Detecting Dangerous Queries: A New Approach for Chosen Ciphertext Security

Susan Hohenberger*
Johns Hopkins University

Allison Lewko†
University of Texas at Austin

Brent Waters‡
University of Texas at Austin

May 16, 2012

Abstract

We present a new approach for creating chosen ciphertext secure encryption. The focal point of our work is a new abstraction that we call *Detectable Chosen Ciphertext Security* (DCCA). Intuitively, this notion is meant to capture systems that are not necessarily chosen ciphertext attack (CCA) secure, but where we can detect whether a certain query CT can be useful for decrypting (or distinguishing) a challenge ciphertext CT^*.

We show how to build chosen ciphertext secure systems from DCCA security. We motivate our techniques by describing multiple examples of DCCA systems including creating them from 1-bit CCA secure encryption — capturing the recent Myers-shelat result (FOCS 2009). Our work identifies DCCA as a new target for building CCA secure systems.

1 Introduction

A central goal of public key cryptography is to design encryption systems that are secure against chosen ciphertext attacks. Public key encryption systems that are chosen ciphertext attack (CCA) secure are robust against powerful adversaries that are able to leverage interaction with a decryptor. Such an attacker is modeled by allowing him to query for the decryption of any ciphertext except a challenge ciphertext for which he is trying to break. This includes ciphertexts derived from the challenge ciphertext[1]. Due to its robustness against powerful attackers, chosen ciphertext security has become the accepted goal for building secure encryption. For this reason, building chosen ciphertext secure systems has been a central pursuit of cryptographers for over twenty years and we have seen many distinct approaches to achieving CCA security.

*Sponsored by the Defense Advanced Research Projects Agency (DARPA) and the Air Force Research Laboratory (AFRL) under contract FA8750-11-2-0211, the Office of Naval Research under contract N00014-11-1-0470, a Microsoft Faculty Fellowship and a Google Faculty Research Award. Applying to all authors, the views expressed are those of the authors and do not reflect the official policy or position of the Department of Defense or the U.S. Government.

†Sponsored by a Microsoft Research Ph.D. Fellowship.

‡Supported by NSF CNS-0915361 and CNS-0952692, AFOSR Grant No: FA9550-08-1-0352, DARPA PROCEED, DARPA N11AP20006, Google Faculty Research award, the Alfred P. Sloan Fellowship, and Microsoft Faculty Fellowship, and Packard Foundation Fellowship.

[1]We use "CCA" and "CCA2" interchangeably in this paper.

Early pioneering work in chosen ciphertext security [24, 14, 27] introduced the technique of leveraging Non-Interactive Zero Knowledge Proofs (NIZKs) [5] to build CCA-secure encryption systems from chosen plaintext secure encryption systems. Roughly, a NIZK is used to prove that a ciphertext is "well-formed" or legal. Later Cramer and Shoup [12, 13] introduced the first practical CCA-secure systems that were built on specific number theoretic assumptions such as Decisional Diffie Hellman. These techniques implicitly embed a certain form of designated verifier Non-Interactive Zero Knowledge proofs in them. More recently, different methods for building chosen ciphertext security from Identity-Based Encryption [7] and Lossy Trapdoor Functions [26] have emerged. In addition, Myers and shelat [23] described general methods for amplifying CCA encryption of 1 bit to many bits.

In this work, we introduce a new approach to obtaining chosen ciphertext secure systems. The focal point of our work is a new abstraction that we call *Detectable Chosen Ciphertext Security* (DCCA). Intuitively, this notion is meant to capture systems that are not necessarily CCA secure, but where we can detect whether a certain query CT can be useful for decrypting (or distinguishing) a challenge ciphertext CT^*.

A system that is DCCA secure will be associated with a boolean function F that takes in three inputs: a public key pk, a challenge ciphertext CT^* and a query ciphertext CT. The function will output 1 if the query CT is "dangerous" for the an attacker wishing to distinguish CT^*. A DCCA secure system must have the following two properties stated here informally:

- **Unpredictability** Without seeing CT^* it should be hard to find a ciphertext CT such that $F(\mathrm{PK}, \mathrm{CT}^*, \mathrm{CT}) = 1$. In other words, an attacker must first see a challenge ciphertext in order to discover a dangerous query for it.

- **Indistinguishability** The system will be secure under a detectable chosen ciphertext attack *if* the attacker is limited to decryption queries of ciphertexts CT where $F(pk, \mathrm{CT}^*, \mathrm{CT}) = 0$ for challenge ciphertext CT^*. I.e. the system is CCA secure if the attacker does not make dangerous queries.

The goal of our work will be to construct fully chosen ciphertext secure systems from detectable CCA-secure systems. We first motivate this goal by observing multiple DCCA systems that naturally occur:

- **Many bit encryption from 1-bit CCA** Suppose we have a 1-bit CCA-secure system and we wish to encrypt multiple bits by concatenating multiple 1-bit encryptions together. The resulting system is no longer chosen ciphertext secure, but is DCCA secure. The detecting function F is 1 iff any of the 1-bit ciphertext components between CT^* and CT are equal. This scenario is akin to the problem of showing that bit encryption is complete considered by Myers and shelat [23], where they worried about such "quoting" attacks.

- **Tag-Based Encryption Systems** MacKenzie, Reiter and Yang [22] and Kiltz [20] define a tag-based encryption scheme as an encryption scheme that takes in an additional "tag" parameter on encryption and decryption. The security game allows an attacker to make decryption queries with any tag parameter t, except for the tag t^* that the challenge ciphertext is encrypted under. Several examples of tag-based schemes exist. Kiltz [20] gave a direct construction from the linear assumption. The CCA1-secure encryption variant of the Canetti, Halevi and Katz [7] construction where the tag is an IBE identity is an additional example.

One can also view the CCA1-secure variant of Peikert and Waters [26] as a tag-based scheme, where the tag is the "branch" in an all-but-one encryption scheme.

Most of the above examples of tag-based encryption can be proven selectively secure, where an attacker must commit to the tag of the challenge ciphertext before seeing the public key. However, if we are willing to utilize complexity leveraging arguments, we can argue that these are adaptively secure. In addition, the CHK-lite transformation will be an adaptively secure tag-based scheme if used with an adaptively secure Identity-Based Encryption system.

We observe that adaptively-secure tag-based encryption immediately gives rise to DCCA-secure encryption. A ciphertext of the DCCA-secure system consists of a random tag t plus a tag-based encryption of the message under the tag t. Decryption follows analogously and the function F simply tests if two ciphertexts have the same tag. Unpredictability follows from having a large tag space. Although it is already possible to transform tag-based encryption into CCA-secure encryption using a strongly unforgeable signature [20], these examples demonstrate natural DCCA systems. We detail this argument in Appendix D.

- **"Sloppy" CCA Encryption** One can envision that in practice an encryption system is CCA secure, but an implementation of it is not due to certain nuances. For instance, suppose a number theoretic library had a slack bit in its representation of group elements (e.g. a bit that was always supposed to be 0, but if set to 1 does not affect any computations.) A CCA attacker could exploit this weakness in an implementation, however, it is possible that the system would still be DCCA secure. Thus, one might use our techniques as a hedge against such problems. This is somewhat analogous to recent work [2] on applying deterministic encryption as a hedge against faulty random bit generation.

In addition to the examples listed above, we believe that it is useful to identify DCCA security as a new "target" for achieving chosen ciphertext security.

Overview of Our Techniques We now give an overview of our construction and proof. Our construction will build a chosen ciphertext secure system from three components: a chosen plaintext secure system, 1-bounded CCA-secure system [2], and a detectable CCA-secure system. Since DCCA security (trivially) implies CPA, and we can build 1-bounded CCA from CPA encryption [25, 11, 10], it follows that all components are realizable from DCCA as a building block.

A public key from our system consists of three components. An "inner" public key $\mathrm{PK}_{\mathrm{in}}$ which is a DCCA public key and two "outer" keys $\mathrm{PK}_A, \mathrm{PK}_B$ respectively from 1-bounded CCA and CPA secure systems. To encrypt a message M, one first chooses the randomness r_A, r_B to be used for the outer encryptions and then encrypts the tuple (r_A, r_B, M) under the inner (detectable) key to compute an inner ciphertext $\mathrm{CT}_{\mathrm{in}}$. Next, the encryption algorithm encrypts $\mathrm{CT}_{\mathrm{in}}$ under the outer public key PK_A using randomness r_A to get CT_A. It then analogously creates CT_B as the encryption of $\mathrm{CT}_{\mathrm{in}}$ under key PK_B and randomness r_B. The output ciphertext is $\mathrm{CT} = (\mathrm{CT}_A, \mathrm{CT}_B)$.

The structure of our ciphertexts is that the two outer ciphertexts both encrypt the same message — the inner ciphertext. This ciphertext itself encrypts the message and the randomness used to create the outer ciphertexts. Thus, the outer ciphertexts indirectly encrypt their own randomness. [3]

[2] A 1-bounded CCA-secure encryption system is secure against one chosen ciphertext query.

[3] This construction implicitly assumes that the length of the random string needed for encryption is dependent only on the security parameter and is independent (or at least smaller than) the message size of the outer ciphertexts.

The decryption algorithm will receive $\mathrm{CT} = (\mathrm{CT}_A, \mathrm{CT}_B)$ and first decrypt CT_A to get $\mathrm{CT}'_{\mathrm{in}}$ and decrypt this to get (r_A', r_b', M') using the appropriate secret keys. Finally, it will check that the ciphertext is well formed by itself encrypting $\mathrm{CT}'_{\mathrm{in}}$ under $\mathrm{PK}_A, \mathrm{PK}_B$ and the respective randomness r_A', r_B' and validating that the output matches CT_A and CT_B before accepting M' as the message. Our encryption system has elements both of the Naor-Yung [24] two key method for our two outer keys and the Myers-shelat [23] method of embedding outer randomness in inner ciphertexts.

Security of our system depends on the premise that no attacker is able to learn the message encrypted in the inner ciphertext. This will follow from the Detectable CCA security *if* we are able to guarantee that an attacker is unable to make any ciphertext queries $\mathrm{CT}_A, \mathrm{CT}_B$ where the decryption of CT_A, denoted $\mathrm{CT}_{\mathrm{in}}$, is related to the inner component of our challenge ciphertext $\mathrm{CT}^*_{\mathrm{in}}$ according to to the DCCA function F. Intuitively, we hope to achieve this from the combination of two features of our system. First, the 1-bounded CCA security of PK_A will (hopefully) make it difficult to create an encryption under PK_A related to $\mathrm{CT}^*_{\mathrm{in}}$. Second, the embedded randomness will allow us to check that ciphertexts are well formed and thus answer multiple ciphertext queries under the Naor-Yung two key type manner.

The trickiness in proving security lies with the embedded randomness which is a two-edge sword. On one hand, forcing the attacker queries to embed randomness allows a reduction algorithm to decrypt if it knows either one of the two outer keys. On the other hand, it is not clear how such a reduction can create valid ciphertexts while playing the 1-bounded CCA game, since a reduction algorithm will not know the randomness r_A to embed. Thus, this circularity creates a fundamental barrier similar to difficulties encountered in attempts to create trapdoor functions from encryption [15].

We deal with this by arguing security in an indirect way that steps around this barrier. We first define a security game specific to our construction called nested indistinguishability. In this game, an attacker will receive a public key and is allowed to make decryption queries. The attacker at some point submits a single message M. The challenger will flip a coin z. If $z = 0$, the challenger creates a valid encryption of M; otherwise, if $z = 1$ the challenger creates a encryption where the innermost message is all 0's — it neither includes the message nor the embedded randomness. The attacker continues to make decryption queries (other than the challenge ciphertext) and wins if it is successfully able to guess z. It follows that if no attacker is successful in this game, then our system is chosen ciphertext secure.

To prove security of this nested indistinguishability game, we begin by defining a "bad event". The bad event is defined to be when the attacker submits a query $(\mathrm{CT}_A, \mathrm{CT}_B)$ such that $\mathrm{CT}_A \neq \mathrm{CT}^*_A$ where CT^*_A is from the challenge ciphertext and the decryption of CT_A gives a ciphertext that is related to the inner challenge ciphertext according to F. If we can argue that such bad events only occur with negligible probability, then security of the nested indistinguishability game follows straightforwardly from DCCA security.

The crux of our proof is how we eliminate the possibility of a bad event. We do so in an indirect manner. We begin by arguing this event cannot happen in the case where $z = 1$, which is where all 0's are encrypted and the randomness is not embedded. In this case, we get the best of both worlds. We are able to require that the attacker's queries have the randomness embedded in them,

We can justify this assumption with the common technique of using a seed to a (variable length) Pseudo Random Generator (PRG) as the input to each encryption algorithm. The PRG can then extend the randomness to whatever length is required by the underlying encryption system. By using this justified assumption in our definitions, we are able to simplify the presentation of our construction and proofs. In contrast, Myers and shelat [23] explicitly carry the PRG technique through their exposition. This choice gives our exposition and proof an advantage in simplicity.

so that we can check ciphertext well-formedness, however, *the challenge ciphertext is not required to embed the outer randomness*. We argue that the bad event does not happen by applying a set of hybrid experiments. First, we change CT_B^* to be an encryption of all 1's. Next, we change the decryption algorithm to decrypt using the secret key for PK_B. Finally, we change CT_A^* to be an encryption of all 1's. In each experiment we argue that the chance of a bad event must be very close to that of the prior experiment. For the last step we leverage the 1-bounded CCA property of the first component. Finally, we note that in the last experiment the probability of a bad event is negligible since the inner challenge ciphertext $\mathrm{CT}_{\mathrm{in}}^*$ is replaced by all 1's and is not even present.

One interesting question is why is 1-bounded CCA security needed for the PK_A since at the last step in the proof we can use the secret key SK_B to execute decryption. While this is true, it is actually possible for the bad event to occur on a malformed ciphertext that will not decrypt. We need the 1-bounded CCA property to detect the occurrence of the bad event in this case during the security reduction.

We are not able to argue the lack of a bad event in a similar manner for the $z = 0$ (embedded randomness) case due to the aforementioned circularity problems. Instead, we can infer this from the lack of event in the $z = 1$ case along with DCCA security. To prove this, we can create an algorithm that plays the DCCA indistinguishability game while simulating the nested indistinguishability game to the attacker. The simulator will choose the outer keys and outer randomness for the challenge ciphertext itself. It submits the message and outer randomness as one inner message and the 0's string as another. Then it will be able to decrypt all ciphertext queries until a bad event happens using its keys in addition to the DCCA decryption oracle. Once a bad event query is made though, it is stuck. However, it need not go any further! The fact that the attacker was able to create a bad event at all must mean that the message and randomness were embedded. It can then break the DCCA distinguishing game. Thus, we can infer that the bad event happens with negligible probability in either case. The remainder of the proof follows straightforwardly.

Comparison to Myers-shelat Myers and shelat [23] showed how to achieve many-bit chosen ciphertext security from 1-bit chosen ciphertext security and motivated us to explore the notion of detectability. They created a system using an inner/outer structure where the inner ciphertext encrypted the outer random coins. Their inner scheme, built from 1-bit CCA, is what they call "unquoteable" secure. Their concept is roughly analogous to a specific instance of a DCCA scheme. Encryptions of many-bit messages are concatenations of 1-bit encryptions; the system is chosen ciphertext secure as long as queries do not copy a 1-bit ciphertext component of the underlying scheme. For the outer scheme, they use a notion of security that is an amalgam of unquoteability and non-malleability. Their outer construction follows a specific adaptation of the Choi et. al. [10] methods applied to the 1-bit primitive. (No two key structure is used.) Their proof relies on defining quoting attacks on *both* the inner and outer layers and then establishing a certain order that outer quoting attacks must happen before inner quoting attacks.

We believe our methods offer benefits in terms of generality, simplicity, and efficiency. First, our general notion of Detectable Chosen Ciphertext Security can be realized by multiple systems. These include the 1-bit to many-bit examples, the tag-based encryption class and future systems that can leverage this as a new target path for creating CCA secure encryption.

Another key difference is that the outer layer of our scheme is built from simple 1-bounded CCA and CPA-secure parts. We argue these provide simpler concepts and are easier to work with. In addition, one can instantiate them from *any* 1-bounded encryption system. For instance, we

can apply any candidate 1-bounded CCA-secure system and do not need to work through the Choi et. al. [10] construction. Instead we can apply the 1-bounded CCA system of Cramer et. al. [11], which is signficantly more efficient and simpler than the non-malleable systems of either PSV [25] or Choi et. al. [10]. We also regard avoiding a combination security definition between 1-bounded CCA (or non-malleability) and detection as a benefit for simplicity. This simplification will also improve efficiency in the case where there is a candidate CPA primitive that is more efficient than the candidate DCCA primtive, since we can build the 1-bounded scheme out of the CPA primitive.

Our choice of abstractions and structure allow us to have a simple proof. We can eliminate the possibility of a bad event using a basic Naor-Yung two key argument. Then once we are able to eliminate this, the rest of the proof follows in a straightforward manner.

Why not CCA1? One intriguing possibility is to try to leverage our techniques to build full chosen ciphertext security from CCA1 security. A natural direction would be to use a CCA1 system for the inner component in place of the detectable encryption scheme. The intuitive rationale would be if the outer keys are 1-bounded CCA or non-malleable then the queries produced by the attacker should not be related to the inner challenge ciphertext and thus CCA1 might suffice. Unfortunately, we were able to create an attack oracle which breaks full CCA security in our scheme, yet does not perturb the 1-bounded CCA or CCA1 primitives, giving evidence that this approach may not work. However, the oracle we use is quite strong and "exotic". This suggests that there might be primitives that lie somewhere in between DCCA and CCA1. One interesting example is the CCA-1 secure "Cramer-Shoup lite" [12] cryptosystem. There exists a malleability attack on a challenge CT^* that produces a query ciphertext which has the same distribution as a fresh encryption of a random message. Hence the CS-lite system is not CCA secure. However, it would be very interesting and surprising if there existed attack algorithms that matched the above oracle.

We describe these issues in more detail in Section 5.

1.1 Related Work

Relaxations of CCA Multiple relaxations of chosen ciphertext security have been proposed in the literature.

One interesting class of relaxations is the notion of Replayable Chosen Ciphertext Security [8] and other similar works [30, 1]. These works aim to capture the concept that some malleability attacks might intuitively be benign. In particular, consider a cryptosystem where an attacker is only able to maul a ciphertext CT encrypting a message M into a different ciphertext C' that encrypts the *same* message M. If an application (or user) makes all decisions based on the decrypted plaintexts as opposed to the representation of the ciphertext such notions might be sufficient.

The primary goal of RCCA is to formally capture a form of "good enough" security under ciphertext attacks. In contrast, Detectable CCA inherently does not have good enough security on its own. In DCCA systems, it may be possible to maul ciphertexts to be encryptions of different messages or even create attack ciphertexts that each target a single bit of a target ciphertext. Thus, our primary focus is to create CCA security from a fundamentally less secure DCCA building block.

We observe that DCCA does not imply RCCA. In [8], the authors gave an example of an RCCA scheme that could not be publicly detected. Conversely, not all DCCA schemes will be RCCA secure. Our bit encryption instance serves as an example. We also note that [8] discusses a notion of detectability and introduces a definition that combines replayable and detectable properties. This combined definition is a particular instance of DCCA. However, they do not explore the notion of

detectability in isolation or how to build CCA security from it. Canetti, Krawcyzk, and Nielsen [8] do show how to create CCA security from RCCA security using the KEM/DEM framework.

Finally, Hofheinz and Kiltz [17] introduce a notion they call Constrained CCA security particular to developing Key Encapsulation Mechanisms. In their definition an attacker must include a predicate p along with each query ciphertext CT. The challenger will only answer the query if the predicate evaluated on the decrypted key of the ciphertext is true and the predicate is false for all but a negligible fraction of possible KEM keys. While this notion is weaker than CCA security, they show that when combined with a (symmetric) authenticated encryption scheme, the resulting system is CCA secure.

Other Related Work Goldwasser and Micali [16] gave the first formal definition of security for public key encryption systems. Naor and Yung [24] and Rackoff and Simon [27] extended this to include chosen ciphertext attacks.

Naor and Yung [24] initiated the approach of leveraging NIZKs to build chosen ciphertext security by introducing their "two key" method. A NIZK would guarantee the integrity of the ciphertext by giving a proof that the same message was encrypted to two keys. While their system gave security against lunchtime or CCA1 attacks, Dolev, Dwork and Naor [14] showed how to achieve full CCA2 security. In addition, they introduced the fundamental concept of non-malleability. Sahai [29] introduced a concept of simulation sound NIZKs that could be used to achieve CCA security through the NY two key structure. Bellare and Sahai [4] gave relations between non-malleability [14] chosen ciphertext security.

Since then, different approaches to achieving CCA security have been proposed. Cramer and Shoup [12, 13] showed techniques for proving ciphertexts were well-structured and abstracted this into projective hash functions. Several other novel cryptosystems make use of specific number-theoretic techniques (e.g. [20, 9, 18]). Boneh, Canetti, Halevi and Katz [6] showed a generic method of achieving chosen ciphertext security from IBE systems. Peikert and Waters [26] gave a new avenue for achieving CCA security with the introduction of Lossy Trapdoor Functions (TDFs). Notably, this gave the first chosen ciphertext secure systems from lattice-based assumptions. Subsequently, various refinements of weaker conditions on the trapdoor functions were introduced [28, 21].

The above techniques are proven secure in the standard model. Bellare and Rogaway [3] show that in the random oracle model chosen ciphertext security can be built from chosen plaintext security.

2 Detectable Chosen Ciphertext Security

In this section, we define *detectable chosen ciphertext security*. An encryption scheme satisfying this definition is called a *detectable* encryption system. Our discussions assume a familiarity with CPA, CCA1 and CCA2 security as well as bounded CCA security and non-malleability. A reader wishing to review these definitions can find them in Appendix A.

2.1 Detectable Encryption

We define a *detectable* encryption scheme as having the usual algorithms $(\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$, as defined in Definition A.1, together with an efficiently-computable boolean function F. Informally, F tests for a "detectable" relationship between two ciphertexts. The security game will mirror

that of CCA2 security, except that decryption queries in the second phase will not be answered for ciphertexts detectably-related to the challenge ciphertext. Our formal definition follows below.

Definition 2.1 (Detectable Encryption System) *A* detectable encryption system *is a tuple of probabilistic polynomial-time algorithms* $(\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec}, F)$ *such that:*

1. $(\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$ *satisfy Definition A.1, where we sometimes denote* $\mathsf{Enc}(pk, m; r)$ *as a deterministic function of the public key* pk*, the message* m *and randomness* r*, and*
2. $F(pk, c', c) \to \{0, 1\}$ *: the detecting function* F *takes as input a public key* pk *and two ciphertexts* c' *and* c*, and outputs a bit.*

Correctness is the same as a regular encryption system.

A detectable encryption system must have two properties, which we now define.

Unpredictability of the Detecting Function F. Informally, given the description of F and a public key pk, for an unknown ciphertext c, it should be hard to find a second ciphertext c' that is "related" to c; i.e., such that $F(pk, c', c) = 1$. We consider both a basic and a strong formalization.

Basic Unpredictability Experiment. Consider the following experiment $\mathsf{Exp}^{\mathsf{predict.basic}}_{\mathcal{A},\Pi}(\lambda)$ defined for a detectable encryption scheme $\Pi = (\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec}, F)$ and an adversary $\mathcal{A}$:

1. Setup: $\mathsf{KeyGen}(1^\lambda)$ is run to obtain keys (pk, sk).
2. Queries: Adversary $\mathcal{A}$ is given pk and access to a decryption oracle $\mathsf{Dec}(sk, \cdot)$. The adversary outputs a message m in the message space associated with pk and a ciphertext c in the ciphertext space associated with pk.
3. Challenge: A ciphertext $c^* \leftarrow \mathsf{Enc}(pk, m)$ is computed.
4. Output: The output of the experiment is defined to be 1 if $F(pk, c^*, c)$, and 0 otherwise.

We also define a stronger variant $\mathsf{Exp}^{\mathsf{predict.strong}}_{\mathcal{A},\Pi}(\lambda)$ of the unpredictability experiment where the adversary is additionally given sk. We observe that strong unpredictability implies basic unpredictability since the adversary can simulate the decryption oracle using the secret key.

Indistinguishability of Encryptions. Next, we formalize the confidentiality guarantee. Consider the following experiment $\mathsf{Exp}^{\mathsf{indist}}_{\mathcal{A},\Pi}(\lambda)$ defined for a detectable encryption scheme $\Pi = (\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec}, F)$ and an adversary $\mathcal{A}$:

1. Setup: $\mathsf{KeyGen}(1^\lambda)$ is run to obtain keys (pk, sk).
2. Phase 1: Adversary $\mathcal{A}$ is given pk and access to a decryption oracle $\mathsf{Dec}(sk, \cdot)$. $\mathcal{A}$ outputs a pair of messages m_0, m_1 of the same length in the message space associated with pk.
3. Challenge: A random bit $b \leftarrow \{0, 1\}$ is chosen, and then a ciphertext $c^* \leftarrow \mathsf{Enc}(pk, m_b)$ is computed and given to $\mathcal{A}$. We call c^* the challenge ciphertext.
4. Phase 2: $\mathcal{A}$ continues to have access to $\mathsf{Dec}(sk, \cdot)$, but may not request a decryption of a ciphertext c such that $F(pk, c^*, c) = 1$. Finally, $\mathcal{A}$ outputs a bit b'.
5. Output: The output of the experiment is defined to be 1 if $b' = b$, and 0 otherwise.

Definition 2.2 (Detectable Chosen Ciphertext Security) *A detectable encryption scheme* $\Pi = (\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec}, F)$ *has an* unpredictable detecting function *and* indistinguishable encryptions under a detectable chosen-ciphertext attack *(or is* DCCA-secure*) if for all probabilistic polynomial-time adversaries* $\mathcal{A}$ *there exists a negligible function negl such that:*

1. *(*F *is unpredictable:)* $\Pr[\mathsf{Exp}^{\mathsf{predict.basic}}_{\mathcal{A},\Pi}(\lambda) = 1] \leq negl(\lambda)$ *and*
2. *(Encryptions are indistinguishable:)* $\Pr[\mathsf{Exp}^{\mathsf{indist}}_{\mathcal{A},\Pi}(\lambda) = 1] \leq \frac{1}{2} + negl(\lambda)$.

2.2 Facts about DCCA Security

For space reasons, we omit the simple proofs of the first two claims. We conjecture that the converse of Claim 2.4 is not true. Indeed if the DDH assumption holds, then the CCA-1 secure Cramer-Shoup lite system would separate these two notions as discussed in the introduction.

Claim 2.3 (CCA2 $\Longrightarrow$ DCCA) *If* $\Pi = (\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$ *is a CCA2-secure encryption scheme, then* $\Pi' = (\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec}, F)$ *is a DCCA-secure encryption scheme where* F *outputs 0 on all inputs except those of the form* $(\cdot, c, c)$.

Claim 2.4 (DCCA $\Longrightarrow$ CCA1) *If* $\Pi = (\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec}, F)$ *is a DCCA-secure encryption scheme, then* $\Pi' = (\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$ *is a CCA1-secure encryption scheme.*

We also claim that one-bit DCCA-secure encryption implies arbitrary-length DCCA-secure encryption. Say $\Pi = (\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec}, F)$ is a detectable encryption system with plaintext space $\{0,1\}$. We can construct a new scheme $\Pi' = (\mathsf{KeyGen}, \mathsf{Enc}', \mathsf{Dec}', F')$ with plaintext space $\{0,1\}^*$ by defining Enc' as follows:

$$\mathsf{Enc}'(pk, m) = \mathsf{Enc}(pk, m_1), \ldots, \mathsf{Enc}(pk, m_n)$$

where $m = m_1 \ldots m_n$. The decryption algorithm Dec' decrypts each ciphertext piece using Dec. The function F' performs n^2 invocations of F, testing each ciphertext piece of C with each ciphertext piece of C', and outputting 1 if any invocation of F returned 1, and 0 otherwise.

Lemma 2.5 (1-bit DCCA encryption implies arbitrary-length DCCA encryption) *Let* Π *and* Π' *be as above. If* Π *is DCCA-secure, then so is* Π'.

We prove this lemma in Appendix B.

3 The Construction: CCA2 Security from DCCA Security

An overview of the techniques used for our construction is provided in Section 1.

The Construction Description We now construct a CCA2-secure public-key encryption scheme $\Pi = (\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$ using three building blocks[4]:

[4] A 1-bounded CCA-secure encryption system is secure if an attacker makes at most one decryption query. One-bounded CCA security can be constructed from CPA security [25, 10]. See Appendix A. CPA security is trivially implied by DCCA security. Thus, there is really only one necessary building block: a DCCA-secure system.

1. a DCCA-secure encryption scheme, denoted $\Pi_{\text{dcca}} = (\mathsf{KeyGen}_{\text{dcca}}, \mathsf{Enc}_{\text{dcca}}, \mathsf{Dec}_{\text{dcca}}, F)$.
2. a 1-bounded CCA-secure encryption scheme with perfect correctness, denoted $\Pi_{\text{1b-cca}} = (\mathsf{KeyGen}_{\text{1b-cca}}, \mathsf{Enc}_{\text{1b-cca}}, \mathsf{Dec}_{\text{1b-cca}})$.
3. a CPA-secure encryption scheme with perfect correctness, denoted $\Pi_{\text{cpa}} = (\mathsf{KeyGen}_{\text{cpa}}, \mathsf{Enc}_{\text{cpa}}, \mathsf{Dec}_{\text{cpa}})$.

We assume that the message space of each system is $\{0,1\}^*$ and that messages of the form (x, y, z) can be uniquely and efficiently encoded as strings in $\{0,1\}^*$, where the encoding length is the same for all inputs of the same length. We assume that λ bits will be sufficient randomness for the encryption algorithm of each system, where 1^λ is the security parameter. We assume that $\Pi_{\text{1b-cca}}$ and Π_{cpa} have perfect correctness for decryption. Finally, we assume that for Π_{dcca} the ciphertext length is a deterministic function of the security parameter and the message length. We discuss the justification behind these assumptions in Section B.

$\mathsf{KeyGen}(1^\lambda)$ Run $\mathsf{KeyGen}_{\text{dcca}}(1^\lambda)$ to produce $(\text{PK}_{\text{in}}, \text{SK}_{\text{in}})$. Run $\mathsf{KeyGen}_{\text{1b-cca}}(1^\lambda)$ to produce $(\text{PK}_A, \text{SK}_A)$. Run $\mathsf{KeyGen}_{\text{cpa}}(1^\lambda)$ to produce $(\text{PK}_B, \text{SK}_B)$. Set the public key as $\text{PK} := (\text{PK}_{\text{in}}, \text{PK}_A, \text{PK}_B)$ and the secret key as $\text{SK} := (\text{SK}_{\text{in}}, \text{SK}_A, \text{SK}_B)$.

$\mathsf{Enc}(\text{PK}, M)$ The encryption algorithm first chooses three random strings $r_{\text{in}}, r_A, r_B \in \{0,1\}^\lambda$. Next, it computes the ciphertext $\text{CT}_{\text{in}} := \mathsf{Enc}_{\text{dcca}}(\text{PK}_{\text{in}}, (r_A, r_B, M); r_{\text{in}})$. It then treats this ciphertext as the message and computes $\text{CT}_A := \mathsf{Enc}_{\text{1b-cca}}(\text{PK}_A, \text{CT}_{\text{in}}; r_A)$ and $\text{CT}_B := \mathsf{Enc}_{\text{cpa}}(\text{PK}_B, \text{CT}_{\text{in}}; r_B)$. Finally, it outputs the encryption as $(\text{CT}_A, \text{CT}_B)$.

$\mathsf{Dec}(\text{SK}, \text{CT})$ The decryption algorithm takes a ciphertext $\text{CT} := (\text{CT}_A, \text{CT}_B)$. It decrypts the first ciphertext as $\text{CT}_{\text{in}} := \mathsf{Dec}_{\text{1b-cca}}(\text{SK}_A, \text{CT}_A)$. It then decrypts this output as $(r_A, r_B, M) := \mathsf{Dec}_{\text{dcca}}(\text{SK}_{\text{in}}, \text{CT}_{\text{in}})$. It then checks that

$$\text{CT}_A = \mathsf{Enc}_{\text{1b-cca}}(\text{PK}_A, \text{CT}_{\text{in}}; r_A) \text{ and } \text{CT}_B = \mathsf{Enc}_{\text{cpa}}(\text{PK}_B, \text{CT}_{\text{in}}; r_B).$$

If all checks pass, it outputs M; otherwise, it outputs $\perp$.

4 Proof of Security

We will now argue that the Section 3 construction is CCA2 secure, assuming the respective security properties of the underlying building blocks. To do so, it will be easier to consider a slight variant of the CCA2 security game, which we call *nested indistinguishability*, where the challenger either encrypts one of the two challenge messages or encrypts a string of zeros. The experiment involves three encryption schemes and combines them in the same manner as our main construction.

Nested Indistinguishability. Consider the experiment $\mathsf{Exp}^{\mathsf{nested}}_{\mathcal{A},\Pi_{\text{dcca}},\Pi_{\text{1b-cca}},\Pi_{\text{cpa}}}(\lambda)$ defined for detectable encryption scheme Π_{dcca}, encryption schemes $\Pi_{\text{1b-cca}}, \Pi_{\text{cpa}}$ and an adversary $\mathcal{A}$:

1. Setup: Run $\mathsf{KeyGen}_{\text{dcca}}$, $\mathsf{KeyGen}_{\text{1b-cca}}$ and $\mathsf{KeyGen}_{\text{cpa}}$ to obtain key pairs $(\text{PK}_{\text{in}}, \text{SK}_{\text{in}})$, $(\text{PK}_A, \text{SK}_A)$ and $(\text{PK}_B, \text{SK}_B)$ respectively. Set $pk := (\text{PK}_{\text{in}}, \text{PK}_A, \text{PK}_B)$ and $sk := (\text{SK}_{\text{in}}, \text{SK}_A, \text{SK}_B)$.

2. Phase 1: Adversary $\mathcal{A}$ is given pk and access to a decryption oracle $\mathsf{Dec}(sk, \cdot)$, which executes the decryption algorithm as defined in Section 3. $\mathcal{A}$ outputs a pair of messages m_0, m_1 of the same length in the message space associated with pk.

3. Challenge: Randomness $\beta, z \leftarrow \{0,1\}$ and $r_A, r_B \leftarrow \{0,1\}^\lambda$ are chosen. Let ℓ denote the length of the encoding of (r_A, r_B, m_β). Then compute:

$$\mathrm{CT}^*_{\mathrm{in}} := \begin{cases} \mathsf{Enc}_{\mathrm{dcca}}(\mathrm{PK}_{\mathrm{in}}, (r_A, r_B, m_\beta)) & \text{if } z = 0; \\ \mathsf{Enc}_{\mathrm{dcca}}(\mathrm{PK}_{\mathrm{in}}, 0^\ell) & \text{if } z = 1. \end{cases} \quad (1)$$

   Next compute $\mathrm{CT}^*_A := \mathsf{Enc}_{\mathrm{1b-cca}}(\mathrm{PK}_A, \mathrm{CT}^*_{\mathrm{in}}; r_A)$ and $\mathrm{CT}^*_B := \mathsf{Enc}_{\mathrm{cpa}}(\mathrm{PK}_B, \mathrm{CT}^*_{\mathrm{in}}; r_B)$. Return to $\mathcal{A}$ the ciphertext $\mathrm{CT}^* := (\mathrm{CT}^*_A, \mathrm{CT}^*_B)$.

4. Phase 2: $\mathcal{A}$ continues to have access to $\mathsf{Dec}(sk, \cdot)$, but may not request a decryption of the challenge ciphertext CT^*. Finally, $\mathcal{A}$ outputs a bit z'.

5. Output: The output of the experiment is defined to be 1 if $z' = z$, and 0 otherwise.

Definition 4.1 (Nested Indistinguishability) *A tuple of systems* $(\Pi_{\mathrm{dcca}}, \Pi_{\mathrm{1b-cca}}, \Pi_{\mathrm{cpa}})$ *has* nested indistinguishable encryptions under a chosen-ciphertext attack *if for all probabilistic polynomial-time adversaries* $\mathcal{A}$ *there exists a negligible function negl such that:*

$$\Pr[\mathsf{Exp}^{\mathsf{nested}}_{\mathcal{A},\Pi_{\mathrm{dcca}},\Pi_{\mathrm{1b-cca}},\Pi_{\mathrm{cpa}}}(\lambda) = 1] \leq \frac{1}{2} + negl(\lambda).$$

It is important to observe that the nested indistinguishability experiment combines $\Pi_{\mathrm{dcca}}, \Pi_{\mathrm{1b-cca}}, \Pi_{\mathrm{cpa}}$ in exactly the same manner as the Section 3 construction. When $z = 1$, it encrypts "properly" and when $z = 0$, it encrypts all zeros.

With a goal of proving CCA2 security, our main task is to argue that our Section 3 construction provides nested indistinguishability. To do this, we must first establish that a certain event does not happen, except with negligible probability. We define this event as follows.

Definition 4.2 (The Bad Query Event) *Let* Π_{dcca}, $\Pi_{\mathrm{1b-cca}}$, *and* Π_{cpa} *be the schemes parameterizing the experiment* $\mathsf{Exp}^{\mathsf{nested}}$. *Let* PK_{in} *be the public key output by running* $\mathsf{KeyGen}_{\mathrm{det}}$ *during the course of the experiment. We say that a* bad query event *has occurred during an execution of this experiment if in Phase 2, the adversary* $\mathcal{A}$ *makes a decryption query of the form* $\mathrm{CT} := (\mathrm{CT}_A, \mathrm{CT}_B)$ *such that*

- *(Query inner is "related" to challenge inner:)* $F(\mathrm{PK}_{in}, \mathrm{CT}^*_{in}, \mathsf{Dec}_{\mathrm{1b-cca}}(\mathrm{SK}_A, \mathrm{CT}_A)) = 1$, *and*
- *(Query ciphertext differs from challenge ciphertext in first half):* $\mathrm{CT}^*_A \neq \mathrm{CT}_A$.

where $\mathrm{CT}^* := (\mathrm{CT}^*_A, \mathrm{CT}^*_B)$ *is the challenge ciphertext and* CT^*_A *is an encryption of* CT^*_{in}. *We note that this event is well defined in both the cases where* $z = 0$ *and* $z = 1$.

4.1 Proof that Bad Query Event Does Not Happen

Claim 4.3 (No Bad Query Event when $z = 1$ (all zeros encrypted)) *Suppose that* Π_{dcca} *is DCCA secure,* $\Pi_{\mathrm{1b-cca}}$ *is 1-bounded CCA secure, and* Π_{cpa} *is CPA secure, all with perfect correctness. Then for all probabilistic polynomial-time adversaries* $\mathcal{A}$, *during a run of experiment* $\mathsf{Exp}^{\mathsf{nested}}_{\mathcal{A},\Pi_{\mathrm{dcca}},\Pi_{\mathrm{1b-cca}},\Pi_{\mathrm{cpa}}}(\lambda)$ *with* $z = 1$, *a bad query event does not take place except with negligible probability in* λ *where the probability is taken over the coins of the adversary and the experiment.*

Proof. We proceed via a series of hybrids. Let BQE denote a bad query event.

Step 1: Pr[BQE in Nested] $\sim$ Pr[BQE in Right-Erased] from CPA-security of Π_{cpa}. We first define a variation of the nested indistinguishability experiment with $z = 1$, which we call the *right-erased* experiment. In this experiment, CT_B^* is formed as $\text{CT}_B^* := \mathsf{Enc}_{\text{cpa}}(\text{PK}_B, 1^k; r_B)$ where k denotes the length of CT_{in}^*. CT_A^* is formed the same as in the nested indistinguishability experiment with $z = 1$. We suppose there exists a PPT adversary $\mathcal{A}$ for the nested indistinguishability experiment which causes the bad query event to occur with non-negligibly different probability in the usual experiment with $z = 1$ compared to the right-erased experiment. We construct a PPT algorithm $\mathcal{B}$ which violates the CPA-security of Π_{cpa}.

$\mathcal{B}$ is given PK_B. $\mathcal{B}$ then runs $\mathsf{KeyGen}_{\text{dcca}}$ and $\mathsf{KeyGen}_{\text{1b-cca}}$ for itself to produce $\text{PK}_{\text{in}}, \text{SK}_{\text{in}}$ and PK_A, SK_A respectively. It gives $\mathcal{A}$ $pk = (\text{PK}_{\text{in}}, \text{PK}_A, \text{PK}_B)$. $\mathcal{B}$ can simulate the decryption oracle $\mathsf{Dec}(sk, \cdot)$ for $\mathcal{A}$ by running the usual decryption algorithm (note that this does not require SK_B).

The adversary $\mathcal{A}$ outputs a pair of messages m_0, m_1 of the same length in the message space associated with pk. $\mathcal{B}$ chooses $r_A \in \{0,1\}^\lambda$ and computes $\text{CT}_{\text{in}}^* = \mathsf{Enc}_{\text{dcca}}(\text{PK}_{\text{in}}, 0^\ell)$, where ℓ is the length of the encoding of (r_A, r_A, m_0). It then computes $\text{CT}_A^* = \mathsf{Enc}_{\text{1b-cca}}(\text{PK}_A, \text{CT}_{\text{in}}^*; r_A)$. It submits CT_{in}^* and 1^k to its challenger as its two messages. It receives CT_B^* as the ciphertext. It gives $\text{CT}^* := (\text{CT}_A^*, \text{CT}_B^*)$ to $\mathcal{A}$.

To respond to remaining decryption queries $\mathcal{A}$ makes, $\mathcal{B}$ runs the usual decryption algorithm (after checking that the query is not equal to the challenge ciphertext). In addition, $\mathcal{B}$ checks for the bad query event by first checking if $\text{CT}_A \neq \text{CT}_A^*$ and then computing $F(\text{PK}_{\text{in}}, \text{CT}_{\text{in}}^*, \mathsf{Dec}_{\text{1b-cca}}(\text{SK}_A, \text{CT}_A))$. We recall that $\mathcal{B}$ generated SK_A, PK_A for itself, so it can compute $\mathsf{Dec}_{\text{1b-cca}}(\text{SK}_A, \text{CT}_A)$.

If CT_B^* is an encryption of CT_{in}^*, then $\mathcal{B}$ has properly simulated the usual experiment with $z = 1$. If it is instead an encryption of 1^k, then $\mathcal{B}$ has properly simulated the right-erased experiment. We note that the bad query event occurs in the simulation if and only if it is detected by $\mathcal{B}$.

We let ϵ denote the probability that the bad query event occurs in the usual experiment with $z = 1$ and δ denote this probability in the right-erased experiment. We suppose $\epsilon - \delta$ is positive and non-negligible (the opposite case is analogous). Now, if $\mathcal{B}$ detects the bad query event, it guesses that CT_B^* is an encryption of CT_{in}^*. Otherwise, it guesses the opposite. $\mathcal{B}$'s probability of guessing correctly in the CPA security game for Π_{cpa} is then equal to $\frac{\epsilon}{2} + \frac{1}{2}(1 - \delta) = \frac{1}{2} + \frac{1}{2}(\epsilon - \delta)$. The quantity $\epsilon - \delta$ is non-negligible, so $\mathcal{B}$ violates the CPA-security of Π_{cpa}. Hence we may conclude that the probability of the bad query event happening in the usual experiment with $z = 1$ is the same (up to a negligible difference) as the probability of the bad query event happening in the right-erased experiment for any PPT adversary.

Step 2: Pr[BQE in Full-Erased] is negligible from the unpredictability of the detecting function of Π_{dcca}. We now define an additional variation of the experiment, which we call the *full-erased* experiment. This is like the right-erased experiment, except that CT_A^* is also an encryption of 1^k, instead of an encryption of CT_{in}^*. We claim that in the full-erased experiment, the bad query event can only occur with negligible probability. To see this, we suppose we have a PPT adversary $\mathcal{A}$ which causes the bad query event to occur with non-negligible probability in the full-erased experiment. We will build a PPT adversary $\mathcal{B}$ for the basic unpredictability experiment which violates unpredictability of the detecting function for Π_{dcca}.

$\mathcal{B}$ is given PK_{in} and access to a decryption oracle $\mathsf{Dec}(\text{SK}_{\text{in}}, \cdot)$. It runs $\mathsf{KeyGen}_{\text{1b-cca}}$ and $\mathsf{KeyGen}_{\text{cpa}}$ for itself to produce PK_A, SK_A and PK_B, SK_B. It gives $(\text{PK}_{\text{in}}, \text{PK}_A, \text{PK}_B)$ to $\mathcal{A}$. $\mathcal{B}$ can simulate the decryption oracle for $\mathcal{A}$ using SK_A and its own decryption oracle. $\mathcal{A}$ outputs m_0, m_1. $\mathcal{B}$ then computes $\text{CT}_A^* = \mathsf{Enc}_{\text{1b-cca}}(\text{PK}_A, 1^k)$ and $\text{CT}_B^* = \mathsf{Enc}_{\text{cpa}}(\text{PK}_B, 1^k)$ and gives

$\mathrm{CT}^* = (\mathrm{CT}^*_A, \mathrm{CT}^*_B)$ to $\mathcal{A}$. We let q denote the number of Phase 2 queries made by $\mathcal{A}$. $\mathcal{B}$ can respond to these queries as before. $\mathcal{B}$ chooses a random $i \in \{1, 2, \ldots, q\}$. It takes the i^{th} Phase 2 query of $\mathcal{A}$, denoted by $(\mathrm{CT}^i_A, \mathrm{CT}^i_B)$, and computes $\mathrm{CT}^i_{\mathrm{in}} = \mathsf{Dec}_{\mathrm{1b-cca}}(\mathrm{SK}_A, \mathrm{CT}^i_A)$. It submits 0^ℓ and $\mathrm{CT}^i_{\mathrm{in}}$ to its challenger. Then, the distribution of $c^* = \mathsf{Enc}_{\mathrm{dcca}}(\mathrm{PK}_{\mathrm{in}}, 0^\ell)$ in the basic unpredictability experiment is precisely the distribution of $\mathrm{CT}^*_{\mathrm{in}}$. Hence, the bad query event for query i corresponds to an output of 1 for basic unpredictability experiment. Thus, if the bad query event occurs with some non-negligible probability ϵ, $\mathcal{B}$ will cause an output of 1 in the basic unpredictability experiment with probability at least $\frac{\epsilon}{q}$, which is non-negligible.

Step 3: Pr[BQE in Right-Erased] $\sim$ Pr[BQE in Full-Erased] from the 1-bounded CCA security of $\Pi_{\mathrm{1b-cca}}$. We now return to considering a PPT adversary $\mathcal{A}$ in the right-erased experiment. We let q denote the number of Phase 2 queries made by $\mathcal{A}$. We suppose that $\mathcal{A}$ causes the bad query event with non-negligible probability. Then there exists some index $i \in \{1, \ldots, q\}$ such that $\mathcal{A}$ causes the bad query event to occur with non-negligible probability on its i^{th} Phase 2 query. In other words, if there exists a PPT adversary $\mathcal{A}$ for which the bad query event occurs with non-negligible probability in the right-erased experiment, then for each value of the security parameter, there exists an index i such that $\mathcal{A}$ causes the BQE to occur on its i^{th} Phase 2 query with non-negligible probability. We note that for *any* i, the probability that $\mathcal{A}$ causes the BQE to occur on its i^{th} Phase 2 query in the full-erased experiment is negligible, as we proved above.

We fix such an i, and we define a PPT algorithm $\mathcal{B}$ which violates the 1-bounded CCA security of $\Pi_{\mathrm{1b-cca}}$. $\mathcal{B}$ receives PK_A from its challenger. It runs $\mathsf{KeyGen}_{\mathrm{dcca}}$ and $\mathsf{KeyGen}_{\mathrm{cpa}}$ for itself to produce $\mathrm{PK}_{\mathrm{in}}, \mathrm{SK}_{\mathrm{in}}$ and $\mathrm{PK}_B, \mathrm{SK}_B$. It gives $(\mathrm{PK}_{\mathrm{in}}, \mathrm{PK}_A, \mathrm{PK}_B)$ to $\mathcal{A}$ as the public key.

$\mathcal{B}$ simulates the decryption oracle for $\mathcal{A}$ as follows. Upon receiving a ciphertext $(\mathrm{CT}_A, \mathrm{CT}_B)$, $\mathcal{B}$ decrypts CT_B using $\mathsf{Dec}_{\mathrm{cpa}}$ with SK_B, and we let $\mathrm{CT}_{\mathrm{in}}$ denote the output. It then decrypts $\mathrm{CT}_{\mathrm{in}}$ using $\mathsf{Dec}_{\mathrm{dcca}}$ with $\mathrm{SK}_{\mathrm{in}}$, and parses the output as r_A, r_B, M. It checks if $\mathrm{CT}_A = \mathsf{Enc}_{\mathrm{1b-cca}}(\mathrm{PK}_A, \mathrm{CT}_{\mathrm{in}}; r_A)$ and if $\mathrm{CT}_B = \mathsf{Enc}_{\mathrm{cpa}}(\mathrm{PK}_B, \mathrm{CT}_{\mathrm{in}}; r_B)$. If both checks pass, it outputs M. Else, it outputs $\perp$.

We claim that this matches the output of the usual decryption algorithm, even though $\mathcal{B}$ is first decrypting CT_B instead of CT_A. To see this, note that the outputs are clearly the same whenever $\mathsf{Dec}_{\mathrm{1b-cca}}(\mathrm{CT}_A, \mathrm{SK}_A) = \mathsf{Dec}_{\mathrm{cpa}}(\mathrm{CT}_B, \mathrm{SK}_B)$. Whenever these are unequal, both decryption methods will output $\perp$. This is because $\mathrm{CT}_A = \mathsf{Enc}_{\mathrm{1b-cca}}(\mathrm{PK}_A, \mathrm{CT}_{\mathrm{in}}; r_A)$ and $\mathrm{CT}_B = \mathsf{Enc}_{\mathrm{cpa}}(\mathrm{PK}_B, \mathrm{CT}_{\mathrm{in}}; r_B)$ imply that $\mathsf{Dec}_{\mathrm{1b-cca}}(\mathrm{CT}_A, \mathrm{SK}_A) = \mathrm{CT}_{\mathrm{in}} = \mathsf{Dec}_{\mathrm{cpa}}(\mathrm{CT}_B, \mathrm{SK}_B)$. (Recall here that we have assumed $\Pi_{\mathrm{1b-cca}}$ and Π_{cpa} have perfect correctness.)

At some point, $\mathcal{A}$ outputs m_0, m_1. $\mathcal{B}$ forms $\mathrm{CT}^*_{\mathrm{in}} = \mathsf{Enc}_{\mathrm{dcca}}(\mathrm{PK}_{\mathrm{in}}, 0^\ell)$ and $\mathrm{CT}^*_B = \mathsf{Enc}_{\mathrm{cpa}}(\mathrm{PK}_B, 1^k)$. It outputs the messages $\mathrm{CT}^*_{\mathrm{in}}$ and 1^k to its challenger, and receives a ciphertext which it sets as CT^*_A. It gives the ciphertext $(\mathrm{CT}^*_A, \mathrm{CT}^*_B)$ to $\mathcal{A}$. It can then respond to $\mathcal{A}$'s Phase 2 decryption queries in the same way as before. When it receives the i^{th} Phase 2 query of $\mathcal{A}$, denoted by $(\mathrm{CT}^i_A, \mathrm{CT}^i_B)$, $\mathcal{B}$ checks for the bad query event by first checking if $\mathrm{CT}^i_A \neq \mathrm{CT}^*_A$ and if so, submitting CT^i_A as its one decryption query to its decryption oracle for PK_A. It can compute $F(\mathrm{PK}_{\mathrm{in}}, \mathrm{CT}^*_{\mathrm{in}}, \mathsf{Dec}(\mathrm{SK}_A, \mathrm{CT}^i_A))$. This equals 1 if and only if the bad query event has occurred for query i, and in this case $\mathcal{B}$ guesses that CT^*_A is an encryption of $\mathrm{CT}^*_{\mathrm{in}}$. Otherwise, $\mathcal{B}$ guesses the opposite.

We observe that when CT^*_A is an encryption of $\mathrm{CT}^*_{\mathrm{in}}$, then $\mathcal{B}$ has properly simulated the right-erased experiment, and when CT^*_A is an encryption of 0^k, then $\mathcal{B}$ has properly simulated the full-erased experiment. We let ϵ denote the non-negligible probability that $\mathcal{A}$ causes the bad query event to occur on (Phase 2) query i in the right-erased experiment, and we let δ denote the corresponding probability for the full-erased experiment. We know that δ must be negligible, therefore $\epsilon - \delta$ is

positive and non-negligible. The probability that $\mathcal{B}$ guesses correctly is: $\frac{1}{2}(1-\delta)+\frac{1}{2}\epsilon = \frac{1}{2}+\frac{1}{2}(\epsilon-\delta)$, so $\mathcal{B}$ achieves a non-negligible advantage in the 1-bounded CCA security game for $\Pi_{\text{1b-cca}}$.

Thus, it must be the case that for all PPT algorithms $\mathcal{A}$, the BQE occurs with only negligible probability in the right-erased experiment, and hence also in the nested experiment with $z = 1$. □

Claim 4.4 (No Bad Query Event when $z = 0$ (real message encrypted)) *As a consequence of Claim 4.3 and the DCCA security of Π_{dcca}, it holds that for all probabilistic polynomial-time adversaries $\mathcal{A}$, during a run of experiment $\mathsf{Exp}^{\mathsf{nested}}_{\mathcal{A},\Pi_{\text{dcca}},\Pi_{\text{1b-cca}},\Pi_{\text{cpa}}}(\lambda)$ with $z = 0$, a bad query event does not take place except with negligible probability in λ where the probability is taken over the coins of the adversary and the experiment.*

Proof. In Claim 4.3, we established that bad query events happen with at most negligible probability when $z = 1$. We will use this fact to argue that they cannot happen much more frequently when $z = 0$. Suppose to the contrary that there exists a PPT adversary $\mathcal{A}$ that forces bad query events to happen with non-negligible probability ϵ when $z = 0$. We create an PPT adversary $\mathcal{B}$ who interacts with $\mathcal{A}$ in a run of the nested indistinguishability experiment to break the DCCA security of Π_{dcca} with detecting function F with probability negligibly-close to $\frac{1}{2} + \frac{\epsilon}{2}$ as follows:

1. Setup: $\mathcal{B}$ obtains PK_{in} from the $\mathsf{Exp}^{\mathsf{indist}}$ challenger. It runs $\mathsf{KeyGen}_{\text{1b-cca}}$ to obtain $(\text{PK}_A, \text{SK}_A)$ and $\mathsf{KeyGen}_{\text{cpa}}$ to obtain $(\text{PK}_B, \text{SK}_B)$.

2. Phase 1: $\mathcal{B}$ gives to $\mathcal{A}$ the public key $\text{PK} = (\text{PK}_{\text{in}}, \text{PK}_A, \text{PK}_B)$. When $\mathcal{A}$ queries the decryption oracle on CT, $\mathcal{B}$ can simulate the normal decryption algorithm using SK_A and the phase 1 oracle $\mathsf{Dec}(\text{SK}_{\text{in}}, \cdot)$. Eventually, $\mathcal{A}$ outputs a pair of messages m_0, m_1.

3. Challenge: Choose random $\beta \in \{0,1\}$ and $r_A, r_B \in \{0,1\}^\lambda$. Send to the $\mathsf{Exp}^{\mathsf{indist}}$ challenger the messages $M_0 = (r_A, r_B, m_\beta)$ and $M_1 = 0^{|M_0|}$, and obtain from this challenger the ciphertext CT^*_{in}. Compute $\text{CT}^*_A := \mathsf{Enc}_{\text{1b-cca}}(\text{PK}_A, \text{CT}^*_{\text{in}}; r_A)$ and $\text{CT}^*_B := \mathsf{Enc}_{\text{cpa}}(\text{PK}_B, \text{CT}^*_{\text{in}}; r_B)$. Return $\text{CT}^* := (\text{CT}^*_A, \text{CT}^*_B)$ to $\mathcal{A}$.

4. Phase 2: When $\mathcal{A}$ queries the decryption oracle on $\text{CT} := (\text{CT}_A, \text{CT}_B)$, compute $\text{CT}_{\text{in}} := \mathsf{Dec}_{\text{1b-cca}}(\text{SK}_A, \text{CT}_A)$. If

   (a) Case 1 (a bad query event): $\text{CT}_A \neq \text{CT}^*_A$ and yet $F(\text{PK}_{\text{in}}, \text{CT}^*_{\text{in}}, \text{CT}_{\text{in}}) = 1$, then abort and output the bit 0.
   (b) Case 2 (partial match with challenge): $\text{CT}_A = \text{CT}^*_A$, then return $\perp$ to $\mathcal{A}$.

   Otherwise, query the phase 2 oracle, $\mathsf{Dec}(\text{SK}_{\text{in}}, \cdot)$, to decrypt CT_{in}, and return its response to $\mathcal{A}$.

5. Output: When $\mathcal{A}$ outputs a bit, $\mathcal{B}$ echos the bit as its output.

Analysis. We begin our analysis by arguing that $\mathcal{B}$ correctly answers all decryption queries except when it aborts. First, we show that a partial match with the challenge, causing the $\perp$ response in Case 2, is correct because that query must be invalid. Since a decryption query on the challenge is forbidden by the experiment, if $\text{CT}_A = \text{CT}^*_A$, then $\text{CT}_B \neq \text{CT}^*_B$. However, we argue that this must be an invalid ciphertext, i.e., one on which the main construction's decryption algorithm would return $\perp$. We see this as follows. Since decryption is deterministic, we have

$T := \mathsf{Dec}_{\mathrm{1b-cca}}(\mathrm{SK}_A, \mathrm{CT}_A) = \mathsf{Dec}_{\mathrm{1b-cca}}(\mathrm{SK}_A, \mathrm{CT}^*_A)$ and $(r_A, r_B, m) := \mathsf{Dec}_{\mathrm{dcca}}(\mathrm{SK}_{\mathrm{in}}, T)$. By the checks enforced by the main construction's decryption algorithm, there is only one "second half" that matches $\mathrm{CT}_A = \mathrm{CT}^*_A$, that is $\mathsf{Enc}_{\mathrm{cpa}}(\mathrm{PK}_B, T; r_B)$. Since the challenge is a valid ciphertext, CT^*_B must be this value and CT_B must cause an error.

When neither Case 1 or Case 2 applies in phase 2, the inner decryption query will succeed since the ciphertext is not detectably related to the challenge. This allows $\mathcal{B}$ to respond correctly.

When a bad query event occurs in Phase 2, $\mathcal{B}$ cannot query $\mathsf{Exp}^{\mathsf{indist}}$'s decryption oracle to decrypt the ciphertext. At first glance, one seems stuck. However, we assumed bad query events happen only when $z = 0$ with all but negligible probability. Thus, $\mathcal{B}$ can guess that $\mathcal{A}$ thinks $z = 0$, which corresponds to M_0 being encrypted in our reduction. Thus, $\mathcal{B}$ can abort and guess 0 at this point.

When $\mathcal{B}$ aborts, it causes the $\mathsf{Exp}^{\mathsf{indist}}$ experiment to output 1 with high probability. When $\mathcal{B}$ does not abort, it causes $\mathsf{Exp}^{\mathsf{indist}}$ experiment to output 1 with probability $\frac{1}{2}$. Since $\mathcal{B}$ aborts with non-negligible probability ϵ when $z = 0$, then $\mathcal{B}$ causes the experiment's output to be 1 with probability non-negligibly greater than $\frac{1}{2}$. □

4.2 Putting the Proof of the Main Theorem Together

Theorem 4.5 (Main Construction is Nested Indistinguishable) *Our main construction in Section 3, comprised of the three building blocks* $\Pi_{\mathrm{dcca}}, \Pi_{\mathrm{1b-cca}}, \Pi_{\mathrm{cpa}}$*, has nested indistinguishable encryptions under a chosen-ciphertext attack under the assumptions that* Π_{dcca} *is DCCA secure,* $\Pi_{\mathrm{1b-cca}}$ *is 1-bounded CCA secure, and* Π_{cpa} *is CPA secure, all with perfect correctness.*

Proof of Theorem 4.5 is given in Appendix C. The crux of the argument is that bad query events do not happen (except with negligible probability). This was already established in Claims 4.3 and 4.4. Armed with this fact, we can prove the nested indistinguishability of the main construction based on the indistinguishability property of the DCCA-security of Π_{dcca}. The reduction and its analysis are similar to those in the proof of Claim 4.4. [5]

The following corollary follows from Theorem 4.5. Informally, if the adversary cannot distinguish an encryption of a message from an encryption of zeros, then she also cannot distinguish between the encryptions of two different messages.

Corollary 4.6 (Main Construction is CCA2 Secure) *Our main construction in Section 3, comprised of the three building blocks* $\Pi_{\mathrm{dcca}}, \Pi_{\mathrm{1b-cca}}, \Pi_{\mathrm{cpa}}$*, is CCA2 secure under the assumptions that* Π_{dcca} *is DCCA secure,* $\Pi_{\mathrm{1b-cca}}$ *is 1-bounded CCA secure, and* Π_{cpa} *is CPA secure, all with perfect correctness.*

5 Why not use CCA1?

We now consider using an arbitrary CCA1-secure scheme in place of the DCCA-secure scheme in our construction in Section 3. To give intuition about why we *believe* this approach fails in general

[5] We note that we alternatively could have merged the proofs of Claim 4.4 and Theorem 4.5. However, we chose to keep the bad event analysis separate for pedagogical purposes at the expense of some redundancy in the description of the related reductions.

to provide a CCA2-secure scheme, we define the following oracle. This oracle enables a CCA2 attack on the construction, without *appearing* to break the CCA1 security of the inner scheme or the 1-bounded CCA security/CPA security of the outer schemes.

Oracle The oracle takes in the public key (consisting of the three public keys for the three building blocks) and a ciphertext CT. It runs the decryption algorithm of the construction on CT. If the decryption algorithm outputs a message M, then the oracle runs the encryption algorithm to produce a new ciphertext $\widetilde{\text{CT}}$ encrypting M. If the decryption algorithm outputs $\perp$ (indicating that the ciphertext was malformed), the oracle encrypts a string of 0's of the appropriate length to produce the new ciphertext $\widetilde{\text{CT}}$. (The length of the 0 string is chosen so that the inner ciphertext has the same size as CT_{in} for CT.) The oracle outputs $\widetilde{\text{CT}}$.

A Chosen Ciphertext Attack on the System. Using the oracle, an attacker can violate the CCA2 security of the construction as follows. Upon receiving the challenge ciphertext CT^*, the attacker sends this to the oracle to obtain a ciphertext $\widetilde{\text{CT}}^*$ encrypting the same message. Since $\widetilde{\text{CT}}^* \neq \text{CT}^*$, it can query $\widetilde{\text{CT}}^*$ to its decryption oracle in the CCA2 security game. It receives the message, thereby violating security.

Why Security of the Underlying Primitives Remains. Intuitively, the oracle should only be useful to an attacker who still has access to the decryption oracle in the security game. This does not violate CCA1 security of the inner scheme, since in the CCA1 security game the attacker loses access to the decryption oracle completely after receiving the challenge ciphertext. It is important to note here that the oracle's output does not give away whether the ciphertext it received was malformed. The oracle also does not break the 1-bounded CCA and CPA security guarantees of the outer encryption schemes, because even though the attacker may use its one query for the 1-bounded CCA-secure scheme after seeing the challenge ciphertext, it will not know the randomness used in the challenge encryption, and hence cannot create from it a well-formed ciphertext. Since the oracle runs the usual decryption algorithm which checks that the ciphertext is well-formed, it will not be useful to the attacker attempting to break security of the outer schemes.

Open Questions. This oracle is quite strong, and this leaves some remaining questions. First, might there be a useful notion between CCA1 security and DCCA security for the inner building block that would suffice for the outer scheme to imply CCA2 security for our construction? Also, it would be interesting to construct a more concrete counterexample to CCA2 security for our construction with a CCA1-secure inner scheme.

Acknowledgments

The authors thank Steven Myers and the anonymous reviewers for helpful comments.

References

[1] Jee Hea An, Yevgeniy Dodis, and Tal Rabin. On the security of joint signature and encryption. In *EUROCRYPT*, volume 2332, pages 83–107, 2002.

[2] Mihir Bellare, Zvika Brakerski, Moni Naor, Thomas Ristenpart, Gil Segev, Hovav Shacham, and Scott Yilek. Hedged public-key encryption: How to protect against bad randomness. In *ASIACRYPT*, pages 232–249, 2009.

[3] Mihir Bellare and Phillip Rogaway. Random oracles are practical: A paradigm for designing efficient protocols. In *ACM Conference on Computer and Communications Security*, pages 62–73, 1993.

[4] Mihir Bellare and Amit Sahai. Non-malleable encryption: Equivalence between two notions, and an indistinguishability-based characterization. In *CRYPTO*, pages 519–536, 1999.

[5] Manuel Blum, Paul Feldman, and Silvio Micali. Non-interactive zero-knowledge and its applications (extended abstract). In *STOC*, pages 103–112, 1988.

[6] Dan Boneh, Ran Canetti, Shai Halevi, and Jonathan Katz. Chosen-ciphertext security from identity-based encryption. *SIAM J. Comput.*, 36(5):1301–1328, 2007.

[7] Ran Canetti, Shai Halevi, and Jonathan Katz. Chosen-ciphertext security from identity-based encryption. In *EUROCRYPT*, pages 207–222, 2004.

[8] Ran Canetti, Hugo Krawczyk, and Jesper Buus Nielsen. Relaxing chosen-ciphertext security. In *CRYPTO*, volume 2729, pages 565–582, 2003.

[9] David Cash, Eike Kiltz, and Victor Shoup. The twin diffie-hellman problem and applications. In *EUROCRYPT*, pages 127–145, 2008.

[10] Seung Geol Choi, Dana Dachman-Soled, Tal Malkin, and Hoeteck Wee. Black-box construction of a non-malleable encryption scheme from any semantically secure one. In *TCC*, pages 427–444, 2008.

[11] Ronald Cramer, Goichiro Hanaoka, Dennis Hofheinz, Hideki Imai, Eike Kiltz, Rafael Pass, Abhi Shelat, and Vinod Vaikuntanathan. Bounded cca2-secure encryption. In *ASIACRYPT*, pages 502–518, 2007.

[12] Ronald Cramer and Victor Shoup. A practical public key cryptosystem provably secure against adaptive chosen ciphertext attack. In *CRYPTO*, pages 13–25, 1998.

[13] Ronald Cramer and Victor Shoup. Universal hash proofs and a paradigm for adaptive chosen ciphertext secure public-key encryption. In *EUROCRYPT*, pages 45–64, 2002.

[14] Danny Dolev, Cynthia Dwork, and Moni Naor. Non-malleable cryptography (extended abstract). In *STOC*, pages 542–552, 1991.

[15] Yael Gertner, Tal Malkin, and Omer Reingold. On the impossibility of basing trapdoor functions on trapdoor predicates. In *FOCS*, pages 126–135, 2001.

[16] Shafi Goldwasser and Silvio Micali. Probabilistic encryption. *J. Comput. Syst. Sci.*, 28(2):270–299, 1984.

[17] Dennis Hofheinz and Eike Kiltz. Secure hybrid encryption from weakened key encapsulation. In *CRYPTO*, volume 4622, pages 553–571, 2007.

[18] Dennis Hofheinz and Eike Kiltz. Practical chosen ciphertext secure encryption from factoring. In *EUROCRYPT*, pages 313–332, 2009.

[19] Jonathan Katz and Yehuda Lindell. *Introduction to Modern Cryptography*. Chapman & Hall, CRC, 2007.

[20] Eike Kiltz. Chosen-ciphertext security from tag-based encryption. In *TCC*, 2006.

[21] Eike Kiltz, Payman Mohassel, and Adam O'Neill. Adaptive trapdoor functions and chosen-ciphertext security. In *EUROCRYPT*, pages 673–692, 2010.

[22] Philip D. MacKenzie, Michael K. Reiter, and Ke Yang. Alternatives to non-malleability: Definitions, constructions, and applications (extended abstract). In *TCC*, pages 171–190, 2004.

[23] Steven Myers and Abhi Shelat. Bit encryption is complete. In *FOCS*, pages 607–616, 2009.

[24] Moni Naor and Moti Yung. Public-key cryptosystems provably secure against chosen ciphertext attacks. In *STOC*, pages 427–437, 1990.

[25] Rafael Pass, Abhi Shelat, and Vinod Vaikuntanathan. Construction of a non-malleable encryption scheme from any semantically secure one. In *CRYPTO*, pages 271–289, 2006.

[26] Chris Peikert and Brent Waters. Lossy trapdoor functions and their applications. In *STOC*, pages 187–196, 2008.

[27] Charles Rackoff and Daniel R. Simon. Non-interactive zero-knowledge proof of knowledge and chosen ciphertext attack. In *CRYPTO*, pages 433–444, 1991.

[28] Alon Rosen and Gil Segev. Chosen-ciphertext security via correlated products. In *TCC*, pages 419–436, 2009.

[29] Amit Sahai. Non-malleable non-interactive zero knowledge and adaptive chosen-ciphertext security. In *FOCS*, pages 543–553, 1999.

[30] Victor Shoup. A proposal for an iso standard for public key encryption. Cryptology ePrint Archive, Report 2001/112, 2001. `http://eprint.iacr.org/`.

A Background on Security Definitions for Encryption

There is a rich body of literature on how to formalize the confidentiality guarantee of encryption as we discussed in Section 1.1. We define several of these concepts here.

First, we recall the algorithms comprising an encryption system.

Definition A.1 (Encryption System) *An* encryption system *is a tuple of probabilistic polynomial-time algorithms* $(\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$ *such that:*

1. $\mathsf{KeyGen}(1^\lambda) \to (pk, sk)$*: the key generation algorithm takes as input the security parameter* 1^λ *and outputs a pair of keys* (pk, sk)*.*

2. $\mathsf{Enc}(pk, m) \to c$: *the encryption algorithm takes as input a public key* pk *and a message* m *from some underlying plaintext space and outputs a ciphertext* c. Enc *is a probabilistic algorithm, although we will sometimes cast it as a deterministic algorithm with an explicit random input,* r*, by writing*

$$c := \mathsf{Enc}(pk, m; r).$$

3. $\mathsf{Dec}(sk, c) \to m$: *the decryption algorithm takes as input a secret key* sk *and a ciphertext* c*, and outputs a message* m *or a special symbol* $\perp$ *denoting failure. Wlog, we assume that this algorithm is deterministic and write* $m := \mathsf{Dec}(sk, c)$.

For the system to be correct*, we require that* $\mathsf{Dec}(sk, \mathsf{Enc}(pk, m)) = m$*, except with negligible probability over* (pk, sk) *output by* $\mathsf{KeyGen}(1^\lambda)$ *and any randomness used by* Enc.

CCA Security Experiment. We now recall the definition of CCA Security. Consider the following experiment $\mathsf{Exp}^{\mathsf{cca}}_{\mathcal{A},\Pi}(\lambda)$ defined for public-key encryption scheme $\Pi = (\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$ and an adversary $\mathcal{A}$:

1. Setup: $\mathsf{KeyGen}(1^\lambda)$ is run to obtain keys (pk, sk).
2. Phase 1: Adversary $\mathcal{A}$ is given pk and access to a decryption oracle $\mathsf{Dec}(sk, \cdot)$. The adversary outputs a pair of messages m_0, m_1 of the same length in the message space associated with pk.
3. Challenge: A random bit $b \leftarrow \{0, 1\}$ is chosen, and then a ciphertext $c^* \leftarrow \mathsf{Enc}(pk, m_b)$ is computed and given to $\mathcal{A}$. We call c^* the challenge ciphertext.
4. Phase 2: $\mathcal{A}$ continues to have access to $\mathsf{Dec}(sk, \cdot)$ provided he does not request a decryption of c^*. Finally, $\mathcal{A}$ outputs a bit b'.
5. Output: The output of the experiment is defined to be 1 if $b' = b$, and 0 otherwise.

Definition A.2 (CCA Security [27]) *A public-key encryption scheme* $\Pi = (\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$ *has* indistinguishable encryptions under a chosen-ciphertext attack *(or is* CCA-secure*) if for all probabilistic polynomial-time adversaries* $\mathcal{A}$ *there exists a negligible function* $negl$ *such that:*

$$\Pr[\mathsf{Exp}^{\mathsf{cca}}_{\mathcal{A},\Pi}(\lambda) = 1] \leq \frac{1}{2} + negl(\lambda).$$

We also consider the following variants of the above definition:

1. **Chosen Plaintext Attack (CPA) Security [16]:** same as above, except that $\mathcal{A}$ is not given access to the decryption oracle in phase 1 or phase 2.
2. **Lunchtime or CCA1 Security [24]:** same as above, except that $\mathcal{A}$ is not given access to the decryption oracle in phase 2.
3. q**-Bounded CCA Security [11]:** same as above, except that the total number of decryption queries made by $\mathcal{A}$ in phase 1 and phase 2 is at most q. In this work, we will use 1-bounded CCA (a.k.a., "one-time CCA") security, where $\mathcal{A}$ can make a single decryption query.

Realizations We note that CPA security implies 1-bounded CCA security [25, 11, 10]. Actually, the above works allow for a stronger notion of one parallel query of many ciphertexts. This is related to the notion of non-malleability [14, 4].

B Plaintext, Randomness and Ciphertext Spaces

For this paper, we assume that detectable schemes allow arbitrary-length messages and that the encryption randomness will always be of the length of the security parameter and therefore independent of the message length. We also assume that the ciphertext size is a deterministic function of the security parameter and the message length.

This will be useful for a property of our systems where we will implicitly encrypt our own randomness by having it nested inside of another ciphertext. Having short randomness is actually more important for the one-time CCA-secure schemes used as a building block in our construction, but we argue generally that this is not a limiting assumption for encryption schemes here.

We justify our main assumptions with two lemmas.

First, we use Lemma 2.5, which asserts that one-bit DCCA-secure encryption implies arbitrary-length DCCA-secure encryption. Recall the construction. Say $\Pi = (\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec}, F)$ is a detectable encryption system with plaintext space $\{0,1\}$. We can construct a new scheme $\Pi' = (\mathsf{KeyGen}, \mathsf{Enc}', \mathsf{Dec}', F')$ with plaintext space $\{0,1\}^*$ by defining Enc' as follows:

$$\mathsf{Enc}'(pk, m) = \mathsf{Enc}(pk, m_1), \ldots, \mathsf{Enc}(pk, m_n)$$

where $m = m_1 \ldots m_n$. The decryption algorithm Dec' decrypts each ciphertext piece using Dec. The function F' performs n^2 invocations of F, testing each ciphertext piece of C with each ciphertext piece of C', and outputting 1 if any invocation of F returned 1, and 0 otherwise. We now prove Lemma 2.5.

Proof. Recall that our construction defined the function F' to perform n^2 invocations of F, testing each ciphertext piece of C with each ciphertext piece of C', and outputting 1 if any invocation of F returned 1, and 0 otherwise. Clearly F' is efficiently computable if F is.

Unpredictability of F'. We suppose there exists a PPT adversary $\mathcal{A}$ that causes the output of the basic unpredictability experiment with Π' to be 1 with non-negligible probability ϵ. We construct a PPT adversary $\mathcal{B}$ against the basic unpredictability experiment with Π. The experiment for $\mathcal{B}$ begins by a run of KeyGen producing pk, sk. $\mathcal{B}$ is given pk and access to a decryption oracle $\mathsf{Dec}(sk, \cdot)$. It gives pk to $\mathcal{A}$. To simulate a decryption oracle for $\mathcal{A}$, $\mathcal{B}$ takes a ciphertext query from $\mathcal{A}$ in the form $c_1, \ldots, c_n$ and separately queries each of $c_1, \ldots, c_n$ to its decryption oracle. It returns the ordered n-tuple of replies to $\mathcal{A}$.

When $\mathcal{A}$ outputs a message $m = m_1 \ldots m_k$ and a ciphertext $c_1, \ldots, c_\ell$, $\mathcal{B}$ chooses two random indices $i \in [k]$, $j \in [\ell]$ and outputs m_i, c_j. The experiment then proceeds to the challenge phase, computing $c_i^* \leftarrow \mathsf{Enc}(pk, m_i)$. This is distributed identically to the i^{th} piece of the challenge ciphertext that would be created in the experiment for $\mathcal{A}$. We let K and L be polynomial-size bounds such that $\mathcal{A}$ chooses $k \leq K$ and $\ell \leq L$ with all but negligible probability. Then the probability that the outcome of $\mathcal{B}$'s is 1 is negligibly close to $\frac{\epsilon}{KL}$.

To see this, we consider simulating the full experiment for $\mathcal{A}$ by also computing $c_{i'}^* \leftarrow \mathsf{Enc}(pk, m_{i'})$ for all $i' \neq i$. We note that when $\mathcal{A}$ succeeds, there must be some indices i^*, j^* such that

$F(pk, c_{j^*}, \mathsf{Enc}(pk, m_{i^*})) = 1$. Fixing all the randomness except for the choice of i, j by $\mathcal{B}$, $\mathcal{B}$ will now succeed in its experiment as long as it chooses $i = i^*$ and $j = j^*$: this occurs with probability $\frac{1}{kl}$, which is $\geq \frac{1}{KL}$ with all but negligible probability. We note that the choices of i, j by $\mathcal{B}$ are made independently of all other random choices occurring in this simulated experiment for $\mathcal{A}$. Hence, $\mathcal{B}$ causes the output of the basic unpredictability experiment with Π' to be 1 with non-negligible probability. We note that the same proof (with trivial adjustments) works for the strong unpredictability experiment.

Indistinguishability of Encryptions. It remains to show a PPT adversary must have a negligible advantage in the indistinguishability of encryptions experiment. We suppose there exists a PPT adversary $\mathcal{A}$ that causes the output of the indistinguishability of encryptions experiment with Π' to equal 1 with probability non-negligibly greater than $\frac{1}{2}$. We construct a PPT adversary $\mathcal{B}$ for the indistinguishability of encryptions experiment with Π. We employ a hybrid argument. We let k denote an upper bound of the length of the messages m_0, m_1 produced by $\mathcal{A}$ as its output for Phase I of the experiment. We then define experiments 1 through k as follows. Each experiment i is like the indistinguishability of encryptions experiment except for how the challenge ciphertext is created. In experiment i, the first $i-1$ pieces of the ciphertext are created by encrypting the first $i-1$ bits of m_0, the i^{th} piece is created by encrypting the i^{th} bit of m_b, and the remaining pieces are encryptions of the bits of m_1, starting from the $i+1$ bit. The output of each experiment is still defined to be 1 when $b = b'$.

We observe that there must exist some i^* such that $\mathcal{A}$ causes the output of experiment i^* with Π' to be 1 with probability non-negligibly greater than $\frac{1}{2}$. The experiment for $\mathcal{B}$ begins by a run of KeyGen producing pk, sk. $\mathcal{B}$ is given pk and access to a decryption oracle $\mathsf{Dec}(sk, \cdot)$. It forwards pk to $\mathcal{A}$. To simulate a decryption oracle for $\mathcal{A}$, $\mathcal{B}$ takes a ciphertext query from $\mathcal{A}$ in the form $c_1, \ldots, c_n$ and separately queries each of $c_1, \ldots, c_n$ to its decryption oracle. It returns the ordered n-tuple of replies to $\mathcal{A}$.

$\mathcal{A}$ produces two messages m_0, m_1 of the same length in the message space associated with pk. $\mathcal{B}$ produces the ciphertext c^* as follows. It encrypts the first $i^* - 1$ bits of m_0 to form the first $i^* - 1$ pieces of c^*, and submits the i^*th bits of m_0, m_1 as its messages to the challenger for the indistinguishability of encryptions experiment for Π. It uses the ciphertext it receives in return as the i^* piece of c^*. It forms the remaining pieces by encrypting the final bits of m_1, (starting from the $i^* + 1$ bit). It gives c^* to $\mathcal{A}$.

$\mathcal{A}$ may continue to make decryption queries, as long as none of the pieces of these queries are "related" to pieces of c^* in the sense defined by F (this is just a restatement of the definition for F'). This restriction allows $\mathcal{B}$ to simulate the decryption oracle on these queries as before, by submitting them separately to its own decryption oracle. Finally, $\mathcal{A}$ will output a bit b', which $\mathcal{B}$ copies as its own output. This will equal b with probability non-negligibly greater than $\frac{1}{2}$, since $\mathcal{A}$ accomplishes this in experiment i^*. □

Next, we rely on the common trick of replacing the randomness for encryption with the output of a pseudorandom generator. Thus the "real" randomness needed for encryption is just a seed of the length of the security parameter. Say $\Pi = (\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec}, F)$ is a detectable encryption system with randomness s of length $\ell = \ell(\lambda)$, where λ is the security parameter and ℓ is a polynomial. Let $G : \{0,1\}^\lambda \to \{0,1\}^{\ell(\lambda)}$ be a pseudorandom generator; if pseudorandom generators exist, then such a pseudorandom generator must exist [19, p. 76]. We can construct a new scheme

$\Pi' = (\mathsf{KeyGen}', \mathsf{Enc}', \mathsf{Dec}, F)$ with randomness s' of length λ. The first step of KeyGen' and Enc' is to run $t := G(s')$. Then they operate exactly as KeyGen and Enc using the appropriate bits of t.

Lemma B.1 (Short Randomness is Sufficient) *Let* Π *and* Π' *be as above. If* Π *is DCCA-secure, then so is* Π'*.*

We omit the straightforward proof of the above lemma. It expands in the obvious way to CPA and CCA2-secure systems.

Finally, we assume that the ciphertext size can be considered a deterministic function of the security parameter and message length. Informally, the length of the ciphertext cannot be predictably related to the *message value*, because this would break the indistinguishability of encryptions property. It is possible that some systems might, for example, have variable-length ciphertexts by appending a variable-length random string to the end of the encryption. However, we focus our attention on schemes that do not allow this.

C Proof of Theorem 4.5 (Main Construction is Nested Indistinguishable)

Proof. Given that bad query events occur with only negligible probability, we use this fact to prove the nested indistinguishability of our main construction based on the indistinguishability property of the DCCA-security of Π_{dcca}.

The Reduction Algorithm. Let 1^λ be the security parameter. Suppose there exists a PPT adversary $\mathcal{A}$ that causes the output of the nested experiment with the main construction to output 1 with probability $\frac{1}{2}+\epsilon$. We construct a PPT adversary $\mathcal{B}$ against the indistinguishability experiment of the DCCA security of Π_{dcca} with detecting function F.

1. Setup: $\mathcal{B}$ obtains PK_{in} from the $\mathsf{Exp}^{\mathsf{indist}}$ challenger. It runs $\mathsf{KeyGen}_{1\text{b}-\text{cca}}$ to obtain $(\text{PK}_A, \text{SK}_A)$ and $\mathsf{KeyGen}_{\text{cpa}}$ to obtain $(\text{PK}_B, \text{SK}_B)$.

2. Phase 1: $\mathcal{B}$ gives to $\mathcal{A}$ the public key $\text{PK} = (\text{PK}_{\text{in}}, \text{PK}_A, \text{PK}_B)$. When $\mathcal{A}$ queries the decryption oracle on CT, $\mathcal{B}$ can simulate the normal decryption algorithm using SK_A and the phase 1 oracle $\mathsf{Dec}(\text{SK}_{\text{in}}, \cdot)$. Eventually, $\mathcal{A}$ outputs a pair of messages m_0, m_1.

3. Challenge: Choose random $\beta \in \{0,1\}$ and $r_A, r_B \in \{0,1\}^\lambda$. Send to the $\mathsf{Exp}^{\mathsf{indist}}$ challenger the messages $M_0 = (r_A, r_B, m_\beta)$ and $M_1 = 0^{|M_0|}$, and obtain from this challenger the ciphertext CT^*_{in}. Compute $\text{CT}^*_A := \mathsf{Enc}_{1\text{b}-\text{cca}}(\text{PK}_A, \text{CT}^*_{\text{in}}; r_A)$ and $\text{CT}^*_B := \mathsf{Enc}_{\text{cpa}}(\text{PK}_B, \text{CT}^*_{\text{in}}; r_B)$. Return $\text{CT}^* := (\text{CT}^*_A, \text{CT}^*_B)$ to $\mathcal{A}$.

4. Phase 2: When $\mathcal{A}$ queries the decryption oracle on $\text{CT} := (\text{CT}_A, \text{CT}_B)$, compute $\text{CT}_{\text{in}} := \mathsf{Dec}_{1\text{b}-\text{cca}}(\text{SK}_A, \text{CT}_A)$. If

   (a) Case 1 (a bad query event): $\text{CT}_A \neq \text{CT}^*_A$ and yet $F(\text{PK}_{\text{in}}, \text{CT}^*_{\text{in}}, \text{CT}_{\text{in}}) = 1$, then abort and take a random guess.
   (b) Case 2 (partial match with challenge): $\text{CT}_A = \text{CT}^*_A$, then return $\perp$ to $\mathcal{A}$.

   Otherwise, query the phase 2 oracle, $\mathsf{Dec}(\text{SK}_{\text{in}}, \cdot)$, to decrypt CT_{in}, and return its response to $\mathcal{A}$.

5. Output: When $\mathcal{A}$ outputs a bit, $\mathcal{B}$ echoes the bit as its output.

Analysis. We begin our analysis by arguing that $\mathcal{B}$ correctly answers all decryption queries except when it aborts. Through Claims 4.3 and 4.4, we have already established that a bad query event, causing the abort in Case 1, happens with only negligible probability assuming that Π_{dcca} is DCCA secure, $\Pi_{\text{1b-cca}}$ is 1-bounded CCA secure, and Π_{cpa} is CPA secure.

Next, we show that a partial match with the challenge, causing the $\perp$ response in Case 2, is correct because that query must be invalid. Since a decryption query on the challenge is forbidden by the experiment, if $\text{CT}_A = \text{CT}_A^*$, then $\text{CT}_B \neq \text{CT}_B^*$. However, we argue that this must be an invalid ciphertext, i.e., one on which the main construction's decryption algorithm would return $\perp$. We see this as follows. Since decryption is deterministic, we have $T := \mathsf{Dec}_{\text{1b-cca}}(\text{SK}_A, \text{CT}_A) = \mathsf{Dec}_{\text{1b-cca}}(\text{SK}_A, \text{CT}_A^*)$ and $(r_A, r_B, m) := \mathsf{Dec}_{\text{dcca}}(\text{SK}_{\text{in}}, T)$. By the checks enforced by the main construction's decryption algorithm, there is only one "second half" that matches $\text{CT}_A = \text{CT}_A^*$, that is $\mathsf{Enc}_{\text{cpa}}(\text{PK}_B, T; r_B)$. Since the challenge is a valid ciphertext, CT_B^* must be this value and CT_B must cause an error.

When neither Case 1 or Case 2 applies in phase 2, the inner decryption query will succeed since the ciphertext is not detectably related to the challenge. This allows $\mathcal{B}$ to respond correctly.

Finally, $\mathcal{B}$ causes all inputs to $\mathcal{A}$ to have the same distribution as the nested experiment. When $\mathcal{B}$ aborts, it causes the $\mathsf{Exp}^{\mathsf{indist}}$ experiment to output 1 with probability $\frac{1}{2}$. When $\mathcal{B}$ does not abort, all decryption queries are answered correctly and this causes the $\mathsf{Exp}^{\mathsf{indist}}$ experiment to output 1 with probability $\frac{1}{2} + \epsilon$. Since $\mathcal{B}$ does not abort with high probability, if ϵ is non-negligible, then $\mathcal{B}$ causes the experiment's output to be 1 with probability non-negligibly greater than $\frac{1}{2}$. □

D Detectable CCA from (Adaptive) Tag-Based Encryption

MacKenzie, Reiter and Yang [22] define a tag-based encryption scheme as an encryption scheme that takes in an additional "tag" parameter on encryption and decryption. We recall this definition.

Consider the following experiment $\mathsf{Exp}^{\mathsf{tbe-atag-cca}}_{\mathcal{A},\Pi}(\lambda)$ defined for a tag-based encryption scheme $\Pi = (\mathsf{TBKeyGen}, \mathsf{TBEnc}, \mathsf{TBDec})$ and an adversary $\mathcal{A}$:

1. Setup: $\mathsf{TBKeyGen}(1^\lambda)$ is run to obtain keys (pk, sk).
2. Phase 1: Adversary $\mathcal{A}$ is given pk and access to a decryption oracle $\mathsf{TBDec}(sk, \cdot, \cdot)$. $\mathcal{A}$ outputs a pair of messages m_0, m_1 of the same length in the message space associated with pk and a target tag t^* from the tag space.
3. Challenge: A random bit $b \leftarrow \{0, 1\}$ is chosen, and then a ciphertext $c^* \leftarrow \mathsf{TBEnc}(pk, t^*, m_b)$ is computed and given to $\mathcal{A}$. We call c^* the challenge ciphertext.
4. Phase 2: $\mathcal{A}$ continues to have access to $\mathsf{TBDec}(sk, \cdot, \cdot)$, but may not request a decryption of a ciphertext with tag t such that $t \neq t^*$. Finally, $\mathcal{A}$ outputs a bit b'.
5. Output: The output of the experiment is defined to be 1 if $b' = b$, and 0 otherwise.

Definition D.1 (Adaptive Tag-Based Security) *A tag-based encryption scheme* $\Pi = (\mathsf{TBKeyGen}, \mathsf{TBEnc}, \mathsf{TBDec})$ *has* indistinguishable encryptions under a tag-based chosen-ciphertext attack *if for all probabilistic polynomial-time adversaries* $\mathcal{A}$ *there exists a negligible function negl such that:*

$$\Pr[\mathsf{Exp}^{\mathsf{tbe-atag-cca}}_{\mathcal{A},\Pi}(\lambda) = 1] \leq \frac{1}{2} + \mathit{negl}(\lambda).$$

Kiltz [20] considered a *selective* variant of this definition where the target tag t^* used in the challenge ciphertext must be output by the adversary before the public key is created.

We now show that any tag-based scheme satisfying Definition D.1 gives rise to a DCCA-secure system. Let $\Pi = (\mathsf{TBKeyGen}, \mathsf{TBEnc}, \mathsf{TBDec})$ be a tag-based encryption system with tag space T. We create a detectable encryption system, $\Pi' = (\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec}, F)$, as:

1. $\mathsf{KeyGen}(1^\lambda)$: Run $\mathsf{TBKeyGen}(1^\lambda)$ and output the resulting key pair.
2. $\mathsf{Enc}(pk, m)$: Select a random tag $t \leftarrow T$ and compute $d \leftarrow \mathsf{TBEnc}(pk, t, m)$. Output the ciphertext $c := (t, d)$.
3. $\mathsf{Dec}(sk, c)$: Parse c as (t, d). Output the result of $\mathsf{TBDec}(sk, t, d)$.
4. $F(pk, c, c')$: Parse c as (t, d) and c' as (t', d'). Output 1 if $t = t'$ and 0 otherwise.

Lemma D.2 (DCCA from Tag-Based Encryption) *If Π is an adaptively-secure tag-based system according to Definition D.1 with an exponentially-large tag space T, then Π' is a DCCA-secure detectable system according to Definition 2.2.*

Proof. This proof involves two parts. First, we argue that the detecting function F is unpredictable. In the basic unpredictability game, the adversary must fix a tag $t \in T$ as part of the ciphertext c. After this is fixed, the challenger chooses a random tag $t^* \in T$ for the challenge ciphertext. The detecting function F outputs 1, causing the experiment to output 1, if and only if $t = t^*$. This happens with probability exactly $1/|T|$. Since we conditioned that T is exponentially-large in the security parameter, we can conclude that the adversary causes the basic unpredictability experiment to output 1 will only negligible probability.

Second, we argue that the encryptions of Π' are indistinguishable under a detectable chosen ciphertext attack. Suppose this is false and there exists a PPT adversary $\mathcal{A}$ with probability $1/2$ plus a non-negligible advantage ϵ, then we use this adversary to construct a PPT adversary $\mathcal{B}$ for the tag-based experiment as follows. $\mathcal{B}$ passes the public key pk to $\mathcal{A}$. When $\mathcal{A}$ makes a decryption query on ciphertext $c := (t, d)$, $\mathcal{B}$ passes this query to its decryption oracle with tag t and ciphertext d and returns the answer. When $\mathcal{A}$ outputs a pair of messages m_0, m_1, $\mathcal{B}$ chooses a random $t^* \in T$ and outputs (m_0, m_1, t^*). The tag-based challenger responds with a ciphertext $c^* := (t^*, d^*)$, which $\mathcal{B}$ passes to $\mathcal{A}$. When $\mathcal{A}$ makes a decryption query it must obey the detecting predicate F which is defined to forbid any ciphertext $c := (t, d)$ such that $t = t^*$, thus $\mathcal{B}$ will be able to pass the query on to its decryption oracle and return the answer. When $\mathcal{A}$ outputs a bit, $\mathcal{B}$ outputs the same bit. It is straightforward to see that $\mathcal{B}$ is able to simulate the DCCA game for $\mathcal{A}$ exactly and will succeed in the tag-based experiment with probability $1/2 + \epsilon$, thereby breaking the security of Π. □