm such that $\mathsf{DDH}(g^a, g^b, g^c) = 1 \Leftrightarrow c = ab\,(p)$, and $\mathsf{H} : \mathbb{G} \to \{0,1\}^{\ell_1(k)}$ is a cryptographic hash function such that $\ell_1(k)$ is a polynomial in k. We also use a strong one-time signature scheme $\mathsf{OTS} = (\mathsf{KG}, \mathsf{Sign}, \mathsf{Vrfy})$ with verification key space $\{0,1\}^{\ell_2(k)}$ such that $\ell_2(k)$ is a polynomial in k and a target collision resistant hash function $\mathsf{TCR} : \mathbb{G} \times \{0,1\}^{\ell_2(k)} \to \mathbb{Z}_p$. The message space is $\mathsf{MsgSp} = \{0,1\}^{\ell_1(k)}$. The scheme works as shown in Figure 4.

$\mathsf{PKE.KG}(\mathsf{par})$:

1. $x \xleftarrow{\$} \mathbb{Z}_p^*$
2. $u \leftarrow g^x$, $v \xleftarrow{\$} \mathbb{G}$
3. $\mathsf{ek} \leftarrow (u, v)$, $\mathsf{dk} \leftarrow x$
4. Return $(\mathsf{ek}, \mathsf{dk})$

$\mathsf{PKE.Enc}(\mathsf{par}, \mathsf{ek}, M)$:

1. $(u, v) \leftarrow \mathsf{ek}$
2. $(vk, sigk) \xleftarrow{\$} \mathsf{OTS.KG}(1^k)$
3. $r \xleftarrow{\$} \mathbb{Z}_p^*$, $c_1 \leftarrow g^r$
4. $t \leftarrow \mathsf{TCR}(c_1, vk)$, $\pi \leftarrow (u^t v)^r$
5. $K \leftarrow \mathsf{H}(u^r)$, $c_2 \leftarrow M \oplus K$
6. $c \leftarrow (c_1, c_2, \pi)$
7. $\sigma \leftarrow \mathsf{OTS.Sign}(sigk, c)$
8. Return $C = (c, \sigma, vk)$

$\mathsf{PKE.Ver}(\mathsf{par}, \mathsf{ek}, C)$:

1. $(u, v) \leftarrow \mathsf{ek}$
2. $(c, \sigma, vk) \leftarrow C$
3. $(c_1, c_2, \pi) \leftarrow c$
4. $t \leftarrow \mathsf{TCR}(c_1, vk)$
5. If $\mathsf{DDH}(c_1, u^t v, \pi) \neq 1$ or $\mathsf{OTS.Vrfy}(c, \sigma, vk) = \perp$ return $\perp$
6. Return $C' = (c_1, c_2)$

$\mathsf{PKE.Dec}'(\mathsf{par}, \mathsf{ek}, \mathsf{dk}, C')$:

1. $(c_1, c_2) \leftarrow C'$
2. $x \leftarrow \mathsf{dk}$
3. $K \leftarrow \mathsf{H}(c_1^x)$, $M \leftarrow c_2 \oplus K$
4. Return M

Fig. 4. Our PKE with Strictly Non-Trivial Publicly Verifiable Ciphertexts

Security Analysis. First we give intuition why our scheme is IND-CCA-secure. Let (c^*, σ^*, vk^*) be the challenge ciphertext. As we discussed above, without the CHK transform, the proposed PKE can be seen as a KEM/DEM combination which is at least IND-CPA-secure due to Herranz *et al.* [14]. As for the KEM, the Hashed Diffie-Hellman (HDH) assumption [2] can be used to prove the IND-CPA security of the resulting PKE. Note that the message does not depend on vk^*, and σ^* is just the signature on c^*. Therefore c^* being an output of the IND-CPA-secure scheme hides the value of the chosen b from the adversary.

We now claim that the IND-CCA adversary $\mathcal{A}$ may access decryption oracle but gains no help in guessing the value of b. Suppose the adversary submits a ciphertext $(c', \sigma', vk') \neq (c^*, \sigma^*, vk^*)$ to the decryption oracle. Now there are two cases: $(a)\, vk' = vk^*$ or $(b)\, vk' \neq vk^*$. When $vk' = vk^*$, the decryption oracle will output $\perp$ as the adversary fails to break the underlying strongly unforgeable one-time signature scheme with respect to vk'. When $vk' \neq vk^*$, the attacker $\mathcal{B}$ against the HDH problem can set the public keys as seen in the IND-CCA security proof for the KEM by Kiltz [17] such that (1) $\mathcal{B}$ can answer except for the challenge ciphertext all decryption queries from $\mathcal{A}$ even without the knowledge of the secret key and (2) $\mathcal{B}$ solves HDH if $\mathcal{A}$ wins. This security is captured by the following theorem, which is proven in Appendix B.

Theorem 2. *Assume that* TCR *is a target collision resistant hash function and* OTS *is a strongly unforgeable one-time signature scheme. Under the Hashed Diffie-Hellman assumption for* $\mathbb{G}$ *and* H*, the PKE scheme* (PKE.KG, PKE.Enc, PKE.Dec = PKE.Dec′ ∘ PKE.Ver) *based on Figure 4 is* IND-CCA*-secure.*

Efficiency. Our PKE scheme in Figure 4 is more efficient than previous schemes with public consistency checks. In our scheme, public keys consist of 2 group elements, the ciphertext overhead is 2 group elements, a one-time signature and a one-time verification key, encryption requires 3.5 group exponentiations (using simultaneous exponentiation) and 1 one-time signature, verification requires 1 group exponentiation, 2 pairings, and 1 one-time signature verification, and lightweight decryption requires only one exponentiation.

Amongst existing PKE constructions with public consistency checks, only two seem to offer the same efficiency for lightweight decryption: Kiltz [17] describes a (direct) PKE construction (in addition to KEM) that is publicly verifiable with the same lightweight decryption cost of 1 group exponentiation, but at the cost of requiring public keys with the number of group elements being linear in the security parameter, as opposed to only 2 group elements in the public key of our scheme. Hanaoka and Kurosawa [13] describe a publicly verifiable KEM that, when combined with a DEM, would yield a (somewhat non-trivial, cf. Section 4) publicly verifiable PKE. Its lightweight decryption would require 1 group exponentiation (plus any costs from the DEM) but its public keys would contain 3 group elements, compared to 2 group elements in our scheme.

3 Publicly Verifiable Ciphertexts in General KEMs

3.1 Definition: General KEM

A *general key encapsulation* mechanism (GKEM) is a tuple GKEM = (PG, KG, Encap, Decap) of four algorithms such that PG and KG have the same syntax as in case of general encryption (cf. Section 2.1) except that message space is replaced with the key space KeySp, whereas the syntax of Encap and Decap matches that of Enc and Dec, respectively, with the only difference that Encap outputs a ciphertext C and a session key $K \in$ KeySp, while Decap outputs either K or $\perp$.

GKEM correctness and adversarial advantage $\mathbf{Adv}^{\mathsf{IND\text{-}xxx}}_{\mathcal{A},\mathsf{GKEM}}(k)$, xxx ∈ {CPA, CCA} in indistinguishability experiments from Figure 1 are also defined analogously to the case of general encryption.

3.2 General KEMs with Public Verifiable Ciphertexts

Definition 3 (Publicly Verifiable GKEM). *A general key encapsulation mechanism* GKEM = (PG, KG, Encap, Decap) *is said to be* publicly verifiable *with respect to auxiliary algorithms* Ver *and* Decap′ *if* Decap = Decap′ ∘ Ver *where* Ver *and* Decap′ *satisfy the following:*

$\mathsf{Ver}(\mathsf{par}, \mathsf{ek}, C, t)$**:** *Given public parameters* par*, the encryption key* ek*, a ciphertext* C*, and a tag* t*, this algorithm outputs either* $\perp$ *if the ciphertext fails the validation, or a (transformed) ciphertext* C'*. Note that* Ver *does not take any secrets as input.*

$\mathsf{Decap}'(\mathsf{par}, \mathsf{ek}, \mathsf{dk}, C', t)$**:** *This deterministic algorithm takes input public parameters* par*, encryption and decryption keys* $(\mathsf{ek}, \mathsf{dk})$*, a ciphertext* C'*, and a tag* t*, and outputs a key* K*.*

Since all IND-CCA-secure general GKEMs trivially achieve public verifiability with respect to $\mathsf{Ver}(\mathsf{par}, \mathsf{ek}, C, t) := C$ and $\mathsf{Decap}' := \mathsf{Decap}$ we can reuse Definition 2 for GKEMs to define their strictly non-trivial public verification.

Theorem 3 (whose proof is identical to that of Theorem 1 and is omitted here) shows that any publicly verifiable IND-CCA-secure GKEM scheme will remain at least IND-CPA-secure if the verification algorithm Ver is run by an honest-but-curious gateway. To account for a non-trivial verification process that may modify the ciphertext, we again apply post-processing to the output of the encapsulation algorithm (cf. Section 2.2).

Theorem 3. *Let* $\mathsf{GKEM} = (\mathsf{PG}, \mathsf{KG}, \mathsf{Encap}, \mathsf{Decap})$ *be an* IND-CCA*-secure general KEM and publicly verifiable with respect to* Ver *and* Decap'*. Let* $\mathsf{Encap}' := \mathsf{Ver} \circ \mathsf{Encap}$ *and let* $\mathsf{GKEM}' := (\mathsf{PG}, \mathsf{KG}, \mathsf{Encap}', \mathsf{Decap}')$*. For every* IND-CPA *adversary* $\mathcal{A}$ *against* GKEM'*, there exists an* IND-CCA *adversary* $\mathcal{B}$ *against* GKEM *such that, for all* $k \geq 0$*,* $\mathbf{Adv}^{\mathsf{IND\text{-}CPA}}_{\mathcal{A},\mathsf{GKEM}'}(k) \leq \mathbf{Adv}^{\mathsf{IND\text{-}CCA}}_{\mathcal{B},\mathsf{GKEM}}(k)$*, where* $\mathcal{B}$ *has (asymptotically) the same running time as* $\mathcal{A}$*.*

3.3 Constructions of Strictly Non-Trivial Publicly Verifiable KEMs

We now present some examples for KEMs with publicly verifiable ciphertexts. First, we discuss the publicly verifiable construction of an identity-based KEM that we obtain immediately from the IND-CCA-secure IB-KEM proposed by Kiltz and Galindo [18]. Parameters $\mathsf{par}' = (\mathbb{G}_1, \mathbb{G}_T, p, g, e, \mathsf{H})$ chosen by parameter generation algorithm $\mathsf{PG}(1^k)$, $k \in \mathbb{Z}_{\geq 0}$, are such that $\mathbb{G}_1$ is a multiplicative cyclic group of prime order $p : 2^{2k} < p$, $\mathbb{G}_T$ is a multiplicative cyclic group of the same order, g is a random generator of $\mathbb{G}_1$, $e : \mathbb{G}_1 \times \mathbb{G}_1 \to \mathbb{G}_T$ is a non-degenerate bilinear map, and $\mathsf{H} : \{0,1\}^{\ell(k)} \to \mathbb{G}_1$ is a random hash function such that $\ell(k)$ is a polynomial in k. We also use a target collision resistant function $\mathsf{TCR} : \mathbb{G}_1 \to \mathbb{Z}_p$. Figure 5 details the scheme.

Note that by defining $\mathsf{KEM.Decap} = \mathsf{KEM.Decap}' \circ \mathsf{KEM.Ver}$ we immediately obtain the original Kiltz-Galindo IB-KEM [18]. It is easy to see that its public verification algorithm $\mathsf{KEM.Ver}$ is strictly non-trivial. Further, Kiltz and Galindo noted that ignoring all operations associated to the identity in their IB-KEM yields a simplified version of the IND-CCA-secure public-key schemes from [8,16]. Therefore, by removing computations related to the ciphertext component c_2 and the key generation algorithm KG from Kiltz-Galindo's IB-KEM, we immediately obtain publicly verifiable constructions of a public-key KEM and a tag-based KEM with strictly non-trivial public verification.

$\mathsf{KEM.PG}(1^k)$:

1. Generate $\mathsf{par}' = (\mathbb{G}_1, \mathbb{G}_T, p, g, e, \mathsf{H})$
2. $\alpha \xleftarrow{\$} \mathbb{G}_1$, $\mathsf{msk} \leftarrow \alpha$
3. $u, v \xleftarrow{\$} \mathbb{G}_1$, $z \leftarrow e(g, \alpha)$
4. $\mathsf{pk} \leftarrow (u, v, z)$
5. $\mathsf{par} \leftarrow (\mathsf{par}', \mathsf{pk})$
6. Return $(\mathsf{par}, \mathsf{msk})$

$\mathsf{KEM.Encap}(\mathsf{par}, \mathsf{id})$:

1. $(\mathsf{par}', \mathsf{pk}) \leftarrow \mathsf{par}$
2. Parse par' and pk
3. $r \xleftarrow{\$} \mathbb{Z}_p^*$, $c_1 \leftarrow g^r$
4. $t \leftarrow \mathsf{TCR}(c_1)$
5. $c_2 \leftarrow \mathsf{H}(\mathsf{id})^r$
6. $c_3 \leftarrow (u^t v)^r$
7. $K \leftarrow z^r \in \mathbb{G}_T$
8. $C \leftarrow (c_1, c_2, c_3) \in \mathbb{G}_1^3$
9. Return (C, K)

$\mathsf{KEM.KG}(\mathsf{par}, \mathsf{msk}, \mathsf{id})$:

1. Parse $(\mathsf{par}', \mathsf{pk}) \leftarrow \mathsf{par}$ and par'
2. $s \xleftarrow{\$} \mathbb{Z}_p$, $\mathsf{dk}[\mathsf{id}] \leftarrow (\alpha \cdot \mathsf{H}(\mathsf{id})^s, g^s)$
3. Return $\mathsf{dk}[\mathsf{id}]$

$\mathsf{KEM.Ver}(\mathsf{par}, \mathsf{pk}, \mathsf{id}, C)$:

1. $(\mathsf{par}', \mathsf{pk}) \leftarrow \mathsf{par}$
2. Parse par' and pk
3. $(c_1, c_2, c_3) \leftarrow C$, $t \leftarrow \mathsf{TCR}(c_1)$
4. If $e(g, c_3) \neq e(c_1, u^t v)$ or $e(g, c_2) \neq e(c_1, \mathsf{H}(\mathsf{id}))$, then return $\perp$
5. Return $C' = (c_1, c_2)$

$\mathsf{KEM.Decap}'(\mathsf{par}, \mathsf{id}, \mathsf{dk}[\mathsf{id}], C')$:

1. $(\mathsf{par}', \mathsf{pk}) \leftarrow \mathsf{par}$
2. Parse par'
3. $(c_1, c_2) \leftarrow C'$, $(d_1, d_2) \leftarrow \mathsf{dk}[\mathsf{id}]$
4. $K \leftarrow e(c_1, d_1)/e(c_2, d_2)$
5. Return K

Fig. 5. Kiltz-Galindo IB-KEM with Publicly Verifiable Ciphertexts

4 Publicly Verifiable Ciphertexts in Hybrid Encryption

Since its invention, the KEM/DEM approach [10,29], being very simple and flexible, has become popular and part of several encryption standards [15,23,30]. It has been shown that if both the KEM and the DEM are secure against chosen-ciphertext attacks, then so is the resulting hybrid encryption scheme [10]. Herranz *et al.* [14] studied necessary and sufficient security conditions for KEMs and DEMs in relation with the security of the hybrid construction. They showed that for the IND-CCA-security of the hybrid scheme, the KEM must be IND-CCA-secure while the security requirement on the DEM can be relaxed from IND-CCA to IND-OTCCA that prevents *one-time* (adaptive) chosen-ciphertext attacks.

Therefore, when dealing with public verifiability of hybrid schemes we must take into account existence of consistency checks in the decryption of DEM (in addition to checks for the KEM part). Since DEM consistency checks are performed using the decapsulated key, hybrid schemes cannot provide strictly non-trivial public verification from Definition 2. We show, however, that these schemes can offer a somewhat relaxed property, where public verifiability refers only to the KEM part, meaning that successful public consistency check of the KEM part is a necessary but not a sufficient condition for the overall success of decryption. In the context of gateway-assisted IND-CCA/CPA conversion this property effectively allows to outsource the consistency check of the KEM part to the gateway. In this way clients would only need to perform private consistency checks for the DEM part, which means negligible costs in comparison to the verification costs for KEMs.

4.1 Definition: Hybrid General Encryption

Let $\mathsf{GKEM} = (\mathsf{PG}, \mathsf{KG}, \mathsf{Encap}, \mathsf{Decap})$ be a general KEM scheme (as defined in Section 3.1) and let $\mathsf{DEM} = (\mathsf{Enc}, \mathsf{Dec})$ be a one-time symmetric key encryption scheme [14]. The two schemes are assumed to be compatible, i.e. session keys output by KEM are appropriate for DEM.

A *hybrid general encryption* (HGE) scheme is a tuple $\mathsf{HGE} = (\mathsf{PG}, \mathsf{KG}, \mathsf{Enc}, \mathsf{Dec})$ of four algorithms as defined in Figure 6.

$\mathsf{HGE.PG}(1^k)$:

1. $(\mathsf{par}, \mathsf{msk}) \xleftarrow{\$} \mathsf{KEM.PG}(1^k)$
2. Return $(\mathsf{par}, \mathsf{msk})$

$\mathsf{HGE.KG}(\mathsf{par}, \mathsf{msk}, \mathsf{id})$:

1. $(\mathsf{ek}, \mathsf{dk}) \xleftarrow{\$} \mathsf{KEM.KG}(\mathsf{par}, \mathsf{msk}, \mathsf{id})$
2. Return $(\mathsf{ek}, \mathsf{dk})$

$\mathsf{HGE.Enc}(\mathsf{par}, \mathsf{ek}, M, t)$:

1. $(C_1, K) \leftarrow \mathsf{KEM.Encap}(\mathsf{par}, \mathsf{ek}, t)$
2. $C_2 \leftarrow \mathsf{DEM.Enc}(K, M)$
3. Return $C = (C_1, C_2)$

$\mathsf{HGE.Dec}(\mathsf{par}, \mathsf{ek}, \mathsf{dk}, C, t)$:

1. $(C_1, C_2) \leftarrow C$
2. $K \leftarrow \mathsf{KEM.Decap}(\mathsf{par}, \mathsf{ek}, \mathsf{dk}, C_1, t)$
3. If $K = \perp$ then return $\perp$
4. $M \leftarrow \mathsf{DEM.Dec}(K, C_2)$
5. Return M (possibly as $\perp$)

Fig. 6. Hybrid General Encryption Scheme HGE

Correctness. A hybrid general encryption scheme $\mathsf{HGE} = (\mathsf{PG}, \mathsf{KG}, \mathsf{Enc}, \mathsf{Dec})$ is *correct* if, for all $(\mathsf{par}, \mathsf{msk}) \in [\mathsf{HGE.PG}]$, all plaintexts M, all identities $\mathsf{id} \in \mathsf{IDSp}$, all $(\mathsf{ek}, \mathsf{dk}) \in [\mathsf{HGE.KG}(\mathsf{par}, \mathsf{msk}, \mathsf{id})]$, and all tags $t \in \mathsf{TagSp}$, we have $\mathsf{HGE.Dec}(\mathsf{par}, \mathsf{ek}, \mathsf{dk}, \mathsf{HGE.Enc}(\mathsf{par}, \mathsf{ek}, M, t), t) = M$ with probability one, where the probability is taken over the coins of $\mathsf{HGE.Enc}$.

4.2 Hybrid General Encryption with Publicly Verifiable Ciphertexts

When defining public verifiability of $\mathsf{HGE} = (\mathsf{PG}, \mathsf{KG}, \mathsf{Enc}, \mathsf{Dec})$ schemes with respect to Ver and Dec', we can essentially reuse Definition 1 for general encryption. Note that message M output by the lightweight decryption algorithm Dec' could also be an error symbol $\perp$. As previously mentioned, in general HGE cannot satisfy Definition 2 of strictly non-trivial public verification since failure of the original decryption procedure $\mathsf{HGE.Dec}$ may not necessarily imply failure of the verification algorithm Ver'. For this reason we define the following relaxed notion:

Definition 4 (Somewhat Non-Trivial Public Verification). *Let* $\mathsf{HGE} = (\mathsf{PG}, \mathsf{KG}, \mathsf{Enc}, \mathsf{Dec})$ *be a hybrid general encryption scheme from Figure 6 that is publicly verifiable with respect to auxiliary algorithms* Ver *and* Dec'*. Let* $(\mathsf{par}, \mathsf{msk}) \leftarrow \mathsf{PG}(1^k)$*.* Ver *is said to be* somewhat non-trivial *if, for all* $\mathsf{id} \in \mathsf{IDSp}$*, all* $t \in \mathsf{TagSp}$*, and* $(\mathsf{ek}, \mathsf{dk}) \leftarrow \mathsf{KG}(\mathsf{par}, \mathsf{msk}, \mathsf{id})$*,*

1. $(\mathsf{Ver}(\mathsf{par}, \mathsf{ek}, C, t) = \perp \vee \mathsf{DEM.Dec}(K, C_2) = \perp) \Leftrightarrow \mathsf{Dec}(\mathsf{par}, \mathsf{ek}, \mathsf{dk}, C, t) = \perp$ *for all* C*, where* $C = (C_1, C_2)$ *and* $k = \mathsf{KEM.Decap}(\mathsf{par}, \mathsf{ek}, \mathsf{dk}, C_1, t)$*, and*

2. *there exists a ciphertext C for which* $\mathsf{Dec}(\mathsf{par}, \mathsf{ek}, \mathsf{dk}, C, t) = \perp$.

Condition 1 requires that successful public verification is necessary but not sufficient for the decryption algorithm to successfully decrypt. In particular, if Ver succeeds then the only reason why HGE.Dec fails is because of a failure in DEM.Dec. Condition 2 remains as in Definition 2.

Theorem 4 (proven in Appendix C) shows that if the underlying general KEM is publicly verifiable with strictly non-trivial verification then the hybrid general encryption scheme is publicly verifiable in the somewhat non-trivial way and that by outsourcing verification of the KEM part the hybrid scheme remains at least IND-CPA-secure.

Theorem 4. *Let* $\mathsf{GKEM} = (\mathsf{PG}, \mathsf{KG}, \mathsf{Encap}, \mathsf{Decap})$ *be an* IND-CCA*-secure general key encapsulation mechanism that is publicly verifiable with respect to* GKEM.Ver *and* $\mathsf{GKEM.Decap}'$, $\mathsf{DEM} = (\mathsf{Enc}, \mathsf{Dec})$ *be an* IND-OTCCA*-secure data encapsulation mechanism, and* $\mathsf{HGE} = (\mathsf{PG}, \mathsf{KG}, \mathsf{Enc}, \mathsf{Dec})$ *be the resulting hybrid general encryption scheme.*

1. *If* GKEM.Ver *is strictly non-trivial then* HGE *is publicly verifiable with respect to a somewhat non-trivial* HGE.Ver *and an algorithm* $\mathsf{HGE.Dec}'$.

2. *Let* $\mathsf{HGE}' := (\mathsf{PG}, \mathsf{KG}, \mathsf{Enc}', \mathsf{Dec}')$ *with* $\mathsf{HGE}'.\mathsf{Enc}' = \mathsf{HGE.Ver} \circ \mathsf{HGE.Enc}$ *and* $\mathsf{HGE}'.\mathsf{Dec}' = \mathsf{HGE.Dec}'$. *For any* IND-CPA *adversary* $\mathcal{A}$ *against* HGE', *there exists an* IND-CPA *adversary* $\mathcal{B}_1$ *against* GKEM' *and an* IND-OTCCA *adversary* $\mathcal{B}_2$ *against* DEM *such that*

$$\mathbf{Adv}^{\mathsf{IND\text{-}CPA}}_{\mathcal{A},\mathsf{HGE}'}(k) \leq \mathbf{Adv}^{\mathsf{IND\text{-}CCA}}_{\mathcal{B}_1,\mathsf{GKEM}'}(k) + \mathbf{Adv}^{\mathsf{IND\text{-}OTCCA}}_{\mathcal{B}_2,\mathsf{DEM}}(k) \quad \forall k \geq 0$$

and $\mathcal{B}_1$ *and* $\mathcal{B}_2$ *have (asymptotically) the same running time as* $\mathcal{A}$.

4.3 Constructions of Hybrid Encryption with Publicly Verifiable Ciphertexts

Herranz et al. [14] showed that if some IND-CCA-secure KEM is combined with an IND-OTCCA-secure DEM then the resulting hybrid encryption scheme is also IND-CCA-secure. As shown by Cramer and Shoup [10], one can easily construct an IND-OTCCA-secure DEM by adding a one-time MAC to a one-time-secure DEM such as one-time pad. Moreover, Theorem 4 states that if the underlying KEM is publicly verifiable then the resulting hybrid encryption scheme is publicly verifiable as well. We can thus immediately obtain a range of publicly verifiable constructions of hybrid encryption schemes with somewhat non-trivial verification from these two building blocks; for instance, we can apply publicly verifiable KEM constructions from Section 3.3.

In the case of tag-based KEM/DEM approach, Abe *et al.* [3] showed that IND-CCA-secure hybrid encryption can be obtained by combining an IND-CCA-secure tag-based KEM with a one-time secure DEM. They also provide constructions of IND-CCA-secure tag-based KEMs that they obtain generically from

IND-CCA-secure public-key KEMs and one-time MACs. Our publicly verifiable public-key-based KEM constructions from Section 3.3 can be used to instantiate their tag-based KEMs, resulting in further publicly verifiable hybrid encryption schemes.

5 Conclusion

In this work we formalized the notion of public verifiability for encryption schemes, KEMs, and hybrid KEM/DEM constructions. By adopting and extending the generalized syntax from [1] our definitions of publicly verifiable schemes and corresponding security results hold for public-key based, identity-based, and tag-based settings. We defined conditions under which public verifiability can be seen as a non-trivial requirement for IND-CCA security and have proven that by outsourcing verification those schemes remain at least IND-CPA secure. We showcased that well-known CHK and NIZK-based transforms offer strictly non-trivial public verification, proposed a new PKE scheme that makes most use of this property, and discussed different flavours of efficient strictly non-trivial publicly verifiable KEMs. With regard to hybrid schemes we showed that although strictly non-trivial verification is not achievable, a relaxed notion of somewhat non-trivial public verifiability can be obtained, which still offers sufficient performance gains in IND-CCA/CPA filters that are useful for applications where outsourcing of ciphertext verification to an honest-but-curios gateway without losing confidentiality is sufficient for practical purposes.

Acknowledgements

This research was supported by the Australian Technology Network (ATN) and German Academic Exchange Service (DAAD) Joint Research Co-operation Scheme. Juan Manuel González Nieto and Douglas Stebila were further supported by the Australia–India Strategic Research Fund project TA020002. Mark Manulis acknowledges support through German Research Foundation (DFG) via grant MA 4957. This work was also supported by CASED and EC-SPRIDE.

References

1. M. Abdalla, M. Bellare, and G. Neven. Robust Encryption. In D. Micciancio, editor, *TCC 2010*, *LNCS*, vol. 5978, pp. 480–497, Springer, 2010.
2. M. Abdalla, M. Bellare, and P. Rogaway. The Oracle Diffie-Hellman Assumptions and an Analysis of DHIES. In *CT-RSA 2001*, *LNCS*, vol. 2020, pp. 143–158, Springer, 2001.
3. M. Abe, R. Gennaro, and K. Kurosawa. Tag-KEM/DEM: A New Framework for Hybrid Encryption. *Journal of Cryptology*, 21(1):97–130, 2008.
4. M. Abe, E. Kiltz, and T. Okamoto. Chosen Ciphertext Security with Optimal Ciphertext Overhead. In *ASIACRYPT 2008*, *LNCS*, vol. 5350, pp. 355–371, Springer, 2008.

5. K. Bentahar, P. Farshim, J. Malone-Lee, and N. P. Smart. Generic Constructions of Identity-Based and Certificateless KEMs. *J. Cryptology*, 21(2):178–199, 2008.
6. D. Boneh and J. Katz. Improved efficiency for CCA-secure cryptosystems built using identity-based encryption. In *CT-RSA 2005*, *LNCS*, vol. 3376, pp. 87–103, Springer, 2005.
7. D. Boneh, X. Boyen, and S. Halevi. Chosen Ciphertext Secure Public Key Threshold Encryption Without Random Oracles. In *CT-RSA 2006*, *LNCS*, vol. 3860, pp. 226–243, Springer, 2006.
8. X. Boyen, Q. Mei, and B. Waters. Direct chosen ciphertext security from identity-based techniques. In V. Atluri, C. Meadows, A. Juels, editors, *ACM CCS 2005*, pp. 320–329, ACM, 2005.
9. R. Canetti, S. Halevi and J. Katz. Chosen-ciphertext security from identity-based encryption. In C. Cachin and J. Camenisch, editors, *EUROCRYPT 2004*, *LNCS*, vol. 3027, pp. 207–222, Springer, 2004.
10. R. Cramer and V. Shoup. Design and analysis of practical public-key encryption schemes secure against adaptive chosen ciphertext attack. *SIAM J. Computing*, 33(1):167–226, 2003.
11. D. Dolev, C. Dwork, and M. Naor Non-Malleable Cryptography (Extended Abstract). *ACM STOC 1991*, pp. 542-552, ACM, 1991.
12. E. Elkind and A. Sahai. A unified methodology for constructing public-key encryption schemes secure against adaptive chosen-ciphertext attack. Cryptology ePrint Archive, Report 2002/042, 2002. `http://eprint.iacr.org/`
13. G. Hanaoka and K. Kurosawa. Efficient Chosen Ciphertext Secure Public Key Encryption under the Computational Diffie-Hellman Assumption. In *ASIACRYPT 2009*, *LNCS*, vol. 5350, pp. 308–325, Springer, 2009.
14. J. Herranz, D. Hofheinz, and E. Kiltz. KEM/DEM: Necessary and sufficient conditions for secure hybrid encryption. Cryptology ePrint Archive, Report 2006/256, 2006. `http://eprint.iacr.org/`
15. H. Imai and A. Yamagishi. CRYPTREC Project — Cryptographic Evaluation Project for the Japanese Electronic Government. In *ASIACRYPT 2000*, *LNCS*, vol. 1976, pp. 399–400, Springer, 2000.
16. E. Kiltz. Chosen-Ciphertext Security from Tag-Based Encryption. In *TCC 2006*, *LNCS*, vol. 3876, pp. 581–600, Springer, 2006.
17. E. Kiltz. Chosen-Ciphertext Secure Key-Encapsulation Based on Gap Hashed Diffie-Hellman. In *PKC 2007*, *LNCS*, vol. 4450, pp. 282–297, Springer, 2007.
18. E. Kiltz and D. Galindo. Direct Chosen-Ciphertext Secure Identity-Based Key Encapsulation Without Random Oracles. In *ACISP 2006*, *LNCS*, vol. 4058, pp. 336–347. Springer, 2006.
19. C. H. Lim and P. J. Lee. Another Method for Attaining Security against Adaptively Chosen Ciphertext Attacks. In *CRYPTO 1993*, *LNCS*, vol. 773, pp. 420–434, Springer, 1993.
20. J. K. Liu, C-K. Chu and J. Zhou. Identity-Based Server-Aided Decryption. In *ACISP 2011*, *LNCS*, vol. 6812, pp. 337–352, Springer, 2011
21. B. Libert and M. Yung. Adaptively Secure Non-interactive Threshold Cryptosystems. In *ICALP 2011*, Part II, *LNCS*, vol. 6756, pp. 588–600. Springer, 2011.
22. M. Naor and M. Yung. Public-key Cryptosystems Provably Secure against Chosen Ciphertext Attacks. *ACM STOC 1990*, pp. 427-437. ACM, 1990.
23. NESSIE. Final report of European project IST-1999-12324: New European Schemes for Signatures, Integrity, and Encryption, April 2004. `https://www.cosic.esat.kuleuven.be/nessie/`.

24. D. H. Phan and D. Pointcheval. Chosen-Ciphertext Security without Redundancy. In *ASIACRYPT 2003*, *LNCS*, vol. 2894, pp. 1–18, Springer, 2003.
25. D. H. Phan and D. Pointcheval. OAEP 3-round: A Generic and Secure Asymmetric Encryption Padding. In *ASIACRYPT 2004*, *LNCS*, vol. 3329, pp. 63–78, Springer, 2004.
26. P. Persiano. About the Existence of Trapdoors in Cryptosystems. Manuscript. Available at `http://libeccio.dia.unisa.it/Papers/Trapdoor/Trapdoor.pdf`
27. J. Rompel. One-Way Functions are Necessary and Sufficient for Secure Signatures. In *STOC 1990*, pp. 387–394, ACM, 1990.
28. A. Sahai. Non-malleable non-interactive zero-knowledge and adaptive chosen-ciphertext security. *FOCS 1999*, pp. 543-553, IEEE, 1999.
29. V. Shoup. A proposal for an ISO standard for public key encryption (version 2.1). Manuscript, 2001. `http://shoup.net/papers`.
30. V. Shoup. ISO 18033-2: An emerging standard for public-key encryption, Final Committee Draft, December 2004. `http://shoup.net/iso/std6.pdf`.
31. V. Shoup and R. Gennaro. Securing Threshold Cryptosystems against Chosen Ciphertext Attack. In *EUROCRYPT 1998*, *LNCS*, vol. 1403, pp. 1-16, Springer, 1998.

A Proof of Theorem 1

Proof. Let $\mathcal{A}$ be an adversary that breaks the IND-CPA security of GE' and runs in time $t_{\mathcal{A}}$. We build an algorithm $\mathcal{B}$ running in time $t_{\mathcal{B}}$ that, using $\mathcal{A}$ as a sub-routine, breaks the IND-CCA security of GE. Let $\mathcal{C}_{\mathsf{GE}}$ denote the challenger in the associated IND-CCA security game for GE.

Algorithm $\mathcal{B}$ interacts with $\mathcal{C}_{\mathsf{GE}}$ and $\mathcal{A}$. With $\mathcal{A}$, $\mathcal{B}$ acts as a challenger playing the IND-CPA security game for GE'. In detail, $\mathcal{B}$ does the following: On input public par, $\mathcal{B}$ forwards them on to $\mathcal{A}$. At some point $\mathcal{A}$ outputs the challenge consisting of two messages M_0 and M_1, a target identity id^*, and a target tag t^*. $\mathcal{B}$ forwards M_0 and M_1 along with id^* and t^* to GE challenger $\mathcal{C}_{\mathsf{GE}}$, which in turn responds with a ciphertext C^* on M_b^* for a random bit b (unknown to $\mathcal{B}$). Since C^* is publicly verifiable, $\mathcal{B}$ hands $\bar{C}^* \leftarrow \mathsf{Ver}(\mathsf{par}, \mathsf{ek}[\mathsf{id}^*], C^*, t^*)$ as the challenge ciphertext over to $\mathcal{A}$. Eventually, $\mathcal{A}$ outputs a bit b', which $\mathcal{B}$ uses as it own output.

Queries of $\mathcal{A}$ to the oracles $\mathcal{O}_{\mathsf{EK}}$ and $\mathcal{O}_{\mathsf{DK}}$ are answered by $\mathcal{B}$ as follows:

- $\mathcal{O}_{\mathsf{EK}}(\mathsf{id})$: $\mathcal{B}$ queries $\mathcal{O}_{\mathsf{EK}}(\mathsf{id})$ to $\mathcal{C}_{\mathsf{GE}}$ and responds to $\mathcal{A}$ with whatever it receives from $\mathcal{C}_{\mathsf{GE}}$. Note that $\mathcal{A}$ is allowed to query $\mathcal{O}_{\mathsf{EK}}$ on id^*.
- $\mathcal{O}_{\mathsf{DK}}(\mathsf{id})$: $\mathcal{B}$ queries $\mathcal{O}_{\mathsf{DK}}(\mathsf{id})$ to $\mathcal{C}_{\mathsf{GE}}$ and responds to $\mathcal{A}$ with whatever it receives from $\mathcal{C}_{\mathsf{GE}}$. Note that $\mathcal{A}$ is not allowed to query $\mathcal{O}_{\mathsf{DK}}$ on id^*.

The total running time of $\mathcal{B}$ is $t_{\mathcal{B}} \leq t_{\mathcal{A}} + t_{\mathsf{Ver}}$ with $t_{\mathcal{A}}$ being the running time of $\mathcal{A}$ and t_{Ver} being the execution time of Ver.

Given the above perfect simulation of oracles, $\mathcal{B}$ clearly breaks the IND-CCA security of GE whenever $\mathcal{A}$ breaks the IND-CPA security of GE'. □

B Proof of Theorem 2

Proof. Let $\mathcal{A}$ be any PPT adversary that breaks the $\mathsf{IND\text{-}CCA}$ security of the PKE scheme with non-negligible advantage, makes at most q decryption queries and runs in time $t_{\mathcal{A}}$. Now we build an algorithm $\mathcal{B}$ running in time $t_{\mathcal{B}}$ that using $\mathcal{A}$ as a sub-routine breaks the HDH assumption with non-negligible advantage.

Before describing the algorithm $\mathcal{B}$, we define the event Forge and find an upper bound for the probability that it occurs. Let (c^*, σ^*, vk^*) be the challenge ciphertext given by $\mathcal{B}$ to $\mathcal{A}$. Let Forge be the event that $\mathcal{A}$ submits to the decryption oracle a ciphertext $(c, \sigma, vk) \neq (c^*, \sigma^*, vk^*)$ such that $(c, \sigma) \neq (c^*, \sigma^*)$ but $\mathsf{OTS.Vrfy}(c, \sigma, vk^*) = 1$. This event also includes the case that such a query is submitted by $\mathcal{A}$ before it receives the challenge ciphertext and therefore $(c, \sigma, vk) \neq (c^*, \sigma^*, vk^*)$ is not needed in this case. This implies that $\mathcal{A}$ can be used to forge the underlying one-time signature scheme OTS with probability $\Pr_{\mathcal{A}}[\mathsf{Forge}]$. The scheme OTS being a strongly unforgeable one-time signature scheme implies that $\Pr_{\mathcal{A}}[\mathsf{Forge}]$ must be negligible in the security parameter k.

We now describe how $\mathcal{B}$ proceeds. Attacker $\mathcal{B}$'s input is a random instance of the HDH problem $(u = g^a, g^b, W)$. The goal of $\mathcal{B}$ is to decide whether $W = \mathsf{H}(g^{ab})$ or W is a random bit string of appropriate length. With $\mathcal{A}$, $\mathcal{B}$ acts as a challenger playing the $\mathsf{IND\text{-}CCA}$ security game for PKE. In detail, $\mathcal{B}$ does the following:

Key Generation & Challenge. First $\mathcal{B}$ runs the key generation algorithm of OTS to generate $(vk^*, sigk^*)$. Then $\mathcal{B}$ selects $d \xleftarrow{\$} \mathbb{Z}_p^*$, computes part of the challenge ciphertext for $\mathcal{A}$ to be $(c_1^*, \pi^*) \leftarrow (g^b, (g^b)^d)$. Now $\mathcal{B}$ computes $t^* \leftarrow \mathsf{TCR}(c_1^*, vk^*)$ and $v \leftarrow u^{-t^*} \cdot g^d$ and sets the public key as (u, v). We say a ciphertext $(c = (c_1, \pi, c_2), \sigma, vk)$ is consistent if $\pi = (u^t v)^r$ for $t \leftarrow \mathsf{TCR}(c_1, vk)$ and $r = \log_g c_1$. Note that for a consistent ciphertext, the setup of the public keys implies that $\pi = (u^t v)^r = (u^t u^{-t^*} g^d)^r = (u^r)^{t-t^*} c_1^d$ and $\mathsf{H}(u^r) = \mathsf{H}((\pi / c_1^d)^{(t-t^*)^{-1}})$.

Now $\mathcal{B}$ runs $\mathcal{A}$ on input the public key (u, v).

Decryption Queries. Adversary $\mathcal{A}$ may query decryption oracle with a ciphertext (c, σ, vk) for which $\mathcal{B}$ proceeds as follows:

- If $\mathsf{OTS.Vrfy}(c, \sigma, vk) \neq 1$, then $\mathcal{B}$ returns $\perp$.
- If $\mathsf{OTS.Vrfy}(c, \sigma, vk) = 1$ and $vk = vk^*$, then the event Forge happens, so $\mathcal{B}$ halts and outputs a random bit.
- If $\mathsf{OTS.Vrfy}(c, \sigma, vk) = 1$ and $vk \neq vk^*$, then for $C = (c_1, \pi, c_2)$, $\mathcal{B}$ computes $t \leftarrow \mathsf{TCR}(c_1, vk)$ and $u^t v$ and verifies the consistency of the ciphertext by checking $\pi \stackrel{?}{=} (u^t v)^r$, i.e. $\mathcal{B}$ aborts if $\mathsf{DDH}(c_1, u^t v, \pi) \neq 1$. Otherwise there are three cases based on $t \leftarrow \mathsf{TCR}(c_1, vk)$:

  Case 1: $t = t^*$ and $c_1 = c_1^*$: Since $\mathcal{B}$ hides c_1^* information theoretically from $\mathcal{A}$, the probability that $c_1 = c_1^*$ is at least q/p, with q being an upper bound on the number of decryption queries $\mathcal{A}$. In this case, $\mathcal{B}$ outputs a random bit and aborts.

Case 2: $t = t^*$ and $c_1 \neq c_1^*$: In this case $\mathcal{B}$ finds a collision $c_1 \neq c_1^*$ but $\mathsf{TCR}(c_1, vk) = \mathsf{TCR}(c_1^*, vk^*)$. So, $\mathcal{B}$ outputs the collision and aborts.

Case 3: $t \neq t^*$: In this case $\mathcal{B}$ decrypts the message successfully as $m \leftarrow \mathsf{H}((\pi / c_1^d)^{(t-t^*)^{-1}}) \oplus c_2$ and returns the message m to $\mathcal{A}$.

GUESS. At some point, $\mathcal{A}$ outputs two different messages M_0 and M_1 of the same length where $\mathcal{A}$ wishes to be challenged. Using already computed challenge ciphertext part $(c_1^*, \pi^*) \leftarrow (g^b, (g^b)^d)$ and $(vk^*, sigk^*)$, $\mathcal{B}$ computes $c_2^* \leftarrow W \oplus M_\delta$ for a random bit δ and sets the challenge ciphertext for $\mathcal{A}$ to be (c^*, σ^*, vk^*), where $c^* \leftarrow (c_1^*, \pi^*, c_2^*)$ and $\sigma^* \leftarrow \mathsf{OTS.Sign}(sigk^*, c^*)$. Now $\mathcal{A}$ may continue its queries to the decryption oracle except for the challenge ciphertext and $\mathcal{B}$ answers them as before. Finally $\mathcal{A}$ outputs a guess bit δ'. If $\delta = \delta'$, then $\mathcal{B}$ outputs $\gamma = 1$ which means that $\mathcal{B}$'s guess is $W = \mathsf{H}(g^{ab})$. If $\delta \neq \delta'$, then $\mathcal{B}$ outputs $\gamma = 0$ which means that $\mathcal{B}$'s guess is that W is a random string.

From the above we see that unless $\mathcal{B}$ receives c_1^* from $\mathcal{A}$ (Case 1) or finds a collision in TCR, $\mathcal{B}$ simulates $\mathcal{A}$'s view perfectly as in the original PKE security experiment.

$\mathcal{B}$'S SUCCESS PROBABILITY AND RUNNING TIME. If $\mathcal{A}$ wins, then $\mathcal{B}$ also wins. Therefore we have, $\forall k \geq 0$,

$$\mathbf{Adv}_{\mathcal{B}}^{\text{HDH}}(k) \geq \mathbf{Adv}_{\mathcal{A},\mathsf{PKE}}^{\mathsf{IND\text{-}CCA}}(k) - \mathbf{Adv}_{\mathsf{TCR},\mathsf{H}}^{\text{Hash-colli}}(k) - \Pr_{\mathcal{A}}[\mathsf{Forge}] - q/p.$$

$\mathcal{B}$'s total running time is $t_{\mathcal{B}} \leq t_{\mathcal{A}} + O(t_{\mathbb{G}})$ where $t_{\mathcal{A}}$ is the running time of $\mathcal{A}$ and $t_{\mathbb{G}}$ is the time to perform a basic operation in $\mathbb{G}$. □

C Proof of Theorem 4

Proof. Statement 1. The first statement of the theorem is proven as follows: if GKEM that is used in the HGE construction is publicly verifiable (in the sense of Definition 3) then there exist two algorithms $\mathsf{GKEM.Ver}$ and $\mathsf{GKEM.Decap}'$ such that $\mathsf{GKEM.Decap} = \mathsf{GKEM.Decap}' \circ \mathsf{GKEM.Ver}$.

We construct now two algorithms $\mathsf{HGE.Ver}$ and $\mathsf{HGE.Dec}'$ as follows:

$\mathsf{HGE.Ver}(\mathsf{par}, \mathsf{ek}, C, t)$**:** Given public parameters par, the encryption key ek, a ciphertext C and a tag t, this algorithm first parses C into C_1 and C_2 and runs $\mathsf{GKEM.Ver}(\mathsf{par}, \mathsf{ek}, C_1, t)$. If $\mathsf{GKEM.Ver}(\cdot)$ outputs $\perp$ the algorithm $\mathsf{HGE.Ver}$ also outputs $C' = \perp$. Otherwise the output of $\mathsf{GKEM.Ver}(\cdot)$ is a new (transformed) ciphertext C_1' and in this case the algorithm $\mathsf{HGE.Ver}$ outputs a ciphertext $C' = (C_1', C_2)$.

$\mathsf{HGE.Dec}'(\mathsf{par}, \mathsf{ek}, \mathsf{dk}, C', t)$**:** The algorithm parses C' as (C_1', C_2) and obtains $K \leftarrow \mathsf{GKEM.Decap}'(\mathsf{par}, \mathsf{ek}, \mathsf{dk}, C_1', t)$. If $K = \perp$ then it outputs $\perp$. Otherwise, it runs $\mathsf{DEM.Dec}(K, C_2)$ and outputs its result (which could also be $\perp$).

From the above, it is easy to see that $\mathsf{HGE.Dec} = \mathsf{HGE.Dec}' \circ \mathsf{HGE.Ver}$. That is, $\mathsf{HGE.Dec}(\mathsf{par}, \mathsf{ek}, \mathsf{dk}, C, t)$ has the same input/output behavior as the following sequence of steps:

1. $(C_1, C_2) \leftarrow C$
2. $C' \leftarrow \mathsf{HGE.Ver}(\mathsf{par}, \mathsf{ek}, C, t)$
3. If $C' = \perp$ then return $\perp$
4. $M \leftarrow \mathsf{HGE.Dec}'(\mathsf{par}, \mathsf{ek}, \mathsf{dk}, C', t)$
5. Return M (possibly as $\perp$)

This construction of $\mathsf{HGE.Dec}$ implies that HGE is publicly verifiable with respect to $\mathsf{HGE.Ver}$ and $\mathsf{HGE.Dec}'$. Now observe that if either $\mathsf{HGE.Ver}$ or $\mathsf{HGE.Dec}'$ fails then so does $\mathsf{HGE.Dec}$. By construction, $\mathsf{DEM.Dec}(K, C_2) = \perp$ leads to the failure of $\mathsf{HGE.Dec}'$. Hence, we get

$$\mathsf{HGE.Ver}(\mathsf{par}, \mathsf{ek}, C, t) = \perp \vee \mathsf{DEM.Dec}\ (\ K, C_2) = \perp \Rightarrow \mathsf{HGE.Dec}(\mathsf{par}, \mathsf{ek}, \mathsf{dk}, C, t) = \perp. \tag{1}$$

As for the opposite implication observe that if $\mathsf{HGE.Dec}$ outputs $\perp$ then either $\mathsf{HGE.Ver}(\mathsf{par}, \mathsf{ek}, C, t) = \perp$ or $\mathsf{HGE.Dec}'(\mathsf{par}, \mathsf{ek}, \mathsf{dk}, C', t) = \perp$. Note that by construction, $\mathsf{HGE.Dec}'$ fails if $\mathsf{GKEM.Decap}'(\mathsf{par}, \mathsf{ek}, \mathsf{dk}, C'_1, t) = \perp$ or $\mathsf{DEM.Dec}(K, C_2) = \perp$. Since $\mathsf{GKEM.Decap} = \mathsf{GKEM.Decap}' \circ \mathsf{GKEM.Ver}$ and $\mathsf{GKEM.Ver}$ is strictly non-trivial we have $\mathsf{GKEM.Decap}'(\mathsf{par}, \mathsf{ek}, \mathsf{dk}, C'_1, t) = \perp$ if and only if $C' = \perp$. Since $C' = \perp$ is equivalent to the failure of $\mathsf{HGE.Ver}$ we have

$$\mathsf{HGE.Dec}(\mathsf{par}, \mathsf{ek}, \mathsf{dk}, C\ , t) = \perp \Rightarrow \mathsf{HGE.Ver}(\mathsf{par}, \mathsf{ek}, C, t) = \perp \vee \mathsf{DEM.Dec}(K, C_2) = \perp. \tag{2}$$

(1) and (2) mean that $\mathsf{HGE.Ver}$ is somewhat non-trivial according to Definition 4.

Statement 2. To prove the second statement, we first need to show that the hybrid general encryption scheme $\mathsf{HGE}' := (\mathsf{PG}, \mathsf{KG}, \mathsf{Enc}', \mathsf{Dec}')$ is IND-CPA-secure. Note that HGE' is a general encryption scheme that is obtained through a hybrid construction of $\mathsf{GKEM}' := (\mathsf{PG}, \mathsf{KG}, \mathsf{Encap}', \mathsf{Decap}')$ and $\mathsf{DEM} = (\mathsf{Enc}, \mathsf{Dec})$, where $\mathsf{GKEM}'.\mathsf{Encap}' := \mathsf{GKEM.Ver} \circ \mathsf{GKEM.Encap}$ and $\mathsf{GKEM}'.\mathsf{Decap}' = \mathsf{GKEM.Decap}'$, as defined in Theorem 3. Theorem 3 also says that GKEM' is IND-CPA-secure. This implies that HGE' is obtained through combination of an IND-CPA-secure KEM and an IND-OTCCA secure DEM. The result of Herranz *et al.* [14], who showed that this combination achieves at least IND-CPA security, helps us to immediately conclude the proof of the second statement. □