Non-Malleable Encryption: Simpler, Shorter, Stronger*

Sandro Coretti[†]
IOHK
sandro.coretti@iohk.io

Yevgeniy Dodis[‡]
New York University
dodis@cs.nyu.edu

Ueli Maurer[§]
ETH Zurich
maurer@inf.ethz.ch

Björn Tackmann[¶]
DFINITY Foundation
bjoern@dfinity.org

Daniele Venturi
Sapienza University of Rome
venturi@di.uniroma1.it

Abstract

One approach towards basing public-key encryption (PKE) schemes on weak and credible assumptions is to build "stronger" or more general schemes generically from "weaker" or more restricted ones. One particular line of work in this context was initiated by Myers and shelat (FOCS '09) and continued by Hohenberger, Lewko, and Waters (Eurocrypt '12), who provide constructions of multi-bit CCA-secure PKE from single-bit CCA-secure PKE.

It is well-known that encrypting each bit of a plaintext string independently is not CCA-secure—the resulting scheme is *malleable*. We therefore investigate whether this malleability can be dealt with using the conceptually simple approach of applying a suitable non-malleable code (Dziembowski *et al.*, ICS '10) to the plaintext and subsequently encrypting the resulting codeword bit-by-bit. We find that an attacker's ability to ask multiple decryption queries requires that the underlying code be *continuously* non-malleable (Faust *et al.*, TCC '14). Since, as we show, this flavor of non-malleability can only be achieved if the code is allowed to "self-destruct," the resulting scheme inherits this property and therefore only achieves a weaker variant of CCA security.

We formalize this new notion of so-called *indistinguishability under self-destruct attacks (IND-SDA)* as CCA security with the restriction that the decryption oracle stops working once the attacker submits an invalid ciphertext. We first show that the above approach based on non-malleable codes yields a solution to the problem of domain extension for IND-SDA-secure PKE, provided that the underlying code is continuously non-malleable against (a reduced form of) bit-wise tampering. Then, we prove that the code of Dziembowski *et al.* is actually already continuously non-malleable against bit-wise tampering.

We further investigate the notion of security under self-destruct attacks and combine IND-SDA security with *non-malleability under chosen-ciphertext attacks (NM-CPA)* to obtain the *strictly* stronger notion of *non-malleability under self-destruct attacks (NM-SDA)*. We show that NM-SDA security can be obtained from basic IND-CPA security by means of a black-box construction based on the seminal work by Choi *et al.* (TCC '08). Finally, we provide a domain extension technique for building a multi-bit NM-SDA scheme from a single-bit NM-SDA scheme. To achieve this goal, we define and construct a novel type of continuous non-malleable code, called *secret-state NMC*, since, as we show, standard continuous NMCs are *insufficient* for the natural "encode-then-encrypt-bit-by-bit" approach to work.

*This paper is the full version of the papers "From Single-Bit to Multi-bit Public-Key Encryption via Non-malleable Codes" (appeared in TCC 2015, LNCS 9014, pp. 532-560, Springer, 2015) and "Non-Malleable Encryption: Simpler, Shorter, Stronger" (appeared in TCC 2016-A, LNCS 9562, pp. 306-335, Springer, 2016).

[†]Supported by SNF project no. 200020-132794. Work done while author was at ETH Zurich.

[‡]Partially supported by gifts from VMware Labs and Google, and NSF grants 1319051, 1314568, 1065288, 1017471.

[§]Supported by SNF project no. 200020-132794.

[¶]Work done while author was at ETH Zurich and UC San Diego. Author was partially supported by the SNF Fellowship P2EZP2_155566 and NSF grants CNS-1228890 and CNS-1116800.

Contents

1 Introduction

A public-key encryption (PKE) scheme enables a sender A to send messages to a receiver B confidentially if B can send a single message, the public key, to A authentically. A encrypts a message with the public key and sends the ciphertext to B via a channel that could be authenticated or insecure, and B decrypts the received ciphertext using the private key. Following the seminal work of Diffie and Hellman [37], the first formal definition of public-key encryption has been provided by Goldwasser and Micali [49], and to date numerous instantiations of this concept have been proposed, e.g., [71, 41, 32, 46, 51, 57, 72, 70], for different security properties and based on various different computational assumptions.

Many security notions for public-key encryption (PKE) have been proposed. The most basic one is that of indistinguishability under chosen-plaintext attacks (IND-CPA) [49], which requires that an adversary with no decryption capabilities be unable to distinguish between the encryption of two messages. Although extremely important and useful for a number of applications, in many cases IND-CPA security is not sufficient. For example, consider the simple setting of an electronic auction, where the auctioneer U publishes a public key pk, and invites several participants $P_1, \ldots, P_q$ to encrypt their bids b_i under pk. As was observed in the seminal paper of Dolev *et al.* [38], although IND-CPA security of encryption ensures that P_1 cannot decrypt a bid of P_2 under the ciphertext e_2, it leaves open the possibility that P_1 can construct a special ciphertext e_1 which decrypts to a *related* bid b_1 (e.g., $b_1 = b_2 + 1$). Hence, to overcome such "malleability" problems, stronger forms of security are required.

The strongest such level of PKE security is indistinguishability under chosen-ciphertext attacks (IND-CCA), where the adversary is given unrestricted, adaptive access to a decryption oracle (modulo not being able to ask on the "challenge ciphertext"). This notion is sufficient for most natural applications of PKE, and several generic [38, 66, 73, 15, 61] and concrete [32, 33, 58, 52] constructions of IND-CCA secure encryption schemes are known by now. Unfortunately, all these constructions either rely on specific number-theoretic assumptions, or use much more advanced machinery (such as non-interactive zero-knowledge proofs or identity-based encryption) than IND-CPA secure encryption. Indeed, despite numerous efforts (e.g., a partial negative result [48]), the relationship between IND-CPA and IND-CCA security remains unresolved until now. This motivates the study of various "middle-ground" security notions between IND-CPA and IND-CCA, which are sufficient for applications, and, yet, might be constructed from simpler basic primitives (e.g., any IND-CPA encryption).

One such influential notion is non-malleability under chosen-plaintext attacks (NM-CPA), originally introduced by Dolev *et al.* [38] with the goal of precisely addressing the auction example above, by demanding that an adversary not be able to maul ciphertexts to other ciphertexts encrypting related plaintexts. As was later shown by Bellare and Sahai [14] and by Pass *et al.* [69], NM-CPA is equivalent to security against adversaries with access to a *non-adaptive* decryption oracle, meaning that the adversary can only ask one "parallel" decryption query. Although NM-CPA appears much closer to IND-CCA than IND-CPA security, a seminal result by Pass *et al.* [68] showed that one can generically build NM-CPA encryption from any IND-CPA-secure scheme, and Choi *et al.* [25] later proved that this transformation can also be achieved via a black-box construction. Thus, NM-CPA schemes can be potentially based on weaker assumptions than IND-CCA schemes, and yet suffice for important applications.

Looking beyond non-malleable encryption, Cramer *et al.* [31] build *bounded-query* chosen-ciphertext secure schemes from chosen-plaintext secure ones, and Lin and Tessaro [60] show how the security of weakly chosen-ciphertext secure schemes can be amplified. A line of work started by Myers, Sergi, and shelat [64] and continued by Dachman-Soled [34] shows how to obtain chosen-ciphertext secure schemes from plaintext-aware ones. Most relevant for our work, however, are the results of Myers and shelat [65] and Hohenberger, Lewko, and Waters [53], who generically build a multi-bit chosen-ciphertext secure scheme from a single-bit chosen-ciphertext secure one.

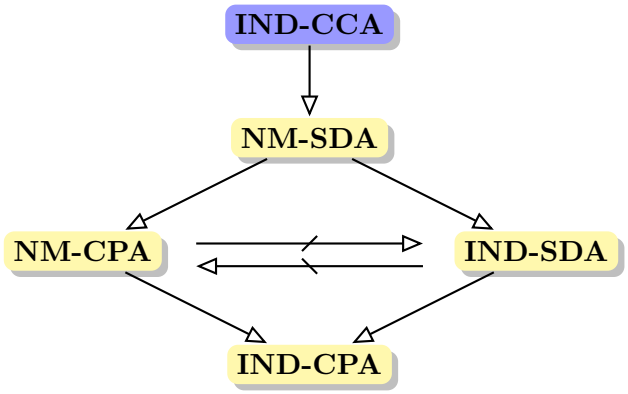

Figure 1: *Diagram of the main relationships between the security notions considered in this paper. $X \to Y$ means that X implies Y; $X \not\to Y$ indicates a separation between X and Y. Notions with the same color are equivalent under black-box transformations; notions with different colors are not known to be equivalent.*

1.1 Contributions

Non-malleability under self-destruct attacks. In this work, we introduce a strengthening of NM-CPA security for PKE that we term *non-malleability under (chosen-ciphertext) self-destruct attacks (NM-SDA)*. Intuitively, in NM-SDA the adversary is allowed to ask many adaptive "parallel" decryption queries (i.e., a query consists of many ciphertexts) up to the point when the first invalid ciphertext is submitted. In such a case, the whole parallel decryption query containing an invalid ciphertext is still answered in full, but no future decryption queries are allowed.

An interesting degenerate case of NM-SDA is when each decryption query consists of a *single* ciphertext. The latter yields a notion weaker than full CCA, which we term *indistinguishability under (chosen-ciphertext) self-destruct attacks (IND-SDA)*. Roughly, IND-SDA security is CCA security with the twist that the decryption oracle stops working once the adversary submits an invalid ciphertext.

As we argue below, both IND-SDA and NM-SDA seem to apply better to the auction example above. First, unlike with basic NM-CPA, with both IND-SDA and NM-SDA the auctioneer can reuse the same public key pk, provided no invalid ciphertexts were submitted. Second, with IND-SDA the auctioneer can reuse the secret key for subsequent auctions, as long as all encrypted bids are valid; unfortunately, if an invalid ciphertext is submitted, even the results of the *current* auction should be discarded, as IND-SDA security is not powerful enough to argue that the decryptions of the remaining ciphertexts are unrelated w.r.t. prior plaintexts. Third, unlike IND-SDA, with NM-SDA the current auction can be safely completed, even if some ciphertexts are invalid. Compared to IND-CCA, however, the auctioneer will still have to change its public key for *subsequent* auctions if some of the ciphertexts are invalid. Still, one can envision situations where parties are penalized for submitting such malformed ciphertexts, in which case NM-SDA security might be practically sufficient, leading to an implementation under (potentially) weaker computational assumptions as compared to using a full-blown IND-CCA PKE.

Having introduced and motivated NM-SDA and IND-SDA security, we provide a comprehensive study of these notions, and their relationship to other PKE security definitions. First, we observe that the notions of NM-CPA and IND-SDA are incomparable, meaning that there are (albeit contrived) schemes that satisfy the former but not the latter notion and vice versa. This also implies that our notion of NM-SDA security, which naturally combines NM-CPA and IND-SDA, is strictly stronger than either of the two other notions (cf. Figure 1). By being stronger than both NM-CPA and IND-SDA, NM-SDA security appears to be the strongest natural PKE security notion that is still weaker (as we give evidence below) than IND-CCA—together with q-bounded CCA-secure PKE [31], to which it seems incomparable.

Domain-extension for PKE. Consider the problem of transforming a single-bit PKE scheme into a multi-bit PKE scheme. A naïve attempt at solving this problem would be to encrypt each bit m_i of a plaintext $m = m_1 \cdots m_k$ under an independent public key pk_i of the single-bit scheme. Unfortunately,

the resulting scheme is *malleable* (even if the underlying single-bit scheme is not): given a ciphertext $e = (e_1, \ldots, e_k)$, where e_i is an encryption of m_i, an attacker can generate a new ciphertext $e' \neq e$ that decrypts to a related message, for instance by copying the first ciphertext component e_1 and replacing the other components by fresh encryptions of, say, 0.

The above malleability issue suggests the following natural "encode-then-encrypt-bit-by-bit" approach: first encode the message using a non-malleable code[1] (a concept introduced by Dziembowski *et al.* [40]) to protect its integrity, obtaining an n-bit codeword $c = c_1 \cdots c_n$; then encrypt each bit c_i of the codeword using public key pk_i as in the naïve protocol from above.

It turns out that non-malleable codes as introduced by [40] are not sufficient: Since they are only secure against a single tampering, the security of the resulting scheme would only hold with respect to a single decryption. *Continuously* non-malleable codes (Faust *et al.* [44]) allow us to extend this guarantee to multiple decryptions. However, such codes "self-destruct" once an attack has been detected, and, therefore, so must any PKE scheme built on top of them. This is a restriction that we prove to be unavoidable for this approach based on non-malleable codes.

We first prove that the above approach allows to build multi-bit NM-SDA-secure (resp. NM-CPA-secure) PKE from single-bit NM-SDA-secure (resp. NM-CPA-secure) PKE, provided that the underlying code satisfies a suitable strengthening (see below) of continuous non-malleable against (a *reduced* form of) bit-wise tampering, which we denote by the tampering family $\mathcal{F}_{\mathsf{set}}$. Summarizing:

Theorem 1 (Informal)**.** *Let λ be the security parameter. There is a black-box construction of a λ-bit NM-SDA (resp. NM-CPA, IND-SDA) PKE scheme from a single-bit NM-SDA (resp. NM-CPA, IND-SDA) PKE scheme, making $\mathcal{O}(\lambda)$ calls to the underlying single-bit scheme.*[2]

The main technical challenge when analyzing the "encode-then-encrypt-bit-by-bit" approach for the cases of NM-CPA and NM-SDA is to deal with the *parallel* decryption queries: in order for the combined scheme to be NM-CPA or NM-SDA secure, the NMC needs to be resilient against parallel tamper queries as well. However, we show that no standard non-malleable code (as originally defined by Dziembowski *et al.* [40]) can achieve this flavor of non-malleability already for a single, big enough, parallel tampering query. Fortunately, we observe that the NMC concept can be extended to allow the decoder to make use of (an initially generated) secret state, which simply becomes part of the secret key in the combined scheme. This modification of NMCs—called secret-state NMCs—allows us to achieve resilience against parallel tampering and may be useful for analyzing other constructions of (non-malleable) cryptographic primitives using NMCs. Hence, our question reduces to building a secret-state non-malleable code resilient against continuous parallel tampering attacks from $\mathcal{F}_{\mathsf{set}}$. We build such a code unconditionally, by combining the notion of linear error-correcting secret sharing (see [40]) with the idea of a secret "trigger set" [25].

On the other hand, in the case of IND-SDA, where each decryption query consists of a single ciphertext, it suffices to use any standard (i.e., without secret state) continuously non-malleable code against $\mathcal{F}_{\mathsf{set}}$. To this end, we show that a simplified variant of the code by Dziembowski *et al.* [40] is already continuously non-malleable against $\mathcal{F}_{\mathsf{set}}$.[3] This constitutes the first *information-theoretically* secure continuously non-malleable code, a contribution that we believe is of independent interest, and forms one of the technical cores of this paper.

Improving security achievable from IND-CPA. Finally, we also prove that there exists a black-box construction of NM-SDA-secure PKE from any IND-CPA-secure PKE scheme. Given the negative result in [48], we take this as evidence that NM-SDA is a strictly weaker notion than full-blown IND-CCA.

[1] Roughly, a code is non-malleable w.r.t. a function class $\mathcal{F}$ if the message obtained by decoding a codeword modified via a function in $\mathcal{F}$ is either the original message or a completely unrelated value.

[2] For longer than λ-bit messages, one can also use standard hybrid encryption.

[3] The full variant of said code achieves continuous non-malleability even against *full* bit-wise tampering.

Theorem 2 (Informal)**.** *There exists a black-box construction of an NM-SDA-secure PKE scheme with rate $\Omega(1/\lambda)$ from an IND-CPA-secure PKE scheme with constant rate, in which the encryption algorithm calls the underlying IND-CPA encryption algorithm $\Theta(\lambda^2)$ times.*

Specifically, we show that a generalization of the construction by Choi *et al.* [25] already achieves NM-SDA security (rather than only NM-CPA security). Our proof much follows the pattern of the original one, except for one key step in the proof, where a brand new proof technique is required. Intuitively, one needs to argue that no sensitive information about the secret "trigger set" is leaked to the adversary, unless one of the ciphertexts is invalid. This is achieved via a rather general technique for analyzing security of so called "parallel stateless self-destruct games," which may be interesting in its own right (e.g., it is also used for several other proofs in this work).

Along the way, we also manage to slightly abstract the transformation of [25] and to re-phrase it in terms of certain linear error-correcting secret-sharing schemes (LECSSs) satisfying a special property (as opposed to using Reed-Solomon codes directly as an example of such a scheme). Aside from a more modular presentation (which gives a more intuitive explanation for the elegant scheme of Choi *et al.* [25]), this also allows us to instantiate the required LECSS more efficiently and thereby improve the rate of the transformation of [25] by a factor linear in the security parameter (while also arguing NM-SDA, instead of NM-CPA, security).[4]

1.2 Related Work

Non-malleable codes. Several non-malleable codes constructions exist in the literature, both in the information-theoretic and in the computational setting, covering a plethora of models including bit-wise independent tampering and permutations [40, 24, 6, 7, 36], block-wise [17] constant-state [21, 55, 3] and split-state [40, 62, 39, 2, 24, 23, 44, 4, 1, 18, 5, 19, 56, 59, 42, 67, 28, 35], tampering by functions with few fixed points and high entropy [54], space-bounded tampering [43, 22], tampering by circuits with limited complexity [45, 54, 10, 20, 8, 11, 12], and bounded polynomial-time tampering [9].

PKE domain extension. For several security notions in public-key cryptography, it is known that single-bit public-key encryption implies multi-bit public-key encryption. For IND-CPA, this question is simple [49], since the parallel repetition of a single-bit scheme (i.e., encrypting every bit of a message separately) yields an IND-CPA secure multi-bit scheme.

For the other notions considered in this paper, i.e., for NM-CPA, IND-SDA, and NM-SDA, as well as for IND-CCA, the parallel repetition (even using independent public keys) is not a scheme that achieves the same security level as the underlying single-bit scheme. While we provide a single-to-multi-bit transformation for NM-CPA, IND-SDA, and NM-SDA, Myers and Shelat [65], as well as Hohenberger *et al.* [53], provide (much) more complicated transformations for IND-CCA security.

Damgård *et al.* [36] showed how to reduce the public/secret key size of our single-to-multi-bit transform for IND-SDA by using a continuously non-malleable code resistant to permutations and overwrites.

Non-malleability from semantic security. The transformation of [25] gives an NM-CPA scheme such that its encryption algorithm calls the underlying IND-CPA scheme $\Theta(\lambda^2)$ times, where λ is the security parameter. For example, assuming a constant-rate IND-CPA encryption, this gives a $\Theta(\lambda)$-bit NM-CPA scheme with the ciphertext length of $\Theta(\lambda^3)$. In contrast, our analysis of their transformation allows to obtain $\Theta(\lambda^3)$-bit ciphertexts to encrypt $\Theta(\lambda^2)$-bit messages while at the same time achieving the stronger notion of NM-SDA.

[4]Note that Choi *et al.* [25] consider the ciphertext blow-up between the underlying IND-CPA scheme and the resulting scheme as quality measure of their construction, while we consider the rate (number of plaintext bits per ciphertext bit) of the resulting scheme.

Construction	Ciphertext length			Security
	$\ell = o(\lambda)$	$\ell = \Theta(\lambda)$	$\ell = \Omega(\lambda^2)$	
[25]	$\Theta(\lambda^3)$	$\Theta(\lambda^2\ell)$	$\Theta(\lambda^2\ell)$	NM-CPA
[26]	$\Theta(\lambda^2)$	$\Theta(\lambda\ell)$	$\Theta(\ell)$	NM-CPA
Ours	$\Theta(\lambda^3)$	$\Theta(\lambda^2\ell)$	$\Theta(\lambda\ell)$	NM-SDA

Construction	Ciphertext length			Security
	$\ell = \Theta(\lambda)$	$\ell = \Theta(\lambda^2)$	$\ell = \Omega(\lambda^3)$	
[25] + [50]	$\Theta(\lambda^2\ell)$	$\Theta(\lambda\ell)$	$\Theta(\ell)$	NM-CPA
[26] + [50]	$\Theta(\lambda\ell)$	$\Theta(\ell)$	$\Theta(\ell)$	NM-CPA
Ours + [50]	$\Theta(\lambda^2\ell)$	$\Theta(\lambda\ell)$	$\Theta(\ell)$	NM-SDA

Figure 2: *A comparison of black-box constructions of non-malleable PKE from semantically-secure PKE. The parameter λ is the security parameter, and ℓ is the plaintext length. We assume the underlying IND-CPA encryption has a constant rate for messages of length $\Omega(\lambda)$; encrypting $o(\lambda)$-long messages with IND-CPA encryption is assumed to be $\Theta(\lambda)$-long. The table on the right uses the hybrid encryption scheme of Herranz* et al. *[50].*

In a recent and concurrent[5] work, Choi *et al.* [26] presented a new transformation that allows to achieve the first black-box construction making $\Theta(\lambda)$ calls to the underlying IND-CPA encryption algorithm; this yields an improved rate, although for the weaker notion of NM-CPA. We provide a more detailed comparison in Figure 2.

Previous publications. An abridged version of this work appeared as [29, 27]. In particular, in [29] we introduced the notion of IND-SDA security,[6] and solved the problem of domain extension for that notion. In [27], instead, we considered NM-SDA security, solved the problem of domain extension for that notion, and showed how to obtain NM-SDA security from IND-CPA security in a black-box way. This paper is the full version of the aforementioned works.

1.3 Paper Organization

The rest of this paper is organized as follows. We start with some preliminary definition, in Section 2. Our general indistinguishability paradigm for analyzing "parallel stateless self-destruct games" can be found in Section 3. The formal notions of NM-CPA, IND-SDA, and NM-SDA are given in Section 4, where we also show that IND-SDA and NM-CPA are incomparable.

Section 5 contains all our results regarding non-malleable codes, in particular the information-theoretic code constructions for continuous bit-wise independent tampering (for both parallel and non-parallel attacks), and the impossibility result for stateless codes in the case of parallel tampering. The proof of Theorem 1 can be found in Section 6, whereas Section 7 is focused on the proof of Theorem 2.

2 Preliminaries

This section introduces notational conventions and basic concepts that we use throughout the work.

2.1 Notation

Bits and symbols. If $x \in \{0,1\}^n$ is an n-bit string, then $x[i]$ denotes its i^{th} bit. For two n-bit strings x and y, $d_{\mathsf{H}}(x,y)$ denotes their hamming distance (i.e., the number of bit positions in which they differ), and $w_{\mathsf{H}}(x)$ denotes the hamming weight of x (i.e., the number of positions $i \in [n]$ such that $x[i] = 1$).

Oracle algorithms. Oracle algorithms are algorithms that can make special oracle calls. An algorithm A with an oracle O is denoted by $A(O)$. Note that oracle algorithms may make calls to other oracle algorithms (e.g., $A(B(O))$).

[5] [26] was published in 2018, while our paper was still under review.

[6] The original name used in [29] is self-destruct chosen-ciphertext attacks security.

Distinguishers and reductions. A *distinguisher* is an (possibly randomized) oracle algorithm $D(\cdot)$ that outputs a single bit. The distinguishing advantage on two (possibly stateful) oracles S and T is defined by

$$\Delta^D(S,T) \quad := \quad |\mathsf{P}[D(S)=1] - \mathsf{P}[D(T)=1]|,$$

where the probabilities are taken over the randomness of D as well as S and T, respectively.

Reductions between distinguishing problems are modeled as oracle algorithms as well. Specifically, when reducing distinguishing two oracles U and V to distinguishing S and T, one exhibits an oracle algorithm $R(\cdot)$ such that $R(U)$ behaves as S and $R(V)$ as T; then, $\Delta^D(S,T) = \Delta^D(R(U),R(V)) = \Delta^{D(R(\cdot))}(U,V)$.

2.2 Linear Error-Correcting Secret Sharing

Definition 1 (Coding scheme)**.** *A (k,n)-coding scheme* $(\mathrm{Enc},\mathrm{Dec})$ *over a field $\mathbb{F}$ consists of a randomized encoding function* $\mathrm{Enc}:\mathbb{F}^k \to \mathbb{F}^n$ *and a deterministic decoding function* $\mathrm{Dec}:\mathbb{F}^n \to \mathbb{F}^k \cup \{\perp\}$ *such that* $\mathrm{Dec}(\mathrm{Enc}(x)) = x$ *(with probability* 1 *over the randomness of the encoding function) for each $x \in \mathbb{F}^k$. The special symbol $\perp$ indicates an* invalid codeword*.*

The following notion of linear error correcting secret sharing, introduced by Dziembowski *et al.* [40], is used in several places in this paper.

Definition 2 (Linear error-correcting/detecting sharing scheme)**.** *Let $\mathbb{F}$ be a finite field. A (k,n,δ,τ)* linear error-correcting secret sharing (LECSS) *over $\mathbb{F}$ is a triple of algorithms $(\mathsf{E},\mathsf{D},\mathsf{R})$ over $\mathbb{F}$ such that (E,D) is a coding scheme and the following properties are satisfied:*

- Linearity: *For any vector w output by E and any $c \in \mathbb{F}^n$,*

$$\mathsf{D}(w+c) = \begin{cases} \perp & \text{if } \mathsf{D}(c) = \perp, \text{ and} \\ \mathsf{D}(w) + \mathsf{D}(c) & \text{otherwise.} \end{cases}$$

- Minimum distance: *For any $c \in \mathbb{F}^n$ with $0 < w_{\mathsf{H}}(c) < \delta n$, $\mathsf{D}(c) = \perp$.*
- Secrecy: *The symbols of a codeword are individually uniform over $\mathbb{F}$ and τn-wise independent (over the randomness of E).*
- Error correction: *It is possible to efficiently correct up to $\delta n/2$ errors, i.e., for any $x \in \mathbb{F}^k$ and any w output by $\mathsf{E}(x)$, if $d_{\mathsf{H}}(c,w) \le t$ for some $c \in \mathbb{F}^n$ and $t < \delta n/2$, then $\mathsf{R}(c,t) = w$.*

A LECSS without the error correction property is called a linear error-detecting sharing scheme (LEDSS)*.*

This paper considers various instantiations of LECSSs, which are described in Sections 5.3.2 and 7.3, where they are used.

2.3 One-Time Signatures

A *digital signature scheme* (DSS) is a triple of algorithms $\Sigma = (KG, S, V)$, where the key-generation algorithm KG outputs a key pair $(\mathsf{sk},\mathsf{vk})$, the (probabilistic) signing algorithm S takes a message m and a signing key sk and outputs a signature $\sigma \leftarrow S_{\mathsf{sk}}(m)$, and the verification algorithm takes a verification key vk, a message m, and a signature σ and outputs a single bit $V_{\mathsf{vk}}(m,\sigma)$. A *(strong) one-time signature (OTS) scheme* is a digital signature scheme that is secure as long as an adversary only observes a single signature. More precisely, OTS security is defined using the following game $G^{\Sigma,\mathsf{ots}}$ played by an adversary A: Initially, the game generates a key pair $(\mathsf{sk},\mathsf{vk})$ and hands the verification key vk to A. Then, A can specify a single message m for which he obtains a signature $\sigma \leftarrow S_{\mathsf{sk}}(m)$. Then, the adversary outputs a pair (m',σ'). The adversary wins the game if $(m',\sigma') \neq (m,\sigma)$ and $V_{\mathsf{vk}}(m',\sigma') = 1$. The *advantage* of A is the probability (over all involved randomness) that A wins the game, and is denoted by $\Gamma^A(G^{\Sigma,\mathsf{ots}})$.

Definition 3 (One-time signature)**.** *A DSS scheme Σ is a (t, ε)*-strong one-time signature scheme *if for all adversaries A with running time at most t, $\Gamma^A(G^{\Sigma,\mathsf{ots}}) \leq \varepsilon$.*

2.4 Message Authentication Codes

A *message authentication code (MAC)* is a pair of algorithms (T, V), where the tagging algorithm T takes as input a message m and a key $K \in \{0,1\}^\lambda$ and outputs a tag $\phi \leftarrow T_K(m)$, and where the verification algorithm V takes a key K, a message m, and a tag ϕ and outputs a bit $V_K(m, \phi)$.

MAC security is defined using the following game G^{mac} played by an adversary A: Initially, the game chooses a random key K. Then, A gets access to a tagging oracle, which returns a tag $\phi \leftarrow T_K(m)$ when given a message m, and to a verification oracle, which outputs $V_K(m, \phi)$ when given a message m and a tag K. The adversary wins the game if he submits to the verification oracle a pair (m, ϕ) that is not a query-answer pair for the tagging oracle and for which $V_K(m, \phi) = 1$.

Definition 4 (Security of MACs)**.** *A MAC (T, V) is* (t, u, v, ε)-secure *if for all adversaries A with running time at most t, making at most u tag queries and at most v verification queries, $\Gamma^A(G^{\mathsf{mac}}) \leq \varepsilon$.*

2.5 Miscellaneous

We make use of the following Chernoff bound.

Theorem 3. *Let $X_1, \ldots, X_n$ be i.i.d. with $X_i \sim \mathrm{Be}(p_i)$. Then, for $X := \sum_i X_i$ and $\mu := \sum_i p_i$,*

$$\mathsf{P}[X \leq (1-\varepsilon)\mu] \quad \leq \quad e^{-\mu\varepsilon^2/2}$$

for any $\varepsilon \in (0, 1]$.

The following fact will be useful:

Proposition 4 ([75, Lemma 3.1.15])**.** *Let X and Y be random variables with statistical distance $d := \Delta(X, Y)$, and let $\bar{X}$ (resp. $\bar{Y}$) be n independent copies of X (resp. Y). Then,*

$$\Delta(\bar{X}, \bar{Y}) \geq 1 - 2e^{-nd^2/2}.$$

2.5.1 Plotkin Bound

The following theorem allows to bound the number of codewords of a code over a binary alphabet with relative minimum distance $\delta > 1/2$.

Theorem 5. *For a code over a binary alphabet with block length n and distance $d \geq \frac{n}{2} + 1$, the maximum number of codewords is*

$$A(n, d) \leq \frac{d}{d - \frac{n}{2}} \leq 1 + \frac{1}{2\varepsilon}$$

where $\varepsilon = \frac{d}{n} - \frac{1}{2}$.

A proof can be found in [63, p. 41].

3 A General Indistinguishability Paradigm

3.1 Parallel Stateless Self-Destruct Games

A recurring issue in this paper are proofs that certain self-destruct games answering successive parallel decryption/tampering queries are indistinguishable. We formalize such games as *parallel stateless self-destruct games*. Examples of such games include those for defining NM-CPA, IND-SDA, and NM-SDA.

Definition 5 (Parallel stateless self-destruct game)**.** *An oracle U is a* parallel stateless self-destruct (PSSD) game *if*

- *it accepts parallel queries in which each component is from some set $\mathcal{X}$ and answers them by vectors with components from some set $\mathcal{Y}$,*
- *$\perp \in \mathcal{Y}$,*
- *there exists a function $g : \mathcal{X} \times \mathcal{R} \to \mathcal{Y}$ such that every query component $x \in \mathcal{X}$ is answered by $g(x, r)$, where $r \in \mathcal{R}$ is the internal randomness of U, and*
- *the game self-destructs, i.e., after the first occurrence of $\perp$ in an answer vector all* further *outputs are $\perp$.*

3.2 The Self-Destruct Lemma

A PSSD game can be transformed into a related one by "bending" the answers to some of the queries $x \in \mathcal{X}$ to the value $\perp$. This is captured by the following definition:

Definition 6 (Bending of a PSSD)**.** *Let U be a PSSD game that behaves according to g and let $\mathcal{B} \subseteq \mathcal{X}$. The* $\mathcal{B}$-bending of U*, denoted by U', is the PSSD game that behaves according to g', where*

$$g'(x, r) \quad = \quad \begin{cases} \perp & \text{if } x \in \mathcal{B}, \\ g(x, r) & \text{otherwise.} \end{cases}$$

The *self-destruct lemma* below states that in order to bound the distinguishing advantage between a PSSD and its bending, one merely needs to analyze a single, non-parallel query, provided that all non-bent queries x can only be answered by a unique value y_x or $\perp$. Intuitively, the lemma says that adaptivity does not help distinguish in such cases.

Lemma 6. *Let U be a PSSD game and U' its $\mathcal{B}$-bending for some $\mathcal{B} \subseteq \mathcal{X}$. If for all $x \notin \mathcal{B}$ there exists $y_x \in \mathcal{Y}$ such that*

$$\{g(x, r) \mid r \in \mathcal{R}\} \quad = \quad \{y_x, \perp\},$$

then, for all distinguishers D,

$$\Delta^D(U, U') \quad \leq \quad p \cdot \max_{x \in \mathcal{B}} \mathsf{P}[g(x, R) \neq \perp],$$

where the probability is over the choice of R.

Proof. Fix a distinguisher D and denote by R and R' the random variables corresponding to the internal randomness of U and U', respectively. Call a value $x \in \mathcal{X}$ *dangerous* if $x \in \mathcal{B}$ and a query dangerous if it contains a dangerous value.

In the random experiment corresponding to the interaction between D and U, define the event E that the first dangerous query contains a dangerous value X with $g(X, R) \neq \perp$ and that the self-destruct has not been provoked yet. Similarly, define the event E' for the interaction between D and U' that the first dangerous query contains a dangerous value X' with $g(X', R') \neq \perp$ and that the self-destruct has not been provoked yet.[7]

Clearly, U and U' behave identically unless E resp. E' occur. Thus, it remains to bound $\mathsf{P}[E] = \mathsf{P}[E']$. To that end, note that adaptivity does not help in provoking E. For any distinguisher D, there exists a *non-adaptive* distinguisher $\tilde{D}$ such that whenever D provokes E, so does D'. D' proceeds as follows: First, it interacts with D only. Whenever D asks a non-dangerous query, D' answers every component $x \notin \mathcal{B}$ by y_x. As soon as D specifies a dangerous query, D' stops its interaction with D and sends all queries to U.

[7] Note that the function g is the *same* in the definitions of either event.

Fix all randomness in experiment $D'(U)$, i.e., the coins of D (inside D') and the randomness r of U. Suppose D would provoke E in the direct interaction with U. In such a case, all the answers by D' are equal to the answers by U, since, by assumption, the answers to components $x \notin \mathcal{B}$ in non-dangerous queries are y_x or $\perp$ and the latter is excluded if E is provoked. Thus, whenever D provokes E, D' provokes it as well.

The success probability of non-adaptive distinguishers D is upper bounded by the probability over R that their first dangerous query provokes E, which is at most $p \cdot \max_{x \in \mathcal{B}} \mathsf{P}[g(x, R) \neq \perp]$. □

4 Non-Malleability under Self-Destruct Attacks

A *public-key encryption (PKE) scheme* with message space $\mathcal{M} \subseteq \{0,1\}^*$ and ciphertext space $\mathcal{C}$ is defined as three algorithms $\Pi = (KG, E, D)$, where the key-generation algorithm KG outputs a key pair $(\mathsf{pk}, \mathsf{sk})$, the (probabilistic) encryption algorithm E takes a message $m \in \mathcal{M}$ and a public key pk and outputs a ciphertext $e \leftarrow E_{\mathsf{pk}}(m)$, and the decryption algorithm takes a ciphertext $e \in \mathcal{C}$ and a secret key sk and outputs a plaintext $m \leftarrow D_{\mathsf{sk}}(e)$. The output of the decryption algorithm can be the special symbol $\perp$, indicating an invalid ciphertext. A PKE scheme is correct if $m = D_{\mathsf{sk}}(E_{\mathsf{pk}}(m))$ (with probability 1 over the randomness in the encryption algorithm) for all messages m and all key pairs $(\mathsf{pk}, \mathsf{sk})$ generated by KG.

4.1 The Definition

Security notions for PKE schemes in this paper are formalized using the distinguishing game $G_b^{\Pi,q,p}$, depicted in Figure 3: The distinguisher (adversary) is initially given a public key and then specifies two messages m_0 and m_1. One of these, namely m_b, is encrypted and the adversary is given the resulting challenge ciphertext. During the entire game, the distinguisher has access to a decryption oracle that allows him to make at most q decryption queries, each consisting of at most p ciphertexts. Once the distinguisher specifies an invalid ciphertext, the decryption oracle self-destructs, i.e., no additional decryption queries are answered.

The general case is obtained when both q and p are arbitrary (denoted by $q = p = *$), which leads to our main definition of non-malleability under (chosen-ciphertext) self-destruct attacks (NM-SDA). For readability, set $G_b^{\Pi,\mathsf{nm\text{-}sda}} := G_b^{\Pi,*,*}$ for $b \in \{0,1\}$. Formally, NM-SDA is defined as follows:

Definition 7 (Non-malleability under self-destruct attacks)**.** *A PKE scheme* Π *is* (t, q, p, ε)-NM-SDA-secure *if for all distinguishers* D *with running time at most* t *and making at most* q *decryption queries of size at most* p *each,*

$$\Delta^D(G_0^{\Pi,\mathsf{nm\text{-}sda}}, G_1^{\Pi,\mathsf{nm\text{-}sda}}) \quad \leq \quad \varepsilon.$$

All other relevant security notions in this paper can be derived as special cases of the above definition, by setting the parameters q and p to different values.

Chosen-plaintext security (IND-CPA). In this variant, the distinguisher is not given access to a decryption oracle, i.e., $q = p = 0$. For readability, set $G_b^{\Pi,\mathsf{ind\text{-}cpa}} := G_b^{\Pi,0,0}$ for $b \in \{0,1\}$ in the remainder of this paper. We say that Π is (t, ε)-IND-CPA-secure if it is, in fact, $(t, 0, 0, \varepsilon)$-NM-SDA-secure.

Non-malleability (NM-CPA). A scheme is non-malleable under chosen-plaintext attacks [68] (NM-CPA), if the adversary can make a single decryption query consisting of arbitrarily many ciphertexts, i.e., $q = 1$ and p arbitrary (denoted by $p = *$). Similarly to above, set $G_b^{\Pi,\mathsf{nm\text{-}cpa}} := G_b^{\Pi,1,*}$ for $b \in \{0,1\}$. We say that Π is (t, p, ε)-NM-CPA-secure if it is, in fact, $(t, 1, p, \varepsilon)$-NM-SDA-secure.[8]

[8] Note that the way NM-CPA is defined here is slightly stronger than the normal notion. This is due to the adversary's ability to ask a parallel decryption query at any time—as opposed to only after receiving the challenge ciphertext in earlier definitions (cf., e.g., [68]).

Distinguishing Game $G_b^{\Pi,q,p}$

init
- $\mathsf{ctr} \leftarrow 0$
- $(\mathsf{pk}, \mathsf{sk}) \leftarrow KG$
- **output** pk

on $(\mathsf{chall}, m_0, m_1)$ *with* $|m_0| = |m_1|$
- $e \leftarrow E_{\mathsf{pk}}(m_b)$
- **output** e

on $(\mathsf{dec}, e^{(1)}, \ldots, e^{(p)})$
- $\mathsf{ctr} \leftarrow \mathsf{ctr} + 1$
- **for** $j \leftarrow 1$ **to** p
  - $m^{(j)} \leftarrow D_{\mathsf{sk}}(e^{(j)})$
  - **if** $e^{(j)} = e$
    - $m^{(j)} \leftarrow \mathsf{test}$
- **output** $(m^{(1)}, \ldots, m^{(p)})$
- **if** $\exists j : m^{(j)} = \perp$ *or* $\mathsf{ctr} \geq q$
  - **self-destruct**

Figure 3: *Distinguishing game* $G_b^{\Pi,q,p}$*, where* $b \in \{0,1\}$*, used to define security of a PKE scheme* $\Pi = (KG, E, D)$*. The numbers* $q, p \in \mathbb{N}$ *specify the maximum number of decryption queries and their size, respectively. The command* **self-destruct** *results in all future decryption queries being answered by* $\perp$*. Whenever one of the ciphertexts* $e^{(j)}$ *equals the challenge ciphertext* e*, the corresponding message* $m^{(j)}$ *is set to the string* test*.*

Indistinguishability under self-destruct attacks (IND-SDA). This variant, introduced in [29], allows arbitrarily many queries to the decryption oracle, but each of them may consist of a single ciphertext only, i.e., q arbitrary (denoted by $q = *$) and $p = 1$. Once more, set $G_b^{\Pi,\mathsf{ind\text{-}sda}} := G_b^{\Pi,*,1}$. We say that Π is (t, q, ε)-IND-SDA-secure if it is, in fact, $(t, q, 1, \varepsilon)$-NM-SDA-secure.

Chosen-ciphertext security (IND-CCA). The standard notion of IND-CCA security can be obtained as a strengthening of NM-SDA where $q = *$, $p = 1$, and the decryption oracle never self-destructs. We do not define this notion formally, as it is not the main focus of this paper.

Asymptotic formulation. To allow for concise statements, sometimes we prefer to use an asymptotic formulation instead of stating concrete parameters. More precisely, we will say that a PKE scheme Π is X-secure (for X $\in$ {IND-CPA, NM-CPA, IND-SDA, NM-SDA}) if for all efficient adversaries the advantage ε in the corresponding distinguishing game is negligible in the security parameter.

4.2 Relating Indistinguishability and Non-Malleability

In this section we provide a separation between the notions of NM-CPA and IND-SDA security. Given such a separation, our notion of NM-SDA security (see Definition 7) is strictly stronger than either of the two other notions.

4.2.1 NM-CPA Does Not Imply IND-SDA

The modified scheme. Let λ be the security parameter and $\Pi = (KG, E, D)$ be a NM-CPA-secure PKE scheme with message space $\mathcal{M} = \{0,1\}^\lambda$. Consider the following modification $\Pi' = (KG', E', D')$ of Π (cf. Figure 4):

- The key generation algorithm KG' works as KG but additionally samples a uniformly random message $\rho \leftarrow \mathcal{M}$, which becomes part of the secret key.
- The encryption algorithm E' works as E except that it prepends a zero-bit to all ciphertexts.
- The decryption algorithm D' proceeds as follows, upon receiving a ciphertext $e' = \beta \| e$. If $\beta = 1$, it outputs ρ. If $\beta = 0$, it decrypts $m \leftarrow D_{\mathsf{sk}}(e)$. If $m = \rho$, the decryption algorithm outputs the secret key, and otherwise m.

PKE Scheme $\Pi' = (KG', E', D')$

Key Generation KG'
- $(\mathsf{pk}, \mathsf{sk}) \leftarrow KG$
- $\rho \leftarrow\!\!\$\ \mathcal{M}$
- $\mathsf{pk}' \leftarrow \mathsf{pk}$
- $\mathsf{sk}' \leftarrow (\rho, \mathsf{sk})$
- **return** $(\mathsf{pk}', \mathsf{sk}')$

Encryption $E'_{\mathsf{pk}'}(m)$
- $e \leftarrow E_{\mathsf{pk}}(m)$
- $e' \leftarrow 0 \,\|\, e$
- **return** e'

Decryption $D'_{\mathsf{sk}'}(e')$
- $\beta \,\|\, e \leftarrow e'$
- $m \leftarrow D_{\mathsf{sk}}(e)$
- **if** $\beta = 1$
  - **return** ρ
- **else**
  - **if** $m = \rho$
    - **return** sk
  - **else**
    - **return** m

Figure 4: *PKE scheme* Π' *based on an NM-CPA-secure PKE scheme* Π.

Security of the modified scheme. PKE scheme Π' clearly is not IND-SDA-secure: A distinguisher simply queries $1\|E_{\mathsf{pk}}(m)$ for some message m to obtain message ρ. By subsequently querying $0\|E_{\mathsf{pk}}(\rho)$, the distinguisher obtains the secret key.

The modified scheme is, however, still NM-CPA-secure as implied by the following lemma:

Lemma 7. *For all $p \in \mathbb{N}$, and all distinguishers D', there exists a distinguisher D such that*

$$\Delta^{D'}(G_0^{\Pi',1,p}, G_1^{\Pi',1,p}) \quad \leq \quad \Delta^{D}(G_0^{\Pi,1,p}, G_1^{\Pi,1,p}) \; + \; 2p \cdot 2^{-\lambda}.$$

Proof. Fix p and a distinguisher D'. Distinguisher D internally runs a copy of D' and works as follows: Initially, it chooses $\rho \leftarrow \mathcal{M}$ uniformly at random and outputs the public key received from its oracle. Upon receiving $(\mathsf{chall}, m_0, m_1)$ from D', D forwards it to its oracle, which returns a ciphertext e^*. D' then passes the value $0\|e^*$ to D'.

Moreover, D answers each component e' of the parallel decryption query received from D' as follows: It first parses e' as $\beta\|e$. Then, if $\beta = 1$, the answer to the query is ρ. Otherwise, D uses its own decryption oracle to decrypt e and answers the query by the answer m. (Of course, D actually asks a single parallel query with the ciphertexts e for all components.)

It is easily seen that D simulates the view D' would have in a direct interaction with the game for Π' unless D' asks a ciphertext that decrypts to ρ. This event occurs with probability at most $p \cdot |\mathcal{M}| = p \cdot 2^{-\lambda}$. The lemma now follows using a simple triangle inequality. □

4.2.2 IND-SDA Does Not Imply NM-CPA

The modified scheme. Let λ be the security parameter and $\Pi = (KG, E, D)$ be an IND-SDA-secure PKE scheme with message space $\mathcal{M} = \{0,1\}^\lambda$. Moreover, let $(\mathrm{Enc}, \mathrm{Dec})$ be a (k, λ)-coding scheme with τ-secrecy for some constant $\tau > 0$ and some $k > 0$. Consider the following modification $\Pi'' = (KG'', E'', D'')$ of Π (cf. Figure 5):

- The key generation algorithm KG'' is the same as KG.
- The encryption algorithm E'' works as follows: To encrypt a message $m \in \{0,1\}^k$, it computes $c \leftarrow \mathrm{Enc}(m)$ and $e \leftarrow E_{\mathsf{pk}}(c)$ and outputs $e'' \leftarrow 0\|0\|0^\nu\|e$, where $\nu := \lceil \log \lambda \rceil$.
- The decryption algorithm D'' proceeds as follows, upon receiving a ciphertext $e'' = \beta\|d\|i\|e$. If $\beta = 0$, $d = 0$, and $i = 0^\nu$, it decrypts $c \leftarrow D_{\mathsf{sk}}(e)$, computes $m \leftarrow \mathrm{Dec}(c)$, and outputs m. If $\beta = 1$ and $c[i] = d$ (i.e., if d is a correct guess for the i^{th} bit of the encoding), D'' outputs 0^k. In all other cases, it outputs $\perp$.[9]

[9]Note that in general, not all ν-bit strings i are valid indices. If the decryption algorithm encounters an invalid index, it also outputs $\perp$. For readability this issue is ignored in the remainder of this section.

PKE Scheme $\Pi'' = (KG'', E'', D'')$

Key Generation KG''
- $(\mathsf{pk}, \mathsf{sk}) \leftarrow KG$
- **return** $(\mathsf{pk}, \mathsf{sk})$

Encryption $E''_{\mathsf{pk}}(m)$
- $c \leftarrow \mathrm{Enc}(m)$
- $e \leftarrow E_{\mathsf{pk}}(c)$
- $e'' \leftarrow 0 \,\|\, 0 \,\|\, 0^\nu \,\|\, e$
- **return** e''

Decryption $D''_{\mathsf{sk}}(e'')$
- $\beta \,\|\, d \,\|\, i \,\|\, e \leftarrow e''$
- $c \leftarrow D_{\mathsf{sk}}(e)$
- $m \leftarrow \mathrm{Dec}(c)$
- **if** $\beta = 0$
  - **if** $(d = 0) \wedge (i = 0^\nu)$
    - **return** m
  - **else**
    - **return** $\perp$
- **else**
  - **if** $(c[i] = d)$
    - **return** 0^k
  - **else**
    - **return** $\perp$

Figure 5: *PKE scheme Π'' based on an IND-SDA-secure PKE scheme Π.*

Security of the modified scheme. PKE scheme Π'' is not NM-CPA-secure: A distinguisher can recover each bit $i \in [n]$ of the encoding c^* encrypted in the challenge ciphertext $0\|0\|0^\nu\|e^*$ by a single parallel query containing ciphertexts

$$e^{(i)} := 1 \,\|\, 0 \,\|\, i \,\|\, e^*.$$

If the answer to the i^{th} query is 0^k, then $c^*[i] = 0$; otherwise $c^*[i] = 1$. Computing $\mathrm{Dec}(c^*)$ yields the plaintext encrypted by the challenge.

The modified scheme is, however, still IND-SDA-secure as implied by the following lemma:

Lemma 8. *For all $q \in \mathbb{N}$, and all distinguishers D'', there exist distinguishers D_0 and D_1 such that*

$$\Delta^{D''}(G_0^{\Pi'',q,1}, G_1^{\Pi'',q,1}) \leq \Delta^{D_0}(G_0^{\Pi,q,1}, G_1^{\Pi,q,1}) + \Delta^{D_1}(G_0^{\Pi,q,1}, G_1^{\Pi,q,1}) + 2^{-\tau\lambda}.$$

Proof. Let $b \in \{0,1\}$ and consider the hybrid game H_b that works exactly as $G_b^{\Pi,q,1}$ except that:

- The ciphertext e^* in the challenge ciphertext $0\|0\|0^\nu\|e^*$ is computed as the encryption of a random λ-bit string ρ (instead of an encoding of m_b);
- Decryption queries of the form $1\|d\|i\|e^*$ are answered based on an internally generated encoding $c_b = \mathrm{Enc}(m_b)$, i.e., the answer is 0^k if $c_b[i] = d$ and $\perp$ otherwise.

D_b internally runs a copy of D'' and proceeds as follows: Initially, it obtains a public key pk from its oracle which it forwards to D''. When $(\mathsf{chall}, m_0, m_1)$ is received from D'', D_b chooses a random λ-bit string ρ, computes $c_b \leftarrow \mathrm{Enc}(m_b)$ and outputs $(\mathsf{chall}, c_b, \rho)$ to its oracle. Subsequently, it obtains a ciphertext e^* and outputs $0\|0\|0^\nu\|e^*$ to D''. D_b answers decryption queries $\beta\|d\|i\|e$ made by D'' as follows (implementing the self-destruct mode if the answer is $\perp$):

- If $\beta = 0$, $d = 0$, and $i = 0^\nu$, D_b proceeds as follows: If $e = e^*$, the answer to the query is test. Otherwise, it outputs e to its own decryption oracle. The value c subsequently received is decoded to $m \leftarrow \mathrm{Dec}(c)$ and output. If $\beta = 0$ but $d \neq 0$ or $i \neq 0^\nu$, D_b responds with $\perp$.
- If $\beta = 1$ and $e = e^*$, D_b outputs 0^k if $c_b[i] = d$ and $\perp$ otherwise.
- If $\beta = 1$ and $e \neq e^*$, D_b outputs e to its own decryption oracle and subsequently obtains a value c. D_b outputs 0^k if $c[i] = d$ and $\perp$ otherwise.

By inspection, one verifies that for $b \in \{0,1\}$:

- If D_b interacts with $G_0^{\Pi,q,1}$, then the view of D'' is the view it would have when interacting with $G_b^{\Pi'',q,1}$;
- If D_b interacts with $G_1^{\Pi,q,1}$, then the view of D'' is the view it would have when interacting with H_b.

Moreover, observe that that the hybrids H_b do not leak any information about c_b except when answering decryption queries with $\beta = 1$ and $e = e^*$. Due to the $\tau\lambda$-secrecy of the coding scheme $(\mathrm{Enc}, \mathrm{Dec})$, H_0 and H_1 can only be told apart if a distinguisher D manages to guess $\tau\lambda$ random bits of the encoding. Thus, $\Delta^D(H_0, H_1) \leq 2^{-\tau\lambda}$.

The lemma now follows using a simple triangle inequality. □

5 Non-Malleable Codes

We start by defining non-malleable codes with secret state, in Section 5.1. Our information-theoretic constructions of continuously non-malleable codes for the case of non-parallel and parallel tampering appear in Section 5.2 and 5.3, respectively. Finally, in Section 5.4, we show that the concept of secret state is inherent for achieving non-malleability against parallel tampering.

5.1 Stateful and Stateless Codes

Non-malleable codes were introduced by Dziembowski *et al.* [40]. Intuitively, they protect encoded messages in such a way that any tampering with the codeword causes the decoding to either output the original message or a completely unrelated value.

Definition 8 (Code with secret state)**.** *A* (k, n)-code with secret state (CSS) *is a triple of algorithms* $(\mathrm{Gen}, \mathrm{Enc}, \mathrm{Dec})$*, where the (randomized)* state-generation algorithm Gen *outputs a* secret state s *from some set* $\mathcal{S}$*, the (randomized)* encoding algorithm Enc *takes a* k*-bit plaintext* x *and outputs an* n*-bit encoding* $c \leftarrow \mathrm{Enc}(x)$*, and the (deterministic)* decoding algorithm Dec *takes an encoding as well as some secret state* $s \in \mathcal{S}$ *and outputs a plaintext* $x \leftarrow \mathrm{Dec}(c, s)$ *or the special symbol* $\perp$*, indicating an* invalid encoding*.*

Note that the secret state is generated once and for all, and can be potentially used to decode multiple codewords. In this sense, CSSs are different from codes with randomized decoding [10], where decoding multiple codewords requires fresh and independent randomness. CSSs are also different from codes in the common reference string (CRS) model, where a *public* CRS is sampled once and for all and given as input to both the encoding and decoding algorithm.

Some of the codes in this work do not need to make use of the secret state; they are fully specified by the algorithms Enc and Dec, and the latter does not take any secret state as input. Sometimes we refer to such codes as *stateless*; see Definition 1.

Tampering attacks are captured by functions f, from a certain function class $\mathcal{F}$, that are applied to an encoding. The original definition by [40] allows an attacker to apply only a single tamper function. This notion was later generalized in [44] to capture continuous attacks, where the adversary can tamper many times with the same target encoding until a tamper query results in an invalid codeword.[10]

In addition to continuous non-malleability, this work considers yet another extension, called continuos *parallel* non-malleability: The attacker may repeatedly specify parallel tamper queries, consisting of multiple tampering functions f. The self-destruct occurs as soon as a single component of a parallel query results in an invalid codeword, but the entire query containing the invalid tampering is answered fully. In order to capture continuous parallel attacks, the definition below permits the attacker to repeatedly specify parallel tamper queries, each consisting of several tamper functions. The process ends as soon as one of the tamper queries leads to an invalid codeword.

[10] Continuous non-malleability is known to be impossible without such a self-destruct feature [47, 44, 29].

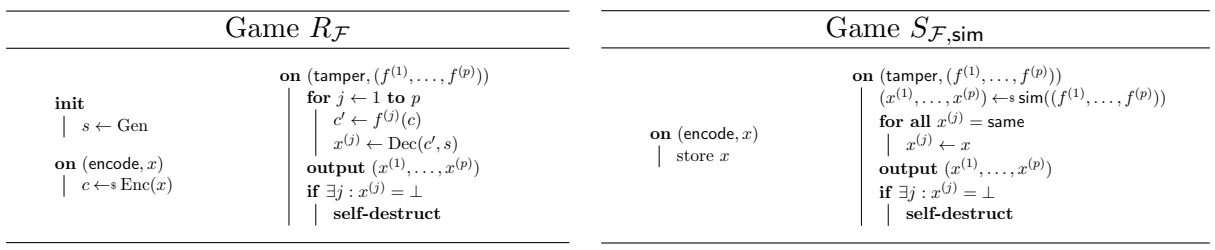

Figure 6: *Distinguishing game* $(R_{\mathcal{F}}, S_{\mathcal{F},\mathsf{sim}})$ *used to define non-malleability of a secret-state coding scheme* $(\mathrm{Gen}, \mathrm{Enc}, \mathrm{Dec})$. *The command* **self-destruct** *has the effect that all* future *queries are answered by* $\perp$.

The non-malleability requirement is captured by considering a real and an ideal experiment. In both experiments, an attacker is allowed to encode a message of his choice. In the real experiment, he may tamper with an actual encoding of that message, whereas in the ideal experiment, the tamper queries are answered by a (stateful) simulator. The simulator is allowed to output the special symbol same, which the experiment replaces by the originally encoded message. In either experiment, if a component of the answer vector to a parallel tamper query is the symbol $\perp$, a self-destruct occurs, i.e., all *future* tamper queries are answered by $\perp$. The experiments are depicted in Figure 6.

Definition 9 (Non-malleable code with secret state)**.** *Let* $q, p \in \mathbb{N}$ *and* $\varepsilon > 0$. *A CSS* $(\mathrm{Gen}, \mathrm{Enc}, \mathsf{D})$ *is* $(\mathcal{F}, q, p, \varepsilon)$-non-malleable *if the following properties are satisfied:*

- Correctness: *For each* $x \in \{0,1\}^k$ *and all* $s \in \mathcal{S}$ *output by* Gen, $\mathsf{D}(\mathrm{Enc}(x), s) = x$ *with probability* 1 *over the randomness of* Enc.
- Non-Malleability: *There exists a (possibly stateful) simulator* sim *such that for any distinguisher* D *asking at most* q *parallel queries, each of size at most* p, $\Delta^D(R_{\mathcal{F}}, S_{\mathcal{F},\mathsf{sim}}) \leq \varepsilon$.

The above definition is similar to the notion of many-many non-malleable codes [19]. The main differences are that Definition 9: (i) is specifically tailored to codes with secret state; and (ii) includes the self-destruct feature. It is well-known that, already in case of bit-wise tampering, assumption (ii) is necessary when considering an arbitrary number of tampering queries (i.e., $q = *$).

It is also easy to adapt Definition 9 to codes without secret state (as the ones considered in [40]). Note that in such a case one obtains the standard notion of non-malleability [40] by setting $q = p = 1$, and continuous non-malleability [44] by letting $p = 1$ and q arbitrary (i.e., $q = *$).

5.2 Non-Malleability under Continuous Tampering

It turns out that a LEDSS with a sufficiently large distance ($\delta > 1/4$) is already *continuously* non-malleable against the class $\mathcal{F}_{\mathsf{set}}$, where a function $f \in \mathcal{F}_{\mathsf{set}}$ is characterized by $(f[1], \ldots, f[n])$, such that $f[j] : \{0,1\} \to \{0,1\}$ is the action of f on the j^{th} bit, for $f[j] \in \{\mathsf{zero}, \mathsf{one}, \mathsf{keep}\}$, with the meaning that it either sets the j^{th} bit to 0 (zero), or to 1 (one), or leaves it unchanged (keep).

Theorem 9. *Let* (E, D) *be a* (k, n, δ, τ)*-LEDSS with* $\delta > 1/4$ *and* $\delta \geq \tau$.[11] *Then, for any* $q \in \mathbb{N}$, (E, D) *is* $(\mathcal{F}_{\mathsf{set}}, q, 1, \varepsilon)$*-non-malleable for*

$$\varepsilon = 2^{-(\tau n - 1)} + \left(\frac{\tau}{(\delta - 1/4)^2}\right)^{\tau n/2}.$$

[11] Note that the requirement $\delta \geq \tau$ can always be achieved by "ignoring" some of the secrecy.

5.2.1 Security Proof (of Theorem 9)

In the remainder of this section, let $\mathcal{F} := \mathcal{F}_{\mathsf{set}}$. For the proof of Theorem 9, fix an arbitrary distinguisher D and let sim be a simulator determined later. The theorem is proved conditioned on the message x encoded by D.

Tamper-query types. Define $A(f)$ to be the set of all indices j such that $f[j] \in \{\mathsf{zero}, \mathsf{one}\}$, and let $q(f) := |A(f)|$; define $B(f)$ to be the set of the indices not in $A(f)$. Moreover, let $\mathsf{val}(\mathsf{zero}) := \mathsf{val}(\mathsf{keep}) := 0$ and $\mathsf{val}(\mathsf{one}) := 1$. In the following, queries $f \in \mathcal{F}$ with $0 \leq q(f) \leq \tau n$, $\tau n < q(f) < (1-\tau)n$, and $(1-\tau)n \leq q(f) \leq n$ are called *low queries*, *middle queries*, and *high queries*, respectively.

On a high level, the proof proceeds as follows: First, one shows that *middle* queries are rejected with high probability. Then, one proves that issuing *low* and *high* queries actually corresponds to guessing bits of the encoding that is being tampered with. Using the secrecy property of the LEDSS, one can show that only with negligible probability, some attacker can guess sufficiently many of those bits before the self-destruct in order to be able to distinguish tampering with an actual encoding from tampering with uniformly random bits, which leads to a simulation strategy.

Analyzing low and high queries. Consider the game $R_{\mathcal{F}}$ and let $c = c[1] \cdots c[n] = \mathsf{E}(x; r)$ be the encoding of the message x initially specified by D, where r are the random bits used by E. Moreover, for a query f, let $\tilde{c} = \tilde{c}[1] \cdots \tilde{c}[n] = f(\mathsf{E}(x; r))$ be the tampered encoding. By the linearity of the LEDSS,

$$\mathsf{D}(\tilde{c}) = \mathsf{D}(c) + \mathsf{D}(d),$$

where $d = \tilde{c} - c$.

- Consider a *low query* f. It fully determines the bits $i \in B(f)$ of d; namely, $d[i] = \mathsf{val}(f[i])$. Let d^* be a codeword such that $d^*[i] = \mathsf{val}(f[i])$ for all $i \in B(f)$. Due to the fact that the LEDSS has distance $\delta \geq \tau$ and $|B(f)| \geq (1-\tau)n$, d^* is unique (and determined solely by f).

  Therefore, $\mathsf{D}(\tilde{c}) \neq \perp$ if and only if for all $i \in A(f)$, $d[i] = d^*[i]$ or, equivalently, $\mathsf{val}(f[i]) - c[i] = d^*[i]$.

- Consider a *high query* f. It fully determines the bits $i \in A(f)$ of $\tilde{c}$; namely, $\tilde{c}[i] = \mathsf{val}(f[i])$. Let $\tilde{c}^*$ be a codeword such that $\tilde{c}^*[i] = \mathsf{val}(f[i])$ for all $i \in A(f)$. Due to the fact that the LEDSS has distance $\delta \geq \tau$ and $|A(f)| \geq (1-\tau)n$, $\tilde{c}^*$ is unique (and determined solely by f).

  Therefore, $\mathsf{D}(\tilde{c}) \neq \perp$ if and only if for all $i \in B(f)$, $\tilde{c}[i] = \tilde{c}^*[i]$ or, equivalently, $c[i] + \mathsf{val}(f[i]) = \tilde{c}^*[i]$.

Handling middle queries. Consider the hybrid game H that proceeds as $R_{\mathcal{F}}$ except that as soon as D specifies a middle query, it outputs $\perp$ and self-destructs.

Lemma 10. $\Delta^D(R_{\mathcal{F}}, H) \leq 2^{-\tau n} + \left(\frac{\tau}{(\delta - 1/4)^2}\right)^{\tau n/2}$.

Proof. The proof uses the self-destruct lemma (cf. Lemma 6 in Section 3).[12] Note that both $R_{\mathcal{F}}$ and H answer queries from $\mathcal{X} := \mathcal{F}$ by values from $\mathcal{Y} := \{0,1\}^k \cup \{\perp\}$. Moreover, observe that their internal randomness is an element uniformly chosen from the space $\mathcal{R}$ of random strings r for the encoding algorithm E.

Let $g : \mathcal{X} \times \mathcal{R} \to \mathcal{Y}$ be the function according to which $R_{\mathcal{F}}$ answers queries, i.e.,

$$g(f, r) := \mathsf{D}(f(\mathsf{E}(x; r))).$$

Hence, $R_{\mathcal{F}}$ is a PSSD game and H is its $\mathcal{B}$-bending (cf. Definition 6) where $\mathcal{B} \subseteq \mathcal{F}$ is the set of middle queries. Moreover, given the above it is easy to see that queries $f \notin \mathcal{B}$, i.e., low and high queries, can only be answered by a unique value y_f or $\perp$. For

[12] Note that Lemma 6 is stated for games that accept *parallel* queries. This is not needed here, i.e., $p = 1$ in the statement of the lemma.

Wrapper $W(\cdot)$

init
- $\forall i \in [n] : c[i] \leftarrow \varnothing$

on first (encode, x)
- output x internally

on (tamper, f) *with* $0 \le q(f) \le \tau n$
- **if** $\exists$*codeword* d^*: $\forall i \in B(f) : \mathsf{val}(f[i]) = d^*[i]$
  - **for** i *where* $f[i] \in A(f)$
    - $g \leftarrow \mathsf{val}(f[i]) - d^*[i]$
    - **if** $c[i] = \varnothing$
      - **output** (i, g) internally
      - **get** $a \in \{\perp, 1\}$
      - **if** $a = \perp$
        - **self-destruct**
      - $c[i] \leftarrow g$
    - **else**
      - **if** $c[i] \neq g$
        - **self-destruct**
  - **if** $\mathrm{Dec}(d^*) = \perp$
    - **self-destruct**
  - **else**
    - **output** $x + \mathrm{Dec}(d^*)$
- **else**
  - **self-destruct**

on (tamper, f) *with* $\tau n < q(f) < n - \tau n$
- **self-destruct**

on (tamper, f) *with* $n - \tau n \le q(f) \le n$
- **if** $\exists$*codeword* $\tilde{c}^* : \forall i \in A(f) : \mathsf{val}(f[i]) = \tilde{c}^*[i]$
  - **for** i *where* $f[i] \in B(f)$
    - $g \leftarrow \tilde{c}^*[i] - \mathsf{val}(f[i])$
    - **if** $c[i] = \varnothing$
      - **output** (i, g) internally
      - **get** $a \in \{\perp, 1\}$
      - **if** $a = \perp$
        - **self-destruct**
      - $c[i] \leftarrow g$
    - **else**
      - **if** $c[i] \neq g$
        - **self-destruct**
  - **if** $\mathrm{Dec}(\tilde{c}^*) = \perp$
    - **self-destruct**
  - **else**
    - **output** $\mathrm{Dec}(\tilde{c}^*)$
- **else**
  - **self-destruct**

Figure 7: *The wrapper $W(\cdot)$. The command* **self-destruct** *causes $W(\cdot)$ to output $\perp$ at B and to halt.*

- *low queries* that value is $y_f := x + \mathsf{D}(d^*)$ and for
- *high queries* that value is $y_f := \mathsf{D}(\tilde{c}^*)$.

Finally, note that by the original analysis of middle queries f in [40],

$$\mathsf{P}[\mathsf{D}(f(\mathsf{E}(x; r))) \neq \perp] \le \left(\frac{\tau}{(\delta - 1/4)^2}\right)^{\tau n/2}.$$

□

Bit-guessing. Consider the hybrid game H. Making tamper queries to this system essentially amounts to trying to "guess" the bits of the encoding $\mathsf{E}(x)$ with the caveat that an incorrect guess leads to the self-destruct. This intuition is formalized by defining a core game B capturing the bit-guessing and a wrapper $W(\cdot)$ such that $W(B)$ and H behave identically.

The core game B works as follows: Initially, it takes a value $x \in \{0,1\}^k$, computes an encoding $c[1] \cdots c[n] \leftarrow \mathsf{E}(x)$ of it, and outputs nothing. Then, it repeatedly accepts guesses $g_i = (j, b)$, where (j, b) is a guess b for c_j. If a guess g_i is correct, B returns $a_i = 1$. Otherwise, it outputs $a_i = \perp$ and self-destructs (i.e., all future answers are $\perp$).

The wrapper $W(\cdot)$ (cf. Figure 7) initially forwards the message x the distinguisher wishes to encode to B, which internally creates an encoding $c[1] \cdots c[n]$ of x. Then, $W(\cdot)$ deals with tampering queries as follows:

- A *low* query f results in $x + \mathsf{D}(d^*)$ if $c[i] = \mathsf{val}(f[i]) - d^*[i]$ for all $i \in A(f)$.
- A *middle* query f results in $\perp$.

- A *high* query f results in $\mathsf{D}(\tilde{c}^*)$ if $c[i] = \tilde{c}^*[i] - \mathsf{val}(f[i])$ for all $i \in B(f)$.

Hence, upon receiving a low or a high query, $W(\cdot)$ issues the corresponding guesses to B. If all guesses succeed, $W(\cdot)$ outputs $x + \mathsf{D}(d^*)$ resp. $\mathsf{D}(\tilde{c}^*)$. Otherwise, it outputs $\perp$ and self-destructs.

Lemma 11. $\Delta^D(H, W(B)) = 0$.

Proof. By inspecting Figure 7, one can see that the wrapper is implemented exactly along the lines argued above, and therefore $W(B)$ perfectly simulates H. □

Simulation. Consider the core game B' that behaves as B except that the initial input x is ignored and the values $c_1, \ldots, c_n$ are chosen uniformly at random and independently.

Lemma 12. $\Delta^D(B, B') \leq 2^{-\tau n}$.

Proof. For both random experiments defined by the interaction of D with B and B', respectively, define the event that D guesses *more* than τn bits correctly. Until this event occurs, both B and B' answer guesses according to bits c_i chosen uniformly at random and independently. Therefore, the distinguishing advantage is bounded by the probability $2^{-\tau n}$ that D provokes this event (in either of the experiments). □

Consider now the game $W(B')$. Due to the nature of B', the behavior of $W(B')$ is independent of the value x that is initially encoded. This allows to easily design a simulator sim such that $W(B')$ and $S_{\mathcal{F},\mathsf{sim}}$ behave identically. It internally creates a simulated encoding consisting of uniformly random bits (just as $W(B')$) and then follows the intuition above. The simulator is described in Figure 8. By inspection, one easily verifies:

Lemma 13. $\Delta^D(W(B'), S_{\mathcal{F},\mathsf{sim}}) = 0$.

The proof of Theorem 9 now follows from a simple triangle inequality.

Proof (of Theorem 9). From Lemmas 10-13, one obtains that for all distinguishers D,

$$\begin{aligned}
&\Delta^D(R_{\mathcal{F}}, S_{\mathcal{F},\mathsf{sim}}) \\
&\leq \Delta^D(R_{\mathcal{F}}, H) + \underbrace{\Delta^D(H, W(B))}_{=0} + \underbrace{\Delta^D(W(B), W(B'))}_{\leq 2^{-\tau n}} + \underbrace{\Delta^D(W(B'), S_{\mathcal{F},\mathsf{sim}})}_{=0} \\
&\leq 2^{-\tau n} + \left(\frac{\tau}{(\delta - 1/4)^2}\right)^{\tau n/2} + 2^{-\tau n} \\
&\leq 2^{-(\tau n - 1)} + \left(\frac{\tau}{(\delta - 1/4)^2}\right)^{\tau n/2}.
\end{aligned}$$

□

5.2.2 Instantiating the Construction

A suitable LEDSS is provided by Dziembowski *et al.* [40] (who consider security against *non-continuous* tampering).

Simulator sim

init
- $\forall i \in [n] : c[i] \leftarrow \{0,1\}$

on (tamper, f) *with* $0 \le q(f) \le \tau n$
- **for** i *where* $f[i] \in A(f)$
  - $d'[i] \leftarrow \mathsf{val}(f[i]) \oplus c[i]$
- **for** i *where* $f[i] \in B(f)$
  - $d'[i] \leftarrow \mathsf{val}(f[i])$
- $d' \leftarrow d'[1] \cdots d'[n]$
- **if** $\mathrm{Dec}(d') = \perp$
  - **output** $\perp$
- **else**
  - **output** same

on (tamper, f) *with* $\tau n < q(f) < (1-\tau)n$
- **output** $\perp$

on (tamper, f) *with* $(1-\tau)n \le q(f) \le n$
- **for** i *where* $f[i] \in A(f)$
  - $c'[i] \leftarrow \mathsf{val}(f[i])$
- **for** i *where* $f[i] \in B(f)$
  - $c'[i] \leftarrow c[i] \oplus \mathsf{val}(f[i])$
- $c' \leftarrow c'[1] \cdots c'[n]$
- **output** $\mathrm{Dec}(c')$

Figure 8: *The simulator* sim.

5.3 Non-Malleability under Continuous Parallel Tampering

Next, we construct a secret-state non-malleable code resilient against continuous parallel tampering attacks from $\mathcal{F}_{\mathsf{set}}$. Later, we will prove that the restriction of the code being stateful is necessary. The intuition behind our construction is the following: If a code has the property (as has been the case with previous schemes secure against (non-parallel) bit-wise tampering) that changing a single bit of a valid encoding results in an invalid codeword, then the tamper function that fixes a particular bit of the encoding and leaves the remaining positions unchanged can be used to determine the value of that bit; this attack is parallelizable, and thus a code of this type cannot provide security against parallel tampering. A similar attack is also possible if the code corrects a fixed (known) number of errors. To circumvent this issue, our construction uses a—for the lack of a better word—"dynamic" error-correction bound: The secret state (which is initially chosen at random) is used to determine the positions of the encoding in which a certain amount of errors is tolerated.

Construction. Let $\mathbb{F} = \mathrm{GF}(2)$ and $\alpha > 0$. Let $(\mathsf{E}, \mathsf{D}, \mathsf{R})$ be a (k, n, δ, τ)-LECSS (cf. Definition 2 in Section 2) with minimum distance δ and secrecy τ over $\mathbb{F}$ such that:[13]

- *Minimum distance:* $\delta > 1/4 + 2\alpha$ and $\delta/2 > 2\alpha$.
- *Constant rate:* $k/n = \Omega(1)$.
- *Constant secrecy:* $\tau = \Omega(1)$.

In the following, we assume that $\alpha \ge \tau$, an assumption that can always be made by ignoring some of the secrecy. Consider the following (k, n)-code with secret state $(\mathrm{Gen}, \mathrm{Enc}, \mathrm{Dec})$:

- Gen: Choose a subset T of $[n]$ of size τn uniformly at random and output it.
- $\mathrm{Enc}(x)$ for $x \in \{0,1\}^k$: Compute $c = \mathsf{E}(x)$ and output it.
- $\mathrm{Dec}(c, T)$ for $c \in \{0,1\}^n$: Find a codeword $w = (w[1], \ldots, w[n])$ with $d_{\mathsf{H}}(w, c) \le \alpha n$, i.e., compute $w \leftarrow \mathsf{R}(c, \alpha n)$. If no such w exists, i.e., $w = \perp$, output $\perp$. Moreover, if $w[j] \neq c[j]$ for some $j \in T$, output $\perp$ as well. Otherwise, decode w to its corresponding plaintext x and output it.

We prove the following theorem:

Theorem 14. *For all* $q, p \in \mathbb{N}$*, the* (k, n)*-code* $(\mathrm{Gen}, \mathrm{Enc}, \mathrm{Dec})$ *based on a* (k, n, δ, τ)*-LECSS satisfying the three conditions above is* $(\mathcal{F}_{\mathsf{set}}, q, p, \varepsilon_{\mathsf{nmc}})$*-non-malleable with*

$$\varepsilon_{\mathsf{nmc}} \quad = \quad p(\mathcal{O}(1) \cdot e^{-\tau n/16} + e^{-\tau^2 n/4}) + p e^{-\tau^2 n}.$$

[13] The reasons for these restrictions become apparent in the proof; of course, α must be chosen small enough in order for these constraints to be satisfiable.

5.3.1 Security Proof

For the proof of Theorem 14, fix $q, p \in \mathbb{N}$ and a distinguisher D making at most q tamper queries of size p each. Set $\mathcal{F} := \mathcal{F}_{\mathsf{set}}$ for the rest of the proof. The goal is to show

$$\Delta^D(R_{\mathcal{F}}, S_{\mathcal{F},\mathsf{sim}}) \quad \leq \quad \varepsilon_{\mathsf{nmc}} \quad = \quad p(\mathcal{O}(1) \cdot e^{-\tau n/16} + e^{-\tau^2 n/4}) + pe^{-\tau^2 n}$$

for a simulator sim to be determined.

On a high level, the proof proceeds as follows: First, it shows that queries that interfere with too many bits of an encoding and at the same time do not fix enough bits (called *middle* queries below) are rejected with high probability. For the remaining query types (called *low* and *high* queries), one can show that their effect on the decoding process can always be determined from the query itself and the bits of the encoding at the positions indexed by the secret trigger set T. Since the size of T is τn, these symbols are uniformly random and independent of the encoded message, which immediately implies a simulation strategy for sim.

Tamper-query types. Recall that $f \in \mathcal{F}_{\mathsf{set}}$ can be characterized by $(f[1], \ldots, f[n])$, where $f[j] : \{0,1\} \to \{0,1\}$ is the action of f on the j^{th} bit, for $f[j] \in \{\mathsf{zero}, \mathsf{one}, \mathsf{keep}\}$, with the meaning that it either sets the j^{th} bit to 0 (zero) or to 1 (one) or leaves it unchanged (keep). Define $A(f)$ to be the set of all indices j such that $f[j] \in \{\mathsf{zero}, \mathsf{one}\}$, and let $q(f) := |A(f)|$. Moreover, let $\mathsf{val}(\mathsf{zero}) := 0$ and $\mathsf{val}(\mathsf{one}) := 1$.

A tamper query f is a *low query* if $q(f) \leq \tau n$, a *middle query* if $\tau n < q(f) < (1-\tau)n$, and a *high query* if $q(f) \geq (1-\tau)n$.

Analyzing query types. The following lemma states that an isolated middle query is rejected with high probability.

Lemma 15. *Let $f \in \mathcal{F}_{\mathsf{set}}$ be a middle query. Then, for any $x \in \{0,1\}^k$,*

$$\mathsf{P}[\mathrm{Dec}(f(\mathrm{Enc}(x))) \neq \perp] \quad \leq \quad \mathcal{O}(1) \cdot e^{-\tau n/16} + e^{-\tau^2 n/4}$$

where the probability is over the randomness of E and the choice of the secret trigger set T.

Proof. Fix $x \in \{0,1\}^k$ and a middle query $f = (f[1], \ldots, f[n])$. Suppose first that $q(f) \geq n/2$. Define

$$\mathcal{W} := \{w \in \mathbb{F}^n \mid w \text{ is codeword} \wedge \exists r : d_{\mathsf{H}}(f(\mathrm{Enc}(x;r)), w) \leq \alpha n\},$$

where r is the randomness of E. That is, $\mathcal{W}$ is the set of all codewords that could possibly be considered while decoding an encoding of x tampered with via f. Consider two distinct codewords $w, w' \in \mathcal{W}$. From the definition of $\mathcal{W}$ it is apparent that $w[j] \neq \mathsf{val}(f[j])$ for at most αn positions $j \in A(f)$ (and similarly for w'), which implies that w and w' differ in at most $2\alpha n$ positions $j \in A(f)$. Therefore, w and w' differ in at least $(\delta - 2\alpha)n$ positions $j \notin A(f)$.

For $w \in \mathcal{W}$, let $\tilde{w}$ be the projection of w onto the unfixed positions $j \notin A(f)$ and set $\tilde{\mathcal{W}} := \{\tilde{w} \mid w \in \mathcal{W}\}$. The above distance argument implies that $|\mathcal{W}| = |\tilde{\mathcal{W}}|$. Moreover, $\tilde{\mathcal{W}}$ is a binary code with block length $n - q(f)$ and relative distance at least

$$\frac{(\delta - 2\alpha)n}{n - q(f)} \geq \frac{(\delta - 2\alpha)n}{n/2} = 2\delta - 4\alpha > 1/2,$$

where the last inequality follows from the fact that δ and α are such that $\delta - 2\alpha > 1/4$. Therefore, by the Plotkin bound (Theorem 5),[14]

$$|\mathcal{W}| = |\tilde{\mathcal{W}}| \leq \mathcal{O}(1).$$

[14]The size constant absorbed by $\mathcal{O}(1)$ here depends on how close $2\delta - 4\alpha$ is to $1/2$.

Denote by $c = (c[1], \ldots, c[n])$ and $\tilde{c} = (\tilde{c}[1], \ldots, \tilde{c}[n])$ the (random variables corresponding to the) encoding $c = \mathsf{E}(x)$ and the tampered encoding $\tilde{c} = f(c)$, respectively. For an arbitrary (n-bit) codeword $w \in \mathcal{W}$,

$$\mathbf{E}[d_{\mathsf{H}}(\tilde{c}, w)] \quad = \quad \sum_{j=1}^{n} \mathbf{E}[d_{\mathsf{H}}(\tilde{c}[j], w[j])] \quad \geq \quad \sum_{j \in J} \mathbf{E}[d_{\mathsf{H}}(\tilde{c}[j], w[j])],$$

where $J \subseteq [n]$ is the set containing the indices of the first τn bits *not* fixed by f. Note that by the definition of middle queries, there are at least that many, i.e., $|J| = \tau n$.

Observe that for $j \in J$, $d_{\mathsf{H}}(\tilde{c}[j], w[j])$ is an indicator variable with expectation $\mathbf{E}[d_{\mathsf{H}}(\tilde{c}[j], w[j])] \geq \frac{1}{2}$, since $c[j]$ is a uniform bit. Thus, $\mathbf{E}[d_{\mathsf{H}}(\tilde{c}, w)] \geq \frac{\tau n}{2}$. Additionally, $(d_{\mathsf{H}}(\tilde{c}[j], w[j]))_{j \in J}$ are independent. Therefore, using a Chernoff bound (Theorem 3), for $\varepsilon > 0$

$$\mathsf{P}[d_{\mathsf{H}}(\tilde{c}, w) \quad < \quad (1 - \varepsilon)\tau n/2] \quad \leq \quad e^{-\tau \varepsilon^2 n/4}.$$

It follows that the probability that there exists $w \in \mathcal{W}$ for which the above does not hold is at most

$$|\mathcal{W}| \cdot e^{-\tau \varepsilon^2 n/4} \quad \leq \quad \mathcal{O}(1) \cdot e^{-\tau \varepsilon^2 n/4},$$

by a union bound. Suppose now that $d_{\mathsf{H}}(\tilde{c}, w) \geq (1 - \varepsilon)\tau n/2$ for all codewords $w \in \mathcal{W}$. Then, over the choice of T, with $|T| = \tau n$,

$$\mathsf{P}[\forall j \in T : d_{\mathsf{H}}(\tilde{c}[j], w[j]) = 0] \quad \leq \quad (1 - (1 - \varepsilon)\tau/2)^{\tau n} \quad \leq \quad e^{-(1-\varepsilon)\tau^2 n/2}.$$

The lemma now follows by setting $\varepsilon := \frac{1}{2}$.

If $q(f) < n/2$ an analogous argument can be made for the difference $d := c - \tilde{c}$ between the encoding and the tampered codeword, as such a query f fixes at least half of the bits of d (to 0, in fact) and $\mathsf{D}(d) \neq \perp$ implies $\mathsf{D}(\tilde{c}) \neq \perp$. □

It turns out that low and high queries always result in $\perp$ or one other value.

Lemma 16. *Low queries $f \in \mathcal{F}_{\mathsf{set}}$ can result only in $\perp$ or the originally encoded message $x \in \{0,1\}^k$. High queries $f \in \mathcal{F}_{\mathsf{set}}$ can result only in $\perp$ or one other value $x_f \in \{0,1\}^k$, which solely depends on f. Furthermore, x_f, if existent, can be found efficiently given f.*

Proof. The statement for low queries is trivial, since a low query f cannot change the encoding beyond the error correction bound αn.

Consider now a high query f and the following efficient procedure:

1. Compute $\tilde{c}_f \leftarrow f(0^n)$.
2. Find a codeword w_f with $d_{\mathsf{H}}(w_f, \tilde{c}_f) \leq 2\alpha n$ (which is possible since $2\alpha < \delta/2$).
3. Output w_f or $\perp$ if none exists.

Consider an arbitrary encoding c and let $\tilde{c} \leftarrow f(c)$ be the tampered encoding. Assume there exists w with $d_{\mathsf{H}}(w, \tilde{c}) \leq \alpha n$. Since a high query f fixes all but τn bits, $d_{\mathsf{H}}(\tilde{c}, \tilde{c}_f) \leq \tau n \leq \alpha n$, and, thus, $d_{\mathsf{H}}(w, \tilde{c}_f) \leq 2\alpha n$, by the triangle inequality. Hence, $w = w_f$.

In other words, if the reconstruction algorithm R on $\tilde{c}$ finds a codeword $w = w_f$ within distance αn, one can find it using the above procedure, which also implies that high queries can only result in $\perp$ or one other message $x_f = \mathsf{R}(w_f, \alpha n)$. □

Handling middle queries. Consider the hybrid game H_1 that behaves as $R_{\mathcal{F}}$, except that it answers all middle queries by $\perp$.

Lemma 17. $\Delta^D(R_{\mathcal{F}}, H_1) \quad \leq \quad p(\mathcal{O}(1) \cdot e^{-\tau n/16} + e^{-\tau^2 n/4})$.

Proof. The lemma is proved using the self-destruct lemma (cf. Lemma 6 in Section 3), conditioned on the message x encoded by D. Note first that both $R_{\mathcal{F}}$ and H_1 answer parallel tamper queries in which each component is from the set $\mathcal{X} := \mathcal{F}$ by vectors whose components are in $\mathcal{Y} := \{0,1\}^k \cup \{\perp\}$. Moreover, both hybrids use as internal randomness a uniformly chosen element from $\mathcal{R} := \{0,1\}^\rho \times \mathcal{S}$, where ρ is an upper bound on the number of random bits used by E and $\mathcal{S}$ is the set of all τn-subsets T of $[n]$. $R_{\mathcal{F}}$ answers each component of a query $f \in \mathcal{X}$ by

$$g(f,(r,T)) \quad := \quad \mathrm{Dec}(f(\mathrm{Enc}(x;r)),T).$$

Define $\mathcal{B} \subseteq \mathcal{X}$ to be the set of all middle queries; H_1 is the $\mathcal{B}$-bending of $R_{\mathcal{F}}$ (cf. Definition 6).

Observe that queries $f \notin \mathcal{B}$ are either low or high queries. For low queries f, the unique answer is $y_f = x$, and for high queries f, $y_f = x_f$ (cf. Lemma 16). Thus, by Lemmas 6 and 15,

$$\Delta^D(R_{\mathcal{F}}, H_1) \quad \leq \quad p \cdot \max_{f \in \mathcal{B}} \mathsf{P}[g(f,(r,T)) \neq \perp] \quad \leq \quad p(\mathcal{O}(1) \cdot e^{-\tau n/16} + e^{-\tau^2 n/4}),$$

where the probability is over the choice of (r,T). □

Handling high queries. Consider the following hybrid game H_2: It differs from H_1 in the way it decodes high queries f. Instead of applying the normal decoding algorithm to the tampered codeword $\tilde{c}$, it proceeds as follows:

1. Find w_f (as in the proof of Lemma 16).
2. If w_f does not exist, return $\perp$.
3. If $\tilde{c}[j] = w_f[j]$ for all $j \in T$, return $\mathsf{D}(w)$. Otherwise, return $\perp$.

Lemma 18. $\Delta^D(H_1, H_2) \quad \leq \quad pe^{-\tau^2 n}$.

Proof. The lemma is proved conditioned on the message x encoded by D *and* the randomness r of the encoding. For the remainder of the proof, r is therefore considered fixed inside H_1 and H_2. The proof, similarly to that of Lemma 17, again uses the self-destruct lemma.

Set $\mathcal{X} := \mathcal{F}$ and $\mathcal{Y} := \{0,1\}^k \cup \{\perp\}$. However, this time, let $\mathcal{R} := \mathcal{S}$. For $f \in \mathcal{X}$ and $T \in \mathcal{R}$, define

$$g(f,T) \quad := \quad \mathrm{Dec}(\tilde{c},T),$$

where $\tilde{c} := f(\mathsf{E}(x;r))$. The bending set $\mathcal{B} \subseteq \mathcal{X}$ is the set of all high queries f such that w_f exists and $d_{\mathsf{H}}(w_f, \tilde{c}) > \alpha n$.[15] It is readily verified that H_2 is a parallel stateless self-destruct game (cf. Definition 5) that behaves according to g, and that H_1 is its $\mathcal{B}$-bending.

Consider a query $f \notin \mathcal{B}$. If f is a low query, the unique answer is $y_f = x$; if it is a middle query, $y_f = \perp$; if it is a high query, $y_f = x_f$ (cf. Lemma 16). Therefore,

$$\Delta^D(H_1, H_2) \quad \leq \quad \max_{f \in \mathcal{B}} \mathsf{P}[g(f,T) \neq \perp] \quad \leq \quad pe^{-\tau^2 n},$$

where the first inequality follows from the self-destruct lemma (Lemma 6) and the second one from the fact that $d_{\mathsf{H}}(x_f, \tilde{c}) > \tau n$ for queries $f \in \mathcal{B}$, and therefore the probability over the choice of T that it is accepted is at most $(1-\tau)^{\tau n} \leq e^{-\tau^2 n}$. □

[15] These are queries potentially accepted by H_2 but not by H_1.

Simulation. By analyzing hybrid H_2, one observes that low and high queries can now be answered knowing only the query itself and the symbols of the encoding indexed by the secret trigger set $T \in \mathcal{S}$.

Lemma 19. *Consider the random experiment of distinguisher D interacting with H_2. There is an efficiently computable function* $\mathrm{Dec}' : \mathcal{F}_{\mathsf{set}} \times \mathcal{S} \times \{0,1\}^{\tau n} \to \{0,1\}^k \cup \{\mathsf{same}, \perp\}$ *such that for any low or high query f, any fixed message x, any fixed encoding c thereof, and any output T of* Gen*,*

$$\left[\mathrm{Dec}'(f, T, (c[j])_{j\in T})\right]_{\mathsf{same}/x} \quad = \quad \mathrm{Dec}(f(c)),$$

where $[\cdot]_{\mathsf{same}/x}$ is the identity function except that same *is replaced by x and where $(c[j])_{j\in T}$ are the symbols of c specified by T.*

Proof. Consider a low query f. Due to the error correction, $\mathrm{Dec}(f(c))$ is the message originally encoded if no bit indexed by T is changed and $\perp$ otherwise. Which one is the case can clearly be efficiently computed from f, T, and $(c[j])_{j\in T}$.

For high queries f the statement follows by inspecting the definition of H_2 and Lemma 16. □

In H_2, by the τn-secrecy of the LECSS, the distribution of the symbols indexed by T is independent of the message x encoded by D. Moreover, the distribution of T is trivially independent of x. This suggests the following simulator sim: Initially, it chooses a random subset T from $\binom{[n]}{\tau n}$ and chooses τn random symbols $(c[j])_{j\in T}$. Every component f of any tamper query is handled as follows: If f is a low or a high query, the answer is $\mathrm{Dec}'(f, T, (c[j])_{j\in T})$; if f is a middle query, the answer is $\perp$. This implies:

Lemma 20. $H_2 \equiv S_{\mathcal{F},\mathsf{sim}}$.

Proof of Theorem 14. Follows from Lemmas 17, 18, and 20 and a triangle inequality. □

5.3.2 Instantiating the Construction

We detail how a LECSS satisfying the properties of Theorem 14 can be constructed by combining high-distance binary codes with a recent result by Cramer *et al.* [30] which essentially allows to "add" secrecy to any code of sufficient rate. The resulting LECSS has secrecy $\tau = \Omega(1)$ and rate $\rho = \Omega(1)$ (cf. Corollary 23 below). The secrecy property depends on the random choice of a universal hash function. Thus, the instantiated code can be seen as a construction in the CRS model.

Let $\mathbb{F} = \mathrm{GF}(2)$ and $\alpha > 0$. We need to construct a (k, n, δ, τ)-LECSS $(\mathsf{E}, \mathsf{D}, \mathsf{R})$ (cf. Definition 2 in Section 2) with minimum distance δ and secrecy τ over $\mathbb{F}$ and the following properties (as required in Section 5.3):

- *Minimum distance:* $\delta > 1/4 + 2\alpha$ and $\delta/2 > 2\alpha$.
- *Constant rate:* $k/n = \Omega(1)$.
- *Constant secrecy:* $\tau = \Omega(1)$.

Let $\mathcal{C}$ be a (n, l)-code with rate $R = \frac{l}{n}$ over $\mathbb{F}$. In the following we write $\mathcal{C}(x)$ for the codeword corresponding to $x \in \mathbb{F}^l$ and $\mathcal{C}^{-1}(c, e)$ for the output of the efficient error-correction algorithm attempting to correct up to e errors on c, provided that $e < \delta n/2$;[16] the output is $\perp$ if there is no codeword within distance e of c.

[16] This assumes that $\mathcal{C}$ is efficiently decodable up to relative distance $\delta/2$. However, while the codes we consider here have this property, for our non-malleable code construction, it would be sufficient to have efficient error correction up to distance 2α for whatever particular choice of the constant α.

Adding secrecy. Let l be such that $k < l < n$. The construction by [30] combines a surjective linear universal hash function $\mathsf{h} : \mathbb{F}^l \to \mathbb{F}^k$ with $\mathcal{C}$ to obtain a LECSS $(\mathsf{E}, \mathsf{D}, \mathsf{R})$ as follows:[17]

- $\mathsf{E}(x)$ for $x \in \{0,1\}^k$: Choose $s \in \{0,1\}^l$ randomly such that $\mathsf{h}(s) = x$ and output $c = \mathcal{C}(s)$.
- $\mathsf{D}(c)$ for $c \in \{0,1\}^n$: Compute $s = \mathcal{C}^{-1}(c, 0)$. If $s = \perp$, output $\perp$. Otherwise, output $x = \mathsf{h}(s)$.
- $\mathsf{R}(c, e)$ for $c \in \{0,1\}^n$ and $e < \delta n/2$: Compute $s = \mathcal{C}^{-1}(c, e)$. If $s = \perp$, output $\perp$. Otherwise, output $x = \mathsf{h}(s)$.

The resulting LECSS has rate $\rho = \frac{k}{ln}$ and retains all distance and error-correction properties of $\mathcal{C}$. Additionally, if R is not too low, the LECSS has secrecy. More precisely, Cramer *et al.* prove the following theorem:

Theorem 21 ([30])**.** *Let $\tau > 0$ and $\eta > 0$ be constants and $\mathcal{H}$ be a family of linear universal hash functions $\mathsf{h} : \mathbb{F}^l \to \mathbb{F}^k$. Given that $R \geq \rho + \eta + \tau + h(\tau)$, there exists a function $\mathsf{h} \in \mathcal{H}$ such that $(\mathsf{E}, \mathsf{D}, \mathsf{R})$ achieves secrecy τ. Moreover, such a function h can be chosen randomly with success probability $1 - 2^{-\eta n}$.*

It should be pointed out that the version of the above theorem in [30] does not claim that any τn bits of an encoding are uniform and independent but merely that they are independent of the message encoded. However, by inspecting their proof, it can be seen that uniformity is guaranteed if $\tau n \leq l - k$, which is the case if and only if $\tau \leq \frac{l}{n} - \frac{k}{n} = R - \rho$, which is clearly implied by the precondition of the theorem.

Zyablov bound. For code $\mathcal{C}$, we use concatenated codes reaching the Zyablov bound:

Theorem 22. *For every $\delta < 1/2$ and all sufficiently large n, there exists a code $\mathcal{C}$ that is*

- *linear,*
- *efficiently encodable,*
- *of distance at least δn,*
- *allows to efficiently correct up to $\delta n/2$ errors,*

and has rate

$$R \geq \max_{0 \leq r \leq 1 - h(\delta + \varepsilon)} r \left(1 - \frac{\delta}{h^{-1}(1-r) - \varepsilon}\right),$$

for $\varepsilon > 0$ and where $h(\cdot)$ is the binary entropy function.

The Zyablov bound is achieved by concatenating Reed-Solomon codes with linear codes reaching the Gilbert-Varshamaov bound (which can be found by brute-force search in this case). Alternatively, Shen [74] showed that the bound is also reached by an explicit construction using algebraic geometric codes.

Choice of parameters. Set $\alpha := 1/200$ and $\delta := 1/4 + 2\alpha + \varepsilon$ for $\varepsilon := 1/500$, say. Then, $\delta - 2\alpha > 1/4$, as required. Moreover, the rate of the Zyablov code with said distance δ can be approximated to be $R \geq 0.0175$. Setting, $\tau := 1/1000$ yields $\tau + h(\tau) \leq 0.0125$, leaving a possible rate for the LECSS of up to $\rho \approx 0.005 - \eta$. Hence:

Corollary 23. *For any $\alpha > 0$ there exists a (k, n, δ, τ)-LECSS $(\mathsf{E}, \mathsf{D}, \mathsf{R})$ with the following properties:*

- Minimum distance: $\delta > 1/4 + 2\alpha$ *and* $\delta/2 > 2\alpha$.
- Constant rate: $k/n = \Omega(1)$.
- Constant secrecy: $\tau = \Omega(1)$.

[17] Note that we switched the roles of l and k here in order to remain consistent with the notation in this paper.

5.4 Impossibility for Codes without State

This section shows that codes without secret state (as originally defined in [40]) cannot achieve (unconditional) non-malleability against parallel tampering. Specifically, the following theorem is proved:

Theorem 24. *Let* $\mathcal{F} := \mathcal{F}_{\mathsf{set}}$. *Let* $(\mathrm{Enc}, \mathrm{Dec})$ *be a* (k, n)*-code without secret state and noticeable rate. There exists a distinguisher* D *asking a single parallel tampering query of size* n^6 *such that, for all simulators* sim *and all* n *large enough,* $\Delta^D(R_{\mathcal{F}}, S_{\mathcal{F},\mathsf{sim}}) \geq 1/2$.

The above impossibility result requires that the rate of the code be sufficiently large ($n = o(2^{k/6})$ suffices, see below for the exact parameters). The distinguisher D is inefficient, so it might still be possible to construct a non-malleable code against parallel tampering with only computational security. This is left as an interesting open problem.

5.4.1 Perfect Correctness

It is instructive to first consider the case where Dec is deterministic and has perfect correctness. The main idea is to define an extraction algorithm that (almost) always succeeds in extracting the encoded message when it interacts with $R_{\mathcal{F}}$ but only does so with a small probability when interacting with $S_{\mathcal{F},\mathsf{sim}}$ (for any sim).

A position $i \in [n]$ is *relevant* if there exists a pair of codewords (c'_i, c''_i), differing only at position i, for which decoding c'_i and c''_i leads to different values. Clearly, in order to decode any codeword $c \in \{0,1\}^n$, one needs to know only the values $c[i]$ for the relevant positions i; all other values play no role in decoding a codeword.

Consider now the following distinguisher D that is given a pair (c'_i, c''_i) (as above) for each *relevant* position $i \in [n]$. D encodes a value x, which defines a target encoding c. Then, D attempts to extract the i^{th} *relevant* bit of c via a tampering query $f_i \in \mathcal{F}$ that keeps the bit in position i and replaces all other values with the bits of c'_i (or, equivalently, c''_i). Since c'_i and c''_i decode to different values, D can determine with a single tampering query (of size at most n) all relevant values $c[i]$ with certainty. Distinguisher D outputs 1 if and only if the above extraction procedure leads to the chosen value x. Clearly, D always outputs 1 when interacting with $R_{\mathcal{F}}$. On the other hand, one can show that D almost never outputs 1 when interacting with $S_{\mathcal{F},\mathsf{sim}}$, which concludes the proof.

5.4.2 The General Case

For the general case, assume that Dec is probabilistic and let ν be the correctness error of the coding scheme, i.e.,

$$\mathsf{P}[\mathrm{Dec}(\mathrm{Enc}(x)) = x] \;\geq\; 1 - \nu$$

for all messages x, where the probability is over the coins of *both* Enc and Dec. Define a position $i \in [n]$ as μ*-relevant* if there exist two codewords $c'_i, c''_i \in \{0,1\}^n$ (i.e., in the range of Enc) differing *exactly* in position i such that

$$\Delta(\mathrm{Dec}(c'_i), \mathrm{Dec}(c''_i)) \;\geq\; \mu.$$

Let Enc_μ be the encoding algorithm obtained from Enc by setting all output positions that are not μ-relevant to 0.

Lemma 25. *If* $(\mathrm{Enc}, \mathrm{Dec})$ *has correctness error* ν*, then* $(\mathrm{Enc}_\mu, \mathrm{Dec})$ *has correctness error* $\nu' = \nu + n\mu$.

Proof. Fix r and x and let $c = \mathrm{Enc}(x; r)$ as well as $c_\mu = \mathrm{Enc}_\mu(x; r)$. By the triangle inequality,

$$\Delta(\mathrm{Dec}(c), \mathrm{Dec}(c_\mu)) \;\leq\; n\mu,$$

for there are at most n non-relevant positions. Note that this inequality also holds if (additionally) r is chosen randomly. Consequently,

$$\mathsf{P}[\mathrm{Dec}(\mathrm{Enc}_\mu(x)) \neq x] \leq \mathsf{P}[\mathrm{Dec}(\mathrm{Enc}(x)) \neq x] + n\mu \leq \nu + n\mu.$$

□

The distinguisher. Let $\rho \in \mathbb{N}$. For each μ-relevant position $i = 1, \ldots, n$, let A_i be an optimal distinguisher for the ρ-fold independent repetitions of $\mathrm{Dec}(c'_i)$ and $\mathrm{Dec}(c''_i)$, where A_i "indicates" ρ-fold c'_i by outputting $c'_i[i]$ (and similarly ρ-fold c''_i by outputting $c''_i[i]$).[18] Consider now the following distinguisher D:

1. Choose $x \leftarrow \{0,1\}^k$ uniformly at random and have it encoded.
2. For each μ-relevant position $i = 1, \ldots, n$, let $f_i = (f_i[1], \ldots, f_i[n])$ where for $j \neq i$

$$f_i[j] = \begin{cases} \mathsf{zero} & \text{if } c'_i[j] = 0, \\ \mathsf{one} & \text{if } c'_i[j] = 1, \end{cases}$$

   and where $f_i[i] = \mathsf{keep}$.
3. Ask the parallel tamper query consisting of ρ copies of each function f_i. For $l = 1, \ldots, \rho$, denote by x'_{il} the answer corresponding to the l^{th} copy of function f_i.
4. For each μ-relevant position $i = 1, \ldots, n$, compute $\bar{c}[i] \leftarrow A_i(x'_{i1}, \ldots, x'_{i\rho})$. For the remaining positions i, set $\bar{c}[i] \leftarrow 0$.
5. Output 1 if $\mathrm{Dec}(\bar{c}[1] \cdots \bar{c}[n]) = x$ and if $x'_{il} \neq x$ for all i, l.[19] Output 0 otherwise.

Real experiment. Consider the interaction of D with the real experiment $R_{\mathcal{F}}$ for $(\mathrm{Enc}, \mathrm{Dec})$. Fix a relevant position i and let $c[i]$ be the corresponding bit of the encoding of $\mathrm{Enc}(x)$. Since $\Delta(\mathrm{Dec}(c'_i), \mathrm{Dec}(c''_i)) \geq \mu$, by virtue of Proposition 4, their ρ-fold independent repetitions have distance at least $1 - 2e^{-\rho\mu^2/2}$, which implies that A_i guesses $c[i]$ incorrectly with probability at most $e^{-\rho\mu^2/2}$. By a union bound over all of the at most n μ-relevant positions i, all bits $c[i]$ are guessed correctly except with probability at most $ne^{-\rho\mu^2/2}$.

Furthermore, the probability that $x'_{il} = x$ for some i, l is bounded by $n\rho 2^{-(k-1)}$ since each query f_i overrides all but a single bit of the encoding.

Finally, using the correctness of Enc_μ (Lemma 25), the probability that D outputs 1 when interacting with $R_{\mathcal{F}}$ is at least $1 - (\nu + n\mu + n\rho 2^{-(k-1)} + ne^{-\rho\mu^2/2})$.

Ideal experiment. Let sim be an arbitrary simulator and consider the interaction of D and $S_{\mathcal{F},\mathsf{sim}}$. Note that if sim outputs same, then D outputs 0, since the ideal experiment replaces same by x. If sim does not output same, after step 1, the interaction of D and $S_{\mathcal{F},\mathsf{sim}}$ is independent of x, and hence, the probability that $\mathrm{Dec}(\bar{c}[1] \cdots \bar{c}[n]) = x$ is at most 2^{-k}.

Parameter choices. Summarizing, the advantage of D is at least $1 - (\nu + n\mu + n\rho 2^{-(k-1)} + 2^{-k} + ne^{-\rho\mu^2/2})$ which is at least $1/2$ for large enough n if one sets, e.g., $\mu = n^{-2}$ and $\rho = n^5$ (assuming that ν is negligible).

[18] In other words, A_i outputting $c'_i[i]$ (resp. $c''_i[i]$) is interpreted as A_i "believing" that it is given a sample of ρ-fold independent repetitions of $\mathrm{Dec}(c'_i)$ (resp. $\mathrm{Dec}(c''_i)$).

[19] The latter condition helps in the analysis of the advantage of D.

PKE Scheme $\Pi' = (KG', E', D')$

Key Generation KG'
- **for** $i \leftarrow 1$ **to** n
  - $(\mathsf{pk}_i, \mathsf{sk}_i) \leftarrow\!\!\$\ KG$
- $\mathsf{pk} \leftarrow (\mathsf{pk}_1, \ldots, \mathsf{pk}_n)$
- $\mathsf{sk} \leftarrow (\mathsf{sk}_1, \ldots, \mathsf{sk}_n)$
- $s \leftarrow \mathrm{Gen}$
- **return** $(\mathsf{pk}, (\mathsf{sk}, s))$

Encryption $E'_{\mathsf{pk}}(m)$
- $c = (c[1], \ldots, c[n]) \leftarrow \mathrm{Enc}(m)$
- **for** $i \leftarrow 1$ **to** n
  - $e_i \leftarrow\!\!\$\ E_{\mathsf{pk}_i}(c[i])$
- **return** $e = (e_1, \ldots, e_n)$

Decryption $D'_{(\mathsf{sk},s)}(e)$
- $(e_1, \ldots, e_n) \leftarrow e$
- **for** $i \leftarrow 1$ **to** n
  - $c[i] \leftarrow\!\!\$\ D_{\mathsf{sk}_i}(e_i)$
  - **if** $c[i] = \perp$
    - **return** $\perp$
- $m \leftarrow \mathrm{Dec}(c[1] \cdots c[n], s)$
- **return** m

Figure 9: *The k-bit PKE scheme $\Pi' = (KG', E', D')$ built from a 1-bit PKE scheme $\Pi = (KG, E, D)$ and a (k, n)-coding scheme with secret state* $(\mathrm{Gen}, \mathrm{Enc}, \mathrm{Dec})$.

6 Domain Extension

This section contains one of our main technical results. We show how single-bit NM-SDA PKE can be combined with *secret-state non-malleable codes* resilient against *continuous parallel tampering*, see Section 5.3, to achieve multi-bit NM-SDA-secure PKE. Moreover, the same transformation works also for the weaker notions of NM-CPA and IND-SDA, where in the latter case it suffices to rely on a stateless code. (Whereas for NM-CPA and NM-SDA secret-state non-malleable codes are necessary.)

In Section 6.1, we describe our non-malleable code based domain extender for NM-SDA PKE, and we analyze its security in Section 6.2. Finally, in Section 6.3, we explain how to adapt the analysis to the cases of NM-CPA and IND-SDA.

6.1 Combining Single-Bit PKE and Non-Malleable Codes

Our construction of a multi-bit NM-SDA-secure PKE scheme Π' from a single-bit NM-SDA-secure scheme Π and a secret-state non-malleable (k, n)-code works as follows: It encrypts a k-bit message m by first computing an encoding $c = (c[1], \ldots, c[n])$ of m and then encrypting each bit $c[j]$ under an independent public key of Π; it decrypts by first decrypting the individual components and then decoding the resulting codeword using the secret state of the non-malleable code; the secret state is part of the secret key. The scheme is depicted in detail in Figure 9.

Intuitively, NM-SDA security (or CCA security in general) guarantees that an attacker can either leave a message intact or replace it by an independently created one. For our construction, which separately encrypts every bit of an encoding of the plaintext, this translates to the following capability of an adversary w.r.t. decryption queries: It can either leave a particular bit of the encoding unchanged or fix it to 0 or to 1. Therefore, the tamper class against which the non-malleable code must be resilient is the class $\mathcal{F}_{\mathsf{set}} \subseteq \{f \mid f : \{0,1\}^n \to \{0,1\}^n\}$ of functions that tamper with each bit of an encoding individually and can either leave it unchanged or set it to a fixed value. More formally, $f \in \mathcal{F}_{\mathsf{set}}$ can be characterized by $(f[1], \ldots, f[n])$, where $f[j] : \{0,1\} \to \{0,1\}$ is the action of f on the j^{th} bit and $f[j] \in \{\mathsf{zero}, \mathsf{one}, \mathsf{keep}\}$ with the meaning that it either sets the j^{th} bit to 0 (zero) or to 1 (one) or leaves it unchanged (keep).

Importantly, the PKE Π' of Figure 9 achieves only the so-called *replayable* variant of NM-SDA security. The notion of replayable CCA (RCCA) security (in general) was introduced by Canetti *et al.* [16] to deal with the fact that for many applications (full) CCA security is unnecessarily strict. In particular, RCCA does not rule out attackers able to maul a given ciphertext into a different valid ciphertext, so long as the underlying plaintext does not change.[20] This flavor of security naturally applies to NM-SDA as well.

Among other things, Canetti *et al.* provide a MAC-based generic transformation of RCCA-secure

[20]The intuitive reason why the construction of Figure 9 only achieves replayable security is that the underlying non-malleable code does not rule out the possibility of changing a given codeword into a different valid codeword that encodes the same message.

PKE Scheme $\Pi'' = (KG'', E'', D'')$

Key Generation KG''	Encryption $E''_{\mathsf{pk}}(m)$	Decryption $D''_{\mathsf{sk}}(e)$
$(\mathsf{pk}, \mathsf{sk}) \leftarrow\!\!\$\ KG'$	$K \leftarrow\!\!\$\ \{0,1\}^\lambda$	$(e_1, e_2) \leftarrow e$
return $(\mathsf{pk}, \mathsf{sk})$	$e_1 \leftarrow E'_{\mathsf{pk}}(m \,\|\, K)$	$m \,\|\, K = D'_{\mathsf{sk}}(e_1)$
	$e_2 \leftarrow T_K(e_1)$	**if** $V_K(e_1, e_2) = 0$
	return $e = (e_1, e_2)$	**return** $\perp$
		return m

Figure 10: *The PKE scheme $\Pi'' = (KG'', E'', D'')$ built from a PKE scheme $\Pi' = (KG', E', D')$ and a MAC (T, V).*

schemes into CCA-secure ones, which we can also apply in our setting (as we show) to obtain a fully NM-SDA-secure scheme Π''. Let $\Pi' = (KG', E', D')$ be a PKE scheme and (T, V) be a MAC (cf. Section 2.4). The transformation, which is depicted in Figure 10, yields a new PKE Π'' and roughly works as follows. The key generation remains unchanged. To encrypt a message m, the new encryption algorithm first chooses a key K for the MAC and computes an encryption $e_1 \leftarrow E'_{\mathsf{pk}}(m \,\|\, K)$ and $e_2 \leftarrow T_K(e_1)$; the ciphertext is (e_1, e_2). The new decryption algorithm decrypts e_1 to (m, K) and verifies the tag e_2. If the tag is valid, the decryption algorithm outputs m; otherwise, it outputs $\perp$. Combining the transformations of Figure 9 and Figure 10, we obtain the following theorem.

Theorem 26. *Let $q, p \in \mathbb{N}$ and Π be a $(t + t_{\mathsf{1bit}}, q, p, \varepsilon_{\mathsf{1bit}})$-NM-SDA-secure 1-bit PKE scheme, (T, V) a $(t + t_{\mathsf{mac}}, 1, qp, \varepsilon_{\mathsf{mac}})$-MAC, and $(\mathrm{Gen}, \mathrm{Enc}, \mathrm{Dec})$ a $(\mathcal{F}_{\mathsf{set}}, q, p, \varepsilon_{\mathsf{nmc}})$-non-malleable (k, n)-code with secret state. Then, the PKE scheme Π'' obtained by combining the transformations of Figure 9 and Figure 10 is (t, q, p, ε)-NM-SDA-secure PKE scheme with*

$$\varepsilon \quad = \quad 2(3(n\varepsilon_{\mathsf{1bit}} + \varepsilon_{\mathsf{nmc}}) + qp \cdot 2^{-\ell} + \varepsilon_{\mathsf{mac}}),$$

where t_{1bit} and t_{mac} are the overheads incurred by the corresponding reductions and ℓ is the length of a verification key for the MAC.

6.2 Security Analysis

The proof of Theorem 26 is divided in two parts. First, we prove that the PKE scheme Π' resulting from combining a single-bit PKE Π and a non-malleable code with secret state $(\mathrm{Gen}, \mathrm{Enc}, \mathrm{Dec})$ as shown in Figure 9 is replayable NM-SDA secure (NM-RSDA). Then, we show that a MAC-based transformation suggested by [16] to obtain IND-CCA security from IND-RCCA security also works in our setting, i.e., the transformation of Figure 10 applied to Π' yields a fully NM-SDA secure PKE scheme Π''.

6.2.1 Replayable NM-SDA Security

The notion of *replayable CCA* security was introduced by Canetti *et al.* [16] to deal with the artificial strictness of CCA security. Intuitively, it potentially allows an attacker to maul a target ciphertext into one that decrypts to the *same* message.[21] This idea carries over seamlessly to the definition of NM-SDA security; the corresponding distinguishing game $G_b^{\Pi,\mathsf{nm\text{-}rsda}}$ is obtained by changing $G_b^{\Pi,\mathsf{nm\text{-}sda}}$ (cf. Figure 3) to answer test whenever a ciphertext $e^{(j)}$ decrypts to m_0 or m_1 (instead of only when $e^{(j)}$ equals the challenge ciphertext).

Definition 10. *A PKE scheme Π is* replayable (t, q, p, ε)-NM-SDA-secure (NM-RSDA) *if for all distinguishers D with running time at most t and making at most q decryption queries of size at most p each,*

$$\Delta^D(G_0^{\Pi,\mathsf{nm\text{-}rsda}}, G_1^{\Pi,\mathsf{nm\text{-}rsda}}) \quad \leq \quad \varepsilon.$$

[21] In contrast, full CCA security requires that any ciphertext created by the attacker (other than the target ciphertext) decrypt to an independent message.

6.2.2 Non-Malleable Codes and PKE

In this section we show that the PKE scheme Π' is NM-RSDA if the underlying single-bit scheme Π is NM-SDA secure. Concretely, we prove:

Theorem 27. *Let $q, p \in \mathbb{N}$ and Π be a $(t_{\mathsf{rsda}} + t_{\mathsf{1bit}}, q, p, \varepsilon_{\mathsf{1bit}})$-NM-SDA-secure 1-bit PKE scheme and let $(\mathrm{Gen}, \mathrm{Enc}, \mathrm{Dec})$ be $(\mathcal{F}_{\mathsf{set}}, q, p, \varepsilon_{\mathsf{nmc}})$-non-malleable. Then, Π' is $(t_{\mathsf{rsda}}, q, p, \varepsilon_{\mathsf{rsda}})$-NM-RSDA-secure PKE scheme with*

$$\varepsilon_{\mathsf{rsda}} \quad = \quad 2(n\varepsilon_{\mathsf{1bit}} + \varepsilon_{\mathsf{nmc}}),$$

where t_{1bit}, t_{rsda} represents the overhead incurred by the reductions.

Before coming to the proof of the above theorem, we discuss some intuition. The proof considers a series of n hybrid experiments. In very rough terms, the i^{th} hybrid generates the challenge ciphertext by computing an encoding $c = (c[1], \ldots, c[n])$ of the challenge plaintext and by replacing the first i bits $c[i]$ of c by random values $\tilde{c}[i]$ before encrypting the encoding bit-wise, leading to the challenge $(e_1^*, \ldots, e_n^*)$. Moreover, when answering decryption queries $(e_1', \ldots, e_n')$, if $e_j' = e_j^*$ for $j \leq i$, the i^{th} hybrid sets the outcome of e_j''s decryption to be the corresponding bit $c[j]$ of the original encoding c, whereas if $e_j' \neq e_j^*$, it decrypts normally (then it decodes the resulting n-bit string normally). This follows the above intuition that a CCA-secure PKE scheme guarantees that if a decryption query is different from the challenge ciphertext, then the plaintext contained in it must have been created independently of the challenge plaintext. The indistinguishability of the hybrids follows from the security of the underlying single-bit scheme Π.

In the n^{th} hybrid, the challenge consists of n encryptions of random values. Thus, the only information about the encoding of the challenge plaintext that an attacker gets is that leaked through decryption queries. But in the n^{th} hybrid there is a 1-to-1 correspondence between decryption queries and the tamper function $f = (f[1], \ldots, f[n])$ applied to the encoding of the challenge plaintext: The case $e_j' = e_j^*$ corresponds to $f[j] = \mathsf{keep}$, and the case $e_j' \neq e_j^*$ corresponds to $f[j] = \mathsf{zero}$ or $f[j] = \mathsf{one}$, depending on whether e_j' decrypts to zero or to one. This allows a reduction to the security of the non-malleable code.

Formally, the proof of Theorem 27 follows directly from the following lemma:

Lemma 28. *For $b \in \{0,1\}$ and $i \in [n]$, there exist reductions $R_{b,i}(\cdot)$ and $W_b(\cdot)$ such that for all distinguishers D,*

$$\Delta^D(G_0^{\Pi',\mathsf{nm\text{-}rsda}}, G_1^{\Pi',\mathsf{nm\text{-}rsda}}) \quad \leq \quad \sum_{b,i} \Delta^{D(R_{b,i}(\cdot))}(G_0^{\Pi,\mathsf{nm\text{-}sda}}, G_1^{\Pi,\mathsf{nm\text{-}sda}}) + \sum_{b} \Delta^{D(W_b(\cdot))}(R_{\mathcal{F}}, S_{\mathcal{F},\mathsf{sim}}),$$

where sim is the simulator for the non-malleable code. Moreover, all reductions preserve the number q and the size p of the queries.

Proof (of Theorem 27). Let t_{1bit} be the maximal occurring overhead caused by the reductions $R_{b,i}(\cdot)$. Fix a distinguisher D having running time t_{rsda} and making at most q decryption queries of size at most p. Due to the preservation property of the reductions, $\Delta^{D(R_{b,i}(\cdot))}(G_0^{\Pi,\mathsf{nm\text{-}sda}}, G_1^{\Pi,\mathsf{nm\text{-}sda}}) \leq \varepsilon_{\mathsf{1bit}}$ and $\Delta^{D(W_b(\cdot))}(R_{\mathcal{F}}, S_{\mathcal{F},\mathsf{sim}}) \leq \varepsilon_{\mathsf{nmc}}$, which using Lemma 28 completes the proof. □

Towards a proof of Lemma 28, consider the following hybrids for $b \in \{0,1\}$ and $i \in [n]$: $H_{b,i}$ proceeds as $G_b^{\Pi',\mathsf{nm\text{-}rsda}}$ except that the challenge query $(\mathsf{chall}, m_0, m_1)$ and decryption queries $(\mathsf{dec}, e^{(1)}, \ldots, e^{(p)})$ are handled differently:

- **Challenge query:** The first i bits of the encoding $c = (c[1], \ldots, c[n])$ of m_b are replaced by uniformly random and independent bits. The resulting n-bit string is then encrypted bit-wise (as done by E'). This results in the challenge ciphertext $e^* = (e_1^*, \ldots, e_n^*)$.

- **Decryption query:** Every component $e^{(l)} = (e'_1, \ldots, e'_n)$ is answered as follows: Hybrid $H_{b,i}$ computes $c' = (c'[1], \ldots, c'[n])$, where

$$c'[i] \quad = \quad \begin{cases} c[j] & \text{if } e'_j = e^*_j, \text{ and} \\ D_{\mathsf{sk}_j}(e'_j) & \text{otherwise.} \end{cases}$$

  Then, $H_{b,i}$ outputs $\mathrm{Dec}(c', s)$ as the answer to the component of the decryption query.[22]

Let $H_{b,0} := G_b^{\Pi',\mathsf{nm\text{-}rsda}}$.

Lemma 29. *For all $b \in \{0,1\}$ and $i \in [n]$, there exist a reduction $R_{b,i}(\cdot)$ such that for all D*

$$\Delta^D(H_{b,i-1}, H_{b,i}) \quad = \quad \Delta^{D(R_{b,i}(\cdot))}(G_0^{\Pi,\mathsf{nm\text{-}sda}}, G_1^{\Pi,\mathsf{nm\text{-}sda}}).$$

Proof. Fix b and i. Hybrid $R_{b,i}(\cdot)$ works as follows: Initially, it generates the secret state $s \leftarrow \mathrm{Gen}$ and $n-1$ key pairs $(\mathsf{pk}_j, \mathsf{sk}_j)$ for $j \in [n] \setminus \{i\}$, obtains pk_i (but not sk_i) from the oracle (from $G_0^{\Pi,\mathsf{nm\text{-}sda}}$ or $G_0^{\Pi,\mathsf{nm\text{-}sda}}$), and outputs $\mathsf{pk} := (\mathsf{pk}_1, \ldots, \mathsf{pk}_n)$. When it receives $(\mathsf{chall}, m_0, m_1)$, it computes an encoding $c = (c[1], \ldots, c[n]) \leftarrow \mathrm{Enc}(m_b)$. Then, it chooses i random bits $\tilde{c}[1], \ldots, \tilde{c}[i]$ and computes

$$e^*_j \quad = \quad \begin{cases} E_{\mathsf{pk}_j}(\tilde{c}[j]) & \text{for } j < i, \text{ and} \\ E_{\mathsf{pk}_j}(c[j]) & \text{for } j > i. \end{cases}$$

Moreover, it outputs $(\mathsf{chall}, c[i], \tilde{c}[i])$ to its oracle and obtains a ciphertext e^*_i. It finally returns $e^* = (e^*_1, \ldots, e^*_n)$.

When $R_{b,i}(\cdot)$ receives a (parallel) decryption query, for each component $e' = (e'_1, \ldots, e'_n)$ it proceeds as follows: For $j \neq i$, it computes $c'[j]$ as $H_{b,i}$ does. Moreover, if $e'_i = e^*_i$, it sets $c'[i] \leftarrow c[i]$. Otherwise, it outputs (dec, e'_i) to its oracle and obtains the answer $c'[i]$.[23] Then, it computes $m' \leftarrow \mathrm{Dec}(c')$. The answer to the component of the decryption query is m', unless $m' \in \{m_0, m_1\}$, in which case the it is test. If one of the component answers is $\perp$, $R_{b,i}(\cdot)$ implements the self-destruct mode, i.e., answers all future queries by $\perp$.

Consider $R_{b,i}(G_0^{\Pi,\mathsf{nm\text{-}sda}})$ and $H_{b,i-1}$. Both generate the public key in the same fashion. As to the challenge ciphertext, the first $i-1$ ciphertext components e_j generated by $R_{b,i}(G_0^{\Pi,\mathsf{nm\text{-}sda}})$ are encryptions of random bits $\tilde{c}[j]$, whereas the i^{th} and the remaining components are encryptions of the corresponding bits of an encoding of m_b (generated by $G_0^{\Pi,\mathsf{nm\text{-}sda}}$ and $R_{b,i}(\cdot)$, respectively). The same is true for $H_{b,i-1}$. The answer to a decryption query component sent to $R_{b,i}(G_0^{\Pi,\mathsf{nm\text{-}sda}})$ is $\mathrm{Dec}(c')$ for $c' = (c'[1], \ldots, c'[n])$, where $c'[j] = D_{\mathsf{sk}_j}(e'_j)$ unless $j < i$ and $e'_j = e_j$, in which case $c'[j] = \tilde{c}[j]$. Again, the same holds for $H_{b,i-1}$. Moreover, both $R_{b,i}(G_0^{\Pi,\mathsf{nm\text{-}sda}})$ and $H_{b,i-1}$ answer test if $\mathrm{Dec}(c') \in \{m_0, m_1\}$. Thus, they behave identically.

$R_{b,i}(G_1^{\Pi,\mathsf{nm\text{-}sda}})$ and $H_{b,i}$ are compared similarly. This concludes the proof. □

Lemma 30. *For $b \in \{0,1\}$, there exists a wrapper $W_b(\cdot)$ such that*

- *$W_b(R_{\mathcal{F}})$ behaves as $H_{b,n}$, and*
- *$W_0(S_{\mathcal{F},\mathsf{sim}})$ and $W_1(S_{\mathcal{F},\mathsf{sim}})$ behave identically.*

Proof. Wrapper $W_b(\cdot)$ works as follows: Initially, it generates n key pairs $(\mathsf{pk}_i, \mathsf{sk}_i)$ for $i \in [n]$ and outputs $\mathsf{pk} := (\mathsf{pk}_1, \ldots, \mathsf{pk}_n)$. When it receives $(\mathsf{chall}, m_0, m_1)$, it picks n random values $\tilde{c}[1], \ldots, \tilde{c}[n]$, computes $e^*_i \leftarrow\!\!\$\, E_{\mathsf{pk}}(\tilde{c}[i])$ for $i = 1, \ldots, n$, and returns $e = (e_1, \ldots, e_n)$. Additionally, it outputs (encode, m_b) to its oracle.

[22] Assume here and below that $\mathrm{Dec}(c') = \perp$ if any of the bits $c'[j]$ equal $\perp$.

[23] In fact, it is important that $R_{b,i}(\cdot)$ output a single parallel decryption query containing all e'_i for the individual components; but it is less cumbersome to describe how individual components are handled.

When it gets a (parallel) decryption query, for every component $e' = (e'_1, \ldots, e'_n)$, it proceeds as follows: First, it creates a tamper query $f = (f[1], \ldots, f[n])$ where

$$f[i] \quad = \quad \begin{cases} \mathsf{zero} & \text{if } e'_i \neq e^*_i \text{ and } D_{\mathsf{sk}_i}(e'_i) = 0, \\ \mathsf{one} & \text{if } e'_i \neq e^*_i \text{ and } D_{\mathsf{sk}_i}(e'_i) = 1, \text{ and} \\ \mathsf{keep} & \text{if } e'_i = e^*_i. \end{cases}$$

Then, it outputs (tamper, f) to its oracle and obtains an answer x'. If $x' \in \{m_0, m_1\}$, the answer to the component query test.[24] Otherwise, it is x'. If one of the component answers is $\perp$, $W_b(\cdot)$ implements the self-destruct mode, i.e., answers all future queries by $\perp$.

Consider $W_b(R_{\mathcal{F}})$ and $H_{b,n}$. Both generate the public key in the same fashion. Furthermore, in either case, the challenge ciphertext consists of n encryptions of random bits. Finally, both answer a decryption query by applying the same tamper function to an encoding of m_b before decoding it. When the decoding of the tampered codeword results in m_0 or m_1, both answer test. Therefore, they behave identically.

Due to the fact that test is output when a decryption query results in m_0 or m_1, the observable behavior is the same in $W_0(S_{\mathcal{F},\mathsf{sim}})$ and $W_1(S_{\mathcal{F},\mathsf{sim}})$.[25]

□

Proof (of Lemma 28). Lemma 28 follows using a triangle inequality. Specifically, for any distinguisher D,

$$\begin{aligned} \Delta^D(G_0^{\Pi',\mathsf{nm\text{-}rsda}}, G_1^{\Pi',\mathsf{nm\text{-}rsda}}) \quad &\leq \quad \sum_i \Delta^D(H_{0,i-1}, H_{0,i}) + \Delta^D(W_0(R_{\mathcal{F}}), W_0(S_{\mathcal{F},\mathsf{sim}})) \\ &\quad + \Delta^D(W_1(S_{\mathcal{F},\mathsf{sim}}), W_1(R_{\mathcal{F}})) + \sum_i \Delta^D(H_{1,i-1}, H_{1,i}) \\ &\leq \quad \sum_{b,i} \Delta^D(R_{b,i}(G_0^{\Pi,\mathsf{nm\text{-}sda}}), R_{b,i}(G_1^{\Pi,\mathsf{nm\text{-}sda}})) \\ &\quad + \sum_b \Delta^{D(W_b(\cdot))}(R_{\mathcal{F}}, S_{\mathcal{F},\mathsf{sim}}), \end{aligned}$$

where the last inequality follows from Lemmas 29 and 30. □

6.2.3 From Replayable to Full NM-SDA Security

Next, we analyze the security of the transformation in Figure 10.

Theorem 31. *Let Π' be a $(t+t_{\mathsf{rsda}}, q, p, \varepsilon_{\mathsf{rsda}})$-NM-RSDA secure PKE scheme and (V, T) a $(t+t_{\mathsf{mac}}, \varepsilon_{\mathsf{mac}})$-secure MAC. Then, Π'' is a (t, q, p, ε)-NM-SDA-secure PKE scheme for*

$$\varepsilon \quad \leq \quad 2(\varepsilon_{\mathsf{rsda}} + qp \cdot 2^{-\ell} + \varepsilon_{\mathsf{mac}}) + \varepsilon_{\mathsf{rsda}},$$

where ℓ is the length of the MAC key.

The theorem follows from the following lemma:

Lemma 32. *For $b \in \{0,1\}$ there exist reductions $R_b(\cdot)$, $R'(\cdot)$, and $R''_b(\cdot)$, such that for all distinguishers D,*

$$\begin{aligned} \Delta^D(G_0^{\Pi'',\mathsf{nm\text{-}sda}}, G_1^{\Pi'',\mathsf{nm\text{-}sda}}) \quad &\leq \quad \sum_b \left(\Delta^{D(R_b(\cdot))}(G_b^{\Pi',\mathsf{nm\text{-}rsda}}, G_1^{\Pi',\mathsf{nm\text{-}rsda}}) + qp \cdot 2^{-\ell} + \Gamma^{D(R''_b(\cdot))}(G^{\mathsf{mac}}) \right) \\ &\quad + \Delta^{D(R'(\cdot))}(G_0^{\Pi',\mathsf{nm\text{-}rsda}}, G_1^{\Pi',\mathsf{nm\text{-}rsda}}). \end{aligned}$$

[24] Again, $W_b(\cdot)$ needs to output a single parallel tamper query containing the tamper functions f for the individual components.

[25] This is where the proof reflects that Π' is only NM-RSDA secure.

where ℓ is the length of the MAC key. Moreover, reductions $R_b(\cdot)$ and $R'(\cdot)$ preserve the number q and the size p of the queries, and reduction $R''_b(\cdot)$ asks a single tag query and $q \cdot p$ verification queries.

Proof of Theorem 31. Let t_{rsda} be the maximal occurring overhead caused by the reductions $R_b(\cdot)$, $R'(\cdot)$ and t_{mac} that by the reductions $R''_b(\cdot)$. Fix a distinguisher D having running time t_{rsda} and making at most q decryption queries of size at most p. Due to the preservation properties of the above reductions, the distinguishing advantages on $G_b^{\Pi',\mathsf{nm\text{-}rsda}}$ are at most $\varepsilon_{\mathsf{rsda}}$ and $\Gamma^{D(R''_b(\cdot))}(G^{\mathsf{mac}})$ is at most $\varepsilon_{\mathsf{mac}}$. □

Hybrid 1. The first hybrid H_b captures the fact that the MAC key in the challenge ciphertext is computationally hidden; it differs from $G_b^{\Pi'',\mathsf{nm\text{-}sda}}$ as follows:

- It generates the challenge ciphertext using two independent MAC keys K^* and K, i.e., $(e_1^*, e_2^*) \leftarrow (E'_{\mathsf{pk}}(m_b \,\|\, K^*), T_K(e_1^*))$.
- When answering (components of parallel) decryption queries $(e'_1, e'_2) \leftarrow (E'_{\mathsf{pk}}(m_b \,\|\, K'), e'_2)$, if $K' = K^*$, the tag is verified using K instead of K^*.

Lemma 33. *There exists a reduction $R_b(\cdot)$ such that for all distinguishers D asking at most q parallel queries of size at most p each,*

$$\Delta^D(G_b^{\Pi'',\mathsf{nm\text{-}sda}}, H_b) \quad \leq \quad \Delta^{D(R_b(\cdot))}(G_0^{\Pi',\mathsf{nm\text{-}rsda}}, G_1^{\Pi',\mathsf{nm\text{-}rsda}}) + qp \cdot 2^{-\ell},$$

where ℓ is the length of the MAC key.

Proof (sketch). Initially, reduction $R_b(\cdot)$ outputs (to D) the public key obtained from its oracle. When it gets $(\mathsf{chall}, m_0, m_1)$, it outputs $((\mathsf{chall}, m_b \,\|\, K, m_b \,\|\, K^*))$ to its oracle and gets a response e_1^*. Then, it computes $e_2^* \leftarrow T_K(e_1^*)$ and outputs (e_1^*, e_2^*). As long as no self-destruct has occurred, $R_b(\cdot)$ answers (components of parallel) decryption queries (e'_1, e'_2) (different from the challenge ciphertext) as follows: It outputs (dec, e'_1) to its oracle. If the answer is test, H_b verifies the tag e'_2 with K and returns m_b to D if it is valid. If the answer is $m' \,\|\, K'$, H_b verifies the tag with K' and returns m' if it is valid.

By inspection one observes that $R_b(G_0^{\Pi',\mathsf{nm\text{-}rsda}})$ behaves as $G_b^{\Pi'',\mathsf{nm\text{-}sda}}$ unless D asks a query (e'_1, e'_2) where e'_1 is an encryption of a message concatenated with K^*; however, since the view of D when interacting with $R_b(G_0^{\Pi',\mathsf{nm\text{-}rsda}})$ is independent of K^*, the probability of this event is bounded by $2^{-\ell}$.

On the other hand, observe that $R_b(G_1^{\Pi',\mathsf{nm\text{-}rsda}})$ behaves exactly as hybrid H_b. □

Hybrid 2. The second hybrid H'_b behaves as H_b except that queries (e'_1, e'_2) where e'_1 contains K^* are always rejected.

Lemma 34. *There exists a reduction $R''_b(\cdot)$ such that for all distinguishers D,*

$$\Delta^D(H_b, H'_b) \quad \leq \quad \Gamma^{D(R''_b(\cdot))}(G^{\mathsf{mac}}).$$

Proof. $R''_b(\cdot)$ is a standard reduction to the strong unforgeability of the MAC. □

Reduction to NM-RSDA. Distinguishing $G_0^{\Pi',\mathsf{nm\text{-}rsda}}$ and $G_1^{\Pi',\mathsf{nm\text{-}rsda}}$ can now be reduced to distinguishing H'_0 and H'_1.

Lemma 35. *There exists a reduction $R'(\cdot)$ such that for all distinguishers D,*

$$\Delta^D(H'_0, H'_1) \quad = \quad \Delta^{DR'(\cdot)}(G_0^{\Pi',\mathsf{nm\text{-}rsda}}, G_1^{\Pi',\mathsf{nm\text{-}rsda}}).$$

Proof (sketch). The reduction translates between the NM-SDA game for Π'' and the NM-RSDA game for Π', using the fact that decryption queries for which the first component contains K^* can be rejected. In particular, when the NM-RSDA game outputs test, a ciphertext can be rejected. □

Putting it together. The proof of Lemma 32 follows by combining Lemma 33, Lemma 34, and Lemma 35.

6.3 Variations

By combining Theorem 26, Theorem 14, and Corollary 23, we obtain a 1-to-k-bit black-box domain extension for NM-SDA making $\mathcal{O}(k)$ calls to the underlying 1-bit scheme.[26] Moreover, it is easy to see that the very same construction works for the case of NM-CPA security, the difference being that one only needs a secret-state non-malleable code tolerating a single parallel tampering query (i.e., $p = 1$). This proves Theorem 1 for the case of NM-SDA and NM-CPA.

The above construction also works for IND-SDA security by instantiating the construction with the coding scheme from Section 5.2 (cf. Theorem 9).[27] This yields Theorem 1 for the case of IND-SDA. Note that the resulting PKE has a shorter secret key, as we do not need to store the secret state for the non-malleable code. The security proof is a special case of that of Theorem 26 where each decryption query has parallelism 1.

7 Construction from CPA Security

In this section we show that NM-SDA security can be achieved in a black-box fashion from IND-CPA security. Specifically, we prove that a generalization using LECSS (cf. Section 2) of the scheme by Choi *et al.* [25] (dubbed the CDMW construction in the remainder of this section) is NM-SDA secure. Using a constant-rate LECSS allows to improve the rate of the CDMW construction from $\Omega(1/\lambda^2)$ to $\Omega(1/\lambda)$, where λ is the security parameter. This abstraction might also give a deeper understanding of the result of [25]. The main difficulty in the analysis is to extend their proof to deal with adaptively chosen parallel decryption queries (with self-destruct).

7.1 The CDMW Construction

The CDMW construction uses a randomized Reed-Solomon code, which is captured as a special case by the notion of a linear error-correcting secret sharing (LECSS) $(\mathsf{E}, \mathsf{D}, \mathsf{R})$ (cf. Section 2).

The LECSS has to satisfy an additional property, which is that given a certain number of symbols chosen uniformly at random and independently and a plaintext x, one can efficiently produce an encoding that matches the given symbols and has the same distribution as $\mathsf{E}(x)$. It is described in more detail in the proof of Lemma 41, where it is needed.[28]

Let $\Pi = (KG, E, D)$ be a PKE scheme with message space $\mathcal{M} = \{0,1\}^\ell$ (we assume $\ell = \Omega(\lambda)$), and let $\Sigma = (KG^{\mathsf{ots}}, S, V)$ be a one-time signature scheme with verification keys of length $\kappa = \mathcal{O}(\lambda)$. Moreover, let $\alpha > 0$ be any constant and (E, D) a (k, n, δ, τ)-LECSS over $\mathrm{GF}(2^\ell)$ with $\delta > 2\alpha$.

The CDMW construction (cf. Figure 11), to encrypt a plaintext $m \in \{0,1\}^{k\ell}$, first computes an encoding $(c_1, \ldots, c_n) \leftarrow \mathsf{E}(m)$ and then creates the $(\kappa \times n)$-matrix $\mathbf{C}$ in which this encoding is repeated in every row. For every entry $\mathbf{C}_{ij}$ of this matrix, there are two possible public keys $\mathsf{pk}_{i,j}^b$; which of them is used to encrypt the entry is determined by the i^{th} bit $v[i]$ of the verification key $\mathsf{verk} = (v[1], \ldots, v[\kappa])$ of a freshly generated key pair for Σ. In the end, the encrypted matrix $\mathbf{E}$ is signed using verk, producing a signature σ. The ciphertext is $(\mathbf{E}, \mathsf{verk}, \sigma)$.

[26] Note that for the construction to be secure, it is necessary that $n = \Omega(\lambda)$ and, therefore, due to the constant rate of the LECSS, the plaintext length is $k = \Omega(\lambda)$ as well.

[27] This requires only a small, purely syntactical change to the coding scheme. In particular, the secret state is simply the empty string.

[28] This property is also known as "reconstruction from partial views", and ECSSs (i.e., LECSSs without linearity) with this additional guarantee are known as reconstructable probabilistic encodings [26]. Of course, the Reed-Solomon-based LECSS from [25] satisfies the reconstruction property.

PKE Scheme $\Pi' = (KG', E', D')$

Key Generation KG'
for $(b,i,j) \in \{0,1\} \times [\kappa] \times [n]$
  $(\mathsf{pk}^b_{i,j}, \mathsf{sk}^b_{i,j}) \leftarrow KG$
$\mathsf{PK} \leftarrow (\mathsf{pk}^b_{i,j})_{b,i,j}$
$\mathsf{SK} \leftarrow (\mathsf{sk}^b_{i,j})_{b,i,j}$
$T \leftarrow\$ \binom{[n]}{\tau n}$
return $(\mathsf{PK}, (\mathsf{SK}, T))$

Encryption $E'_{\mathsf{PK}}(m)$
$(c_1, \ldots, c_n) \leftarrow \mathsf{E}(m)$
$(\mathsf{verk}, \mathsf{sigk}) \leftarrow KG^{\mathsf{ots}}$
$(v[1], \ldots, v[\kappa]) \leftarrow \mathsf{verk}$
for $(i,j) \in [\kappa] \times [n]$
  $e_{i,j} \leftarrow E_{\mathsf{pk}^{v[i]}_{i,j}}(c_j)$
$\mathbf{E} \leftarrow (e_{i,j})_{i,j}$
$\sigma \leftarrow S_{\mathsf{sigk}}(\mathbf{E})$
return $(\mathbf{E}, \mathsf{verk}, \sigma)$

Decryption $D'_{(\mathsf{SK},T)}(\mathbf{E}, \mathsf{verk}, \sigma)$
if $V_{\mathsf{verk}}(\mathbf{E}, \sigma) = 0$
  return $\perp$
for $j \in T$
  decrypt j^{th} column of $\mathbf{E}$
  if *not all entries identical*
    return $\perp$
decrypt first row of $\mathbf{E}$ to c
$(m, w) \leftarrow \mathsf{R}(c, \alpha n)$
if $w = \perp$ *or* $\exists j \in T : c_j \neq w_j$
  return $\perp$
return m

Figure 11: *The CDMW PKE scheme* Π' *constructed from a CPA-secure scheme* Π *[25]. We write* $\binom{[n]}{\tau n}$ *for the collection of all subsets of* $[n]$ *with size* τn.

The decryption first verifies the signature. Then, it decrypts all columns indexed by a set $T \subset [n]$, chosen as part of the secret key, and checks that each column consists of a single value only. Finally, it decrypts the first row and tries to find a codeword with relative distance at most α. If so, it checks whether the codeword matches the first row in the positions indexed by T. If all checks pass, it outputs the plaintext corresponding to the codeword; otherwise it outputs $\perp$.

In the remainder of this section, we sketch the proof of the following theorem, which implies Theorem 2.

Theorem 36. *Let* $t \in \mathbb{N}$ *and* Π *be a* $(t + t_{\mathsf{cpa}}, \varepsilon_{\mathsf{cpa}})$*-IND-CPA-secure PKE scheme,* $\alpha > 0$, (E, D) *a* (k, n, δ, τ)*-LECSS with* $\delta > 2\alpha$*, and* Σ *a* $(t + t_{\mathsf{ots}}, \varepsilon_{\mathsf{ots}})$*-secure OTS scheme with verification-key length* κ*. Then, for any* $q, p \in \mathbb{N}$*, PKE scheme* Π' *is* (t, q, p, ε)*-NM-SDA-secure with*

$$\varepsilon \quad = \quad (1-\tau)\kappa n \cdot \varepsilon_{\mathsf{cpa}} + 2 \cdot \varepsilon_{\mathsf{ots}} + 4 \cdot p(1-\tau)^{\alpha n},$$

where t_{cpa} *and* t_{ots} *represent the overhead incurred by corresponding reductions.*

Instantiating the construction. Note that the security proof below does not use the linearity of the LECSS. The CDMW construction can be seen as using a Reed-Solomon-based LECSS with rate $\mathcal{O}(1/\kappa)$. If the construction is instantiated with a constant-rate LECSS, the final rate improves over CDMW by a factor of $\Omega(\kappa) = \Omega(\lambda)$. More concretely, assuming a constant-rate CPA encryption, a ciphertext of length $\mathcal{O}(\lambda^3)$ can encrypt a plaintext of length $\Omega(\lambda^2)$ as compared to $\Omega(\lambda)$ for plain CDMW. As shown in Section 7.3, the LECSS can be instantiated with constructions based on Reed-Solomon or algebraic geometric codes (which also satisfy the additional property mentioned above), both with constant rate. Among the constant-rate codes, algebraic geometric codes allow to choose the parameters optimally also for shorter plaintexts.

7.2 Security Proof of the CDMW Construction

7.2.1 Overview

The proof follows the original one by [25]. The main change is that one needs to argue that, unless they contain invalid ciphertexts, adaptively chosen parallel queries do not allow the attacker to obtain useful information, in particular on the secret set T. This is facilitated by using the *self-destruct lemma* (cf. Section 3). The proof proceeds in three steps using two hybrid games H_b and H'_b:

- The first hybrid H_b gets rid of signature forgeries for the verification key used to create the challenge ciphertext. The indistinguishability of the hybrid from $G_b^{\Pi', \mathsf{nm\text{-}sda}}$ follows from the security of the OTS scheme and requires only minor modifications compared to the original proof.

- The second hybrid H'_b uses an alternative decryption algorithm. The indistinguishability of H'_b and H_b holds unconditionally; this step requires new techniques compared to the original proof.
- Finally, the distinguishing advantage between H'_0 and H'_1 is bounded by a reduction to the IND-CPA security of the underlying scheme Π; the reduction again resembles the one in [25].

7.2.2 Dealing with Forgeries

For $b \in \{0,1\}$, hybrid H_b behaves as $G_b^{\Pi',\mathsf{nm\text{-}sda}}$ but generates the signature key pair $(\mathsf{sigk}^*, \mathsf{verk}^*)$ used for the challenge ciphertext initially and rejects any decryption query $(\mathbf{E}', \sigma', \mathsf{verk}')$ if $\mathsf{verk}' = \mathsf{verk}^*$.

Lemma 37. *For $b \in \{0,1\}$, there exists a reduction $R'_b(\cdot)$ such that for all distinguishers D,*

$$\Delta^D(G_b^{\Pi',\mathsf{nm\text{-}sda}}, H_b) \quad \leq \quad \Gamma^{R'_b(D)}(G^{\Sigma,\mathsf{ots}}).$$

Proof. $R'_b(\cdot)$ is a standard reduction to the unforgeability of Σ. □

7.2.3 Alternative Decryption Algorithm

For $b \in \{0,1\}$, hybrid H'_b behaves as H_b but for the way it answers decryption queries $(\mathbf{E}', \sigma', \mathsf{verk}')$: As before, it first verifies the signature σ' and checks that each column of $\mathbf{E}'$ consists of encryptions of a single value. Then, it determines the first position i at which verk' and verk^* differ, i.e., where $v'[i] \neq v^*[i]$. It decrypts the i^{th} row of $\mathbf{E}$ and checks if there is a codeword w within distance $2\alpha n$.[29] If such w does not exist or else if w does not match the *first* row in a position indexed by T, the check fails. Otherwise, the plaintext corresponding to w is output.

Lemma 38. *For $b \in \{0,1\}$ and all distinguishers D, $\Delta^D(H_b, H'_b) \leq 2 \cdot p(1-\tau)^{\alpha n}$.*

The proof of Lemma 38 shows that the original and alternative decryption algorithms are indistinguishable not just for a single parallel query (as is sufficient for NM-CPA) but even against adaptively chosen parallel queries (with self-destruct). It is the main technical contribution of this section.

At the core of the proof is an analysis of how different types of encoding matrices $\mathbf{C}$ are handled inside the two decryption algorithms. To that end, one can define two games B and B' (below) that capture the behaviors of the original and the alternative decryption algorithms, respectively. The proof is completed by bounding $\Delta(B, B')$ (for all distinguishers) and showing the existence of a wrapper W_b such that $W_b(B)$ behaves as H_b and $W_b(B')$ as H'_b (also below). This proves the lemma since $\Delta^D(H_b, H'_b) = \Delta^D(W_b(B), W_b(B')) = \Delta^{D(W_b(\cdot))}(B, B')$.

The games B and B' behave as follows: Both initially choose a random size-τ subset of $[n]$. Then, they accept parallel queries with components of the type $(\mathbf{C}, i)$ for $\mathbf{C} \in \mathbb{F}^{\kappa \times n}$ and $i \in [\kappa]$. The answer to each component is computed as follows:

1. Both games check that all columns indexed by T consist of identical entries.
2. Game B tries to find a codeword w with distance less than αn from the *first* row (regardless of i), whereas B' tries to find w within $2\alpha n$ of row i. Then, if such a w is found, *both* games check that it matches the *first* row of $\mathbf{C}$ in the positions indexed by T.
3. If all checks succeed, the answer to the (component) query is w; otherwise, it is $\perp$.

Both games then output the answer vector and implement the self-destruct, i.e., if any of the answers is $\perp$, all *future* queries are answered by $\perp$.

Claim 39. *For $b \in \{0,1\}$ and all distinguishers D, $\Delta^D(B, B') \leq 2 \cdot p(1-\tau)^{\alpha n}$.*

[29] Recall that the actual decryption algorithm always decrypts the first row and tries to find w within distance αn.

Encoding matrices. Towards a proof of Claim 39, consider the following partition of the set of encoding matrices $\mathbf{C}$ (based on the classification in [25]):

1. There exists a codeword w within αn of the first row of $\mathbf{C}$, and all rows have distance at most αn.
2. (a) There exist two rows in $\mathbf{C}$ with distance greater than αn.
   (b) The rest; in this case the first row differs in more than αn positions from any codeword.

Observe that queries $(\mathbf{C}, i)$ with $\mathbf{C}$ of type 1 are treated identically by both B and B': A codeword w within αn of the first row of $\mathbf{C}$ is certainly found by B; since all rows have distance at most αn, w is within $2\alpha n$ of row i and thus also found by B'. Furthermore, note that if $\mathbf{C}$ is of type 2b, it is always rejected by B (but not necessarily by B').

Consider the hybrids C and C' that behave as B and B', respectively, but always reject *all* type-2 queries. Since type-1 queries are treated identically, C and C' are indistinguishable. Moreover:

Claim 40. *For all distinguishers D,*

$$\Delta^D(B, C) \quad \leq \quad p(1-\tau)^{\alpha n} \qquad \textit{and} \qquad \Delta^D(C', B') \quad \leq \quad p(1-\tau)^{\alpha n}.$$

The proof of Claim 40 follows a generic paradigm, at whose core is the so-called *self-destruct lemma*, which deals with the indistinguishability of hybrids with the self-destruct property and is explained in detail in Section 3. Roughly, this lemma applies whenever the first hybrid (in this case B resp. B') can be turned into the second one (in this case C resp. C') by changing ("bending") the answers to a subset (the "bending set") of the possible queries to always be $\perp$, and when additionally non-bent queries have a unique answer (cf. the statement of Lemma 6). Intuitively, the lemma states that parallelism and adaptivity do not help distinguish (much) in such cases.

Proof. To use the self-destruct lemma, note that B, C, C', and B' all answer queries from $\mathcal{X} := \mathbb{F}^{\kappa \times n} \times [\kappa]$ by values from $\mathcal{Y} := \mathbb{F}^n$. Moreover, note that they use as internal randomness a uniformly chosen element T from the set $\mathcal{R} := \binom{[n]}{\tau n}$ of size-τn subsets of $[n]$.

Consider first B and C. Let $g : \mathcal{X} \times \mathcal{R} \to \mathcal{Y}$ correspond to how B answers queries $(\mathbf{C}, i)$ (see above). Let $\mathcal{B}$ be the set $\mathcal{B}$ of all type-2a-queries. Then, C is its $\mathcal{B}$-bending (cf. Definition 6). Observe that queries $x = (\mathbf{C}, i) \notin \mathcal{B}$ are either of type 1 or 2b. For the former, the unique answer y_x is the codeword w within αn of the first row of $\mathbf{C}$; for the latter, y_x is $\perp$. Therefore, using the self-destruct lemma (Lemma 6), for all distinguishers D,

$$\Delta^D(B, C) \quad \leq \quad p \cdot \max_{(\mathbf{C},i) \in \mathcal{B}} \mathsf{P}[g((\mathbf{C}, i), T) \neq \perp],$$

where the probability is over the choice of T. Since type-2a queries have two rows with distance greater than αn, the probability over the choice of T that this remains unnoticed is at most $(1-\tau)^{\alpha n}$.

For the second part of the claim, consider B' and C'. Now, let $g : \mathcal{X} \times \mathcal{R} \to \mathcal{Y}$ correspond to how B' answers queries $(\mathbf{C}, i)$ (see above again), and let $\mathcal{B}$ be the set $\mathcal{B}$ of all type-2-queries. Then, C' is the $\mathcal{B}$-bending of B'.

Note that all queries $x = (\mathbf{C}, i) \notin \mathcal{B}'$ are of type 1, and the unique answer y_x is the codeword w within $2\alpha n$ of row i of $\mathbf{C}$. Therefore, using Lemma 6 again, for all distinguishers D,

$$\Delta^D(B', C') \quad \leq \quad p \cdot \max_{(\mathbf{C},i) \in \mathcal{B}'} \mathsf{P}[g'((\mathbf{C}, i), T) \neq \perp],$$

where the probability is again over the choice of T. Since type-2a queries have two rows with distance greater than αn and in type-2b queries the first row differs in more than αn positions from any codeword, the probability over the choice of T that this remains unnoticed is at most $(1-\tau)^{\alpha n}$. □

Proof (of Claim 39). The proof follows using the triangle inequality:

$$\Delta^D(B, B') \quad \leq \quad \Delta^D(B, C) + \Delta^D(C, C') + \Delta^D(C', B') \quad \leq \quad 2 \cdot p(1-\tau)^{\alpha n}.$$

□

Wrapper. It remains to show that there exists a wrapper W_b such that $W_b(B)$ behaves as H_b and $W_b(B')$ as H'_b. The construction of W_b is straight forward: H_b and H'_b generate all keys and the challenge in the identical fashion; therefore, W_b can do it the same way. W_b answers decryption queries $(\mathbf{E}', \mathsf{verk}', \sigma')$ by first verifying the signature σ' and rejecting queries if σ' is invalid or if verk' is identical to the verification key verk^* chosen for the challenge, decrypting the entire matrix $\mathbf{E}'$ to $\mathbf{C}'$ and submitting $(\mathbf{C}', i)$ to the oracle (either B or B'), where i is the first position at which verk' and verk^* differ, and decoding the answer w and outputting the result or simply forwarding it if it is $\perp$. Moreover, W_b implements the self-destruct. By inspection it can be seen that $W_b(B)$ implements the original decryption algorithm and $W_b(B')$ the alternative one.

7.2.4 Reduction to IND-CPA Security

Lemma 41. *There exists a reduction $R(\cdot)$ such that for all distinguishers D,*

$$\Delta^D(H'_0, H'_1) \quad = \quad (1-\tau)\kappa n \cdot \Delta^{D(R(\cdot))}(G_0^{\Pi,\mathsf{ind\text{-}cpa}}, G_1^{\Pi,\mathsf{ind\text{-}cpa}}).$$

Proof (sketch). The proof is a straight-forward generalization of the original proof by [25]; the only difference is that it needs to process multiple parallel decryption queries and implement the self-destruct feature appropriately. For ease of exposition, we describe the reduction to a many-public-key version of the CPA game for Π.[30]

Reduction $R(\cdot)$ initially chooses the secret set T and creates the challenge OTS key pair with verification key $\mathsf{verk}^* = (v^*[1], \ldots, v^*[\kappa])$ and all key pairs $(\mathsf{pk}^b_{i,j}, \mathsf{sk}^b_{i,j})$ with $j \in T$ or $b \neq v^*[i]$. The remaining $(1-\tau)\kappa n$ key pairs are generated by the CPA game.

Recall that the LECSS is assumed to satisfy the following property: Given τn symbols $(c_i)_{i\in T}$ chosen uniformly at random and independently and any plaintext $x \in \mathbb{F}^k$, one can efficiently sample symbols $(c_i)_{i\notin T}$ such that $(c_1, \ldots, c_n)$ has the same distribution as $\mathsf{E}(x)$. Using this fact, $R(\cdot)$ creates the challenge for m_0 and m_1 as follows: It picks the random symbols $(c_i)_{i\in T}$ and completes them to two full encodings c_{m_0} and c_{m_1} with the above procedure, once using m_0 and once using m_1 as the plaintext. Let $\mathbf{C}_{m_0}$ and $\mathbf{C}_{m_1}$ be the corresponding matrices (obtained by copying the encodings κ times). Observe that the two matrices match in the columns indexed by T. These entries are encrypted by $R(\cdot)$, using the public key $\mathsf{pk}^b_{i,j}$ for entry (i,j) for which $b \neq v^*[i]$. Denote by $\mathbf{C}'_{m_0}$ and $\mathbf{C}'_{m_1}$ the matrices $\mathbf{C}_{m_0}$ and $\mathbf{C}_{m_1}$ with the columns in T removed. The reduction outputs $(\mathsf{chall}, \mathbf{C}'_{m_0}, \mathbf{C}'_{m_1})$ to its oracle and obtains the corresponding ciphertexts, which it combines appropriately with the ones it created itself to form the challenge ciphertext.

Finally, note that since the reduction knows all the secret keys $\mathsf{pk}^b_{i,j}$ with $b \neq v^*[i]$, it can implement the alternative decryption algorithm (and the self-destruct). □

7.2.5 Overall Proof

Proof (of Theorem 36). Let t_{cpa} be the overhead caused by reduction $R(\cdot)$ and t_{ots} the larger of the overheads caused by $R'_0(\cdot)$ and $R'_1(\cdot)$. Moreover, let D be a distinguisher with running time at most t. Using the triangle inequality, and Lemmas 37, 38, and 41,

$$\begin{aligned}
\Delta^D(G_0^{\Pi',\mathsf{nm\text{-}sda}}, G_1^{\Pi',\mathsf{nm\text{-}sda}}) \quad &\leq \quad \Delta^D(G_0^{\Pi',\mathsf{nm\text{-}sda}}, H_0) + \Delta^D(H_0, H'_0) \\
&\quad + \Delta^D(H'_0, H'_1) + \Delta^D(H'_1, H_1) + \Delta^D(H_1, G_1^{\Pi',\mathsf{nm\text{-}sda}}) \\
&\leq \quad \Gamma^{D(R'_0(\cdot))}(G^{\Sigma,\mathsf{ots}}) + 2 \cdot p(1-\tau)^{\alpha n} \\
&\quad + (1-\tau)\kappa n \cdot \Delta^{D(R(\cdot))}(G_0^{\Pi,\mathsf{ind\text{-}cpa}}, G_1^{\Pi,\mathsf{ind\text{-}cpa}}) \\
&\quad + 2 \cdot p(1-\tau)^{\alpha n} + \Gamma^{D(R'_1(\cdot))}(G^{\Sigma,\mathsf{ots}})
\end{aligned}$$

[30] In the many-public-key version of the CPA game, an attacker can play the CPA game for several independently generated public keys simultaneously; this is equivalent to the normal formulation by a standard hybrid argument [13].

$$\leq \quad \varepsilon_{\mathsf{ots}} + 2 \cdot p(1-\tau)^{\alpha n} + (1-\tau)\kappa n \cdot \varepsilon_{\mathsf{cpa}} + 2 \cdot p(1-\tau)^{\alpha n} + \varepsilon_{\mathsf{ots}}.$$

□

7.3 LECSS for the CDMW Construction

In this section we show how to instantiate the LECSS used for the CDMW construction in Section 7. Let $\mathbb{F}$ be a finite field of size $L = 2^\ell$, where ℓ is the plaintext length of the IND-CPA scheme used in the construction. Then, there are the following variants of a (k, n, δ, τ)-LECSS:

- *CDMW Reed-Solomon codes:* The original CDMW construction can be seen as using a Reed-Solomon-based LECSS with rate $\Theta(1/\lambda)$, which is suboptimal (see next item).
- *Constant-Rate Reed-Solomon codes:* Cheraghchi and Guruswami [24] provide a LECSS based on a construction by Dziembowski *et al.* [40] and on Reed-Solomon (RS) codes with $\ell = \Theta(\log n)$. One can show that it achieves the following parameters (not optimized): $\alpha = 1/8$, $\tau = 1/8$ and rate $k/n \geq 1/4$ (i.e., all constant).
- *Algebraic geometric codes:* Using algebraic geometric (AG) codes, Cramer *et al.* [31] provide a LECSS with $\ell = \mathcal{O}(1)$ and still constant error correction, secrecy, and rate (but with worse concrete constants than Reed-Solomon codes).

Note that asymptotically, RS and AG codes are equally good: both have constant rate, distance, and secrecy. However, since with AG codes ℓ is constant (i.e., they work over an alphabet of constant size), the minimal plaintext length can be shorter than with RS codes.

References

[1] Divesh Aggarwal, Yevgeniy Dodis, Tomasz Kazana, and Maciej Obremski. Non-malleable reductions and applications. In *STOC*, pages 459–468, 2015.

[2] Divesh Aggarwal, Yevgeniy Dodis, and Shachar Lovett. Non-malleable codes from additive combinatorics. In *STOC*, pages 774–783, 2014.

[3] Divesh Aggarwal, Nico Döttling, Jesper Buus Nielsen, Maciej Obremski, and Erick Purwanto. Continuous non-malleable codes in the 8-split-state model. In *EUROCRYPT*, pages 531–561, 2019.

[4] Divesh Aggarwal, Stefan Dziembowski, Tomasz Kazana, and Maciej Obremski. Leakage-resilient non-malleable codes. In *TCC*, pages 398–426, 2015.

[5] Divesh Aggarwal, Tomasz Kazana, and Maciej Obremski. Inception makes non-malleable codes stronger. In *TCC*, pages 319–343, 2017.

[6] Shashank Agrawal, Divya Gupta, Hemanta K. Maji, Omkant Pandey, and Manoj Prabhakaran. Explicit non-malleable codes against bit-wise tampering and permutations. In *CRYPTO*, pages 538–557, 2015.

[7] Shashank Agrawal, Divya Gupta, Hemanta K. Maji, Omkant Pandey, and Manoj Prabhakaran. A rate-optimizing compiler for non-malleable codes against bit-wise tampering and permutations. In *TCC*, pages 375–397, 2015.

[8] Marshall Ball, Dana Dachman-Soled, Siyao Guo, Tal Malkin, and Li-Yang Tan. Non-malleable codes for small-depth circuits. In *IEEE FOCS*, pages 826–837, 2018.

[9] Marshall Ball, Dana Dachman-Soled, Mukul Kulkarni, Huijia Lin, and Tal Malkin. Non-malleable codes against bounded polynomial time tampering. In *EUROCRYPT*, pages 501–530, 2019.

[10] Marshall Ball, Dana Dachman-Soled, Mukul Kulkarni, and Tal Malkin. Non-malleable codes for bounded depth, bounded fan-in circuits. In *EUROCRYPT*, pages 881–908, 2016.

[11] Marshall Ball, Dana Dachman-Soled, Mukul Kulkarni, and Tal Malkin. Non-malleable codes from average-case hardness: AC^0, decision trees, and streaming space-bounded tampering. In *EUROCRYPT*, pages 618–650, 2018.

[12] Marshall Ball, Siyao Guo, and Daniel Wichs. Non-malleable codes for decision trees. In *CRYPTO*, pages 413–434, 2019.

[13] Mihir Bellare, Alexandra Boldyreva, and Silvio Micali. Public-key encryption in a multi-user setting: Security proofs and improvements. In *EUROCRYPT*, pages 259–274, 2000.

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

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

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

[17] Nishanth Chandran, Vipul Goyal, Pratyay Mukherjee, Omkant Pandey, and Jalaj Upadhyay. Block-wise non-malleable codes. In *ICALP*, pages 31:1–31:14, 2016.

[18] Nishanth Chandran, Bhavana Kanukurthi, and Srinivasan Raghuraman. Information-theoretic local non-malleable codes and their applications. In *TCC*, pages 367–392, 2016.

[19] Eshan Chattopadhyay, Vipul Goyal, and Xin Li. Non-malleable extractors and codes, with their many tampered extensions. In *ACM STOC*, pages 285–298, 2016.

[20] Eshan Chattopadhyay and Xin Li. Non-malleable codes and extractors for small-depth circuits, and affine functions. In *ACM STOC*, pages 1171–1184, 2017.

[21] Eshan Chattopadhyay and David Zuckerman. Non-malleable codes against constant split-state tampering. In *FOCS*, pages 306–315, 2014.

[22] Binyi Chen, Yilei Chen, Kristina Hostáková, and Pratyay Mukherjee. Continuous space-bounded non-malleable codes from stronger proofs-of-space. In *CRYPTO*, pages 467–495, 2019.

[23] Mahdi Cheraghchi and Venkatesan Guruswami. Capacity of non-malleable codes. In *Innovations in Theoretical Computer Science*, pages 155–168, 2014.

[24] Mahdi Cheraghchi and Venkatesan Guruswami. Non-malleable coding against bit-wise and split-state tampering. In *TCC*, pages 440–464, 2014.

[25] 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.

[26] Seung Geol Choi, Dana Dachman-Soled, Tal Malkin, and Hoeteck Wee. Improved, black-box, non-malleable encryption from semantic security. *Des. Codes Cryptogr.*, 86(3):641–663, 2018.

[27] Sandro Coretti, Yevgeniy Dodis, Björn Tackmann, and Daniele Venturi. Non-malleable encryption: Simpler, shorter, stronger. In *TCC*, pages 306–335, 2016.

[28] Sandro Coretti, Antonio Faonio, and Daniele Venturi. Rate-optimizing compilers for continuously non-malleable codes. In *ACNS*, pages 3–23, 2019.

[29] Sandro Coretti, Ueli Maurer, Björn Tackmann, and Daniele Venturi. From single-bit to multi-bit public-key encryption via non-malleable codes. In *TCC*, pages 532–560, 2015.

[30] Ronald Cramer, Ivan Bjerre Damgård, Nico Döttling, Serge Fehr, and Gabriele Spini. Linear secret sharing schemes from error correcting codes and universal hash functions. In *EUROCRYPT*, pages 313–336, 2015.

[31] 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.

[32] Ronald Cramer and Victor Shoup. A practical public key cryptosystem provably secure against adaptive chosen ciphertext attack. In *CRYPTO*, volume 1462 of *LNCS*, pages 13–25, 1998.

[33] 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.

[34] Dana Dachman-Soled. A black-box construction of a CCA2 encryption scheme from a plaintext aware encryption scheme. In Hugo Krawczyk, editor, *PKC*, LNCS. Springer, 2014.

[35] Dana Dachman-Soled and Mukul Kulkarni. Upper and lower bounds for continuous non-malleable codes. In *PKC*, pages 519–548, 2019.

[36] Ivan Damgård, Tomasz Kazana, Maciej Obremski, Varun Raj, and Luisa Siniscalchi. Continuous NMC secure against permutations and overwrites, with applications to CCA secure commitments. In *TCC*, pages 225–254, 2018.

[37] Whitfield Diffie and Martin E. Hellman. New directions in cryptography. *IEEE Transactions on Information Theory*, 22(6):644–654, 1976.

[38] Danny Dolev, Cynthia Dwork, and Moni Naor. Non-malleable cryptography. *SIAM J. Comput.*, 30(2):391–437, 2000.

[39] Stefan Dziembowski, Tomasz Kazana, and Maciej Obremski. Non-malleable codes from two-source extractors. In *CRYPTO*, pages 239–257, 2013.

[40] Stefan Dziembowski, Krzysztof Pietrzak, and Daniel Wichs. Non-malleable codes. In *Innovations in Theoretical Computer Science*, pages 434–452, 2010.

[41] Taher ElGamal. A public key cryptosystem and a signature scheme based on discrete logarithms. In *CRYPTO*, pages 10–18, 1984.

[42] Antonio Faonio, Jesper Buus Nielsen, Mark Simkin, and Daniele Venturi. Continuously non-malleable codes with split-state refresh. In *ACNS*, pages 121–139, 2018.

[43] Sebastian Faust, Kristina Hostáková, Pratyay Mukherjee, and Daniele Venturi. Non-malleable codes for space-bounded tampering. In *CRYPTO*, pages 95–126, 2017.

[44] Sebastian Faust, Pratyay Mukherjee, Jesper Buus Nielsen, and Daniele Venturi. Continuous non-malleable codes. In *TCC*, pages 465–488, 2014.

[45] Sebastian Faust, Pratyay Mukherjee, Daniele Venturi, and Daniel Wichs. Efficient non-malleable codes and key-derivation for poly-size tampering circuits. In *EUROCRYPT*, pages 111–128, 2014.

[46] Eiichiro Fujisaki, Tatsuaki Okamoto, David Pointcheval, and Jacques Stern. RSA-OAEP is secure under the RSA assumption. In Joe Kilian, editor, *CRYPTO*, volume 2139 of *LNCS*, pages 260–274, Heidelberg, 2001. Springer.

[47] Rosario Gennaro, Anna Lysyanskaya, Tal Malkin, Silvio Micali, and Tal Rabin. Algorithmic tamper-proof (ATP) security: Theoretical foundations for security against hardware tampering. In *TCC*, pages 258–277, 2004.

[48] Yael Gertner, Tal Malkin, and Steven Myers. Towards a separation of semantic and CCA security for public key encryption. In *TCC*, pages 434–455, 2007.

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

[50] Javier Herranz, Dennis Hofheinz, and Eike Kiltz. Some (in)sufficient conditions for secure hybrid encryption. *Inf. Comput.*, 208(11):1243–1257, 2010.

[51] Dennis Hofheinz and Eike Kiltz. Practical chosen ciphertext secure encryption from factoring. In Antoine Joux, editor, *EUROCRYPT*, volume 5479 of *LNCS*, pages 313–332, Heidelberg, 2009. Springer.

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

[53] Susan Hohenberger, Allison B. Lewko, and Brent Waters. Detecting dangerous queries: A new approach for chosen ciphertext security. In *EUROCRYPT*, pages 663–681, 2012.

[54] Zahra Jafargholi and Daniel Wichs. Tamper detection and continuous non-malleable codes. In *TCC*, pages 451–480, 2015.

[55] Bhavana Kanukurthi, Sai Lakshmi Bhavana Obbattu, and Sruthi Sekar. Four-state non-malleable codes with explicit constant rate. In *TCC*, pages 344–375, 2017.

[56] Aggelos Kiayias, Feng-Hao Liu, and Yiannis Tselekounis. Practical non-malleable codes from l-more extractable hash functions. In *ACM CCS*, pages 1317–1328, 2016.

[57] Eike Kiltz, Krzysztof Pietrzak, Martijn Stam, and Moti Yung. A new randomness extraction paradigm for hybrid encryption. In Antoine Joux, editor, *EUROCRYPT*, volume 5479 of *LNCS*, pages 590–609, Heidelberg, 2009. Springer.

[58] Kaoru Kurosawa and Yvo Desmedt. A new paradigm of hybrid encryption scheme. In *CRYPTO*, pages 426–442, 2004.

[59] Xin Li. Improved non-malleable extractors, non-malleable codes and independent source extractors. In *ACM STOC*, pages 1144–1156, 2017.

[60] Huijia Lin and Stefano Tessaro. Amplification of chosen-ciphertext security. In *EUROCRYPT*, pages 503–519, 2013.

[61] Yehuda Lindell. A simpler construction of CCA2-secure public-key encryption under general assumptions. In *EUROCRYPT*, pages 241–254, 2003.

[62] Feng-Hao Liu and Anna Lysyanskaya. Tamper and leakage resilience in the split-state model. In *CRYPTO*, pages 517–532, 2012.

[63] F.J. MacWilliams and N.J.A. Sloane. *The Theory of Error-Correcting Codes*. North-holland Publishing Company, 2nd edition, 1978.

[64] Steven Myers, Mona Sergi, and Abhi Shelat. Blackbox construction of a more than non-malleable CCA1 encryption scheme from plaintext awareness. In Ivan Visconti and Roberto De Prisco, editors, *Security and Cryptography for Networks*, volume 7485 of *LNCS*, pages 149–165. Springer, 2012.

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

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

[67] Rafail Ostrovsky, Giuseppe Persiano, Daniele Venturi, and Ivan Visconti. Continuously non-malleable codes in the split-state model from minimal assumptions. In *CRYPTO*, pages 608–639, 2018.

[68] 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.

[69] Rafael Pass, Abhi Shelat, and Vinod Vaikuntanathan. Relations among notions of non-malleability for encryption. In *ASIACRYPT*, pages 519–535, 2007.

[70] Chris Peikert and Brent Waters. Lossy trapdoor functions and their applications. *SIAM J. Comput.*, 40(6):1803–1844, 2011.

[71] Ronald L. Rivest, Adi Shamir, and Leonard M. Adleman. A method for obtaining digital signatures and public-key cryptosystems (reprint). *Commun. ACM*, 26(1):96–99, 1983.

[72] Alon Rosen and Gil Segev. Chosen-ciphertext security via correlated products. *SIAM J. Comput.*, 39(7):3058–3088, 2010.

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

[74] Ba-Zhong Shen. A Justesen construction of binary concatenated codes that asymptotically meet the Zyablov bound for low rate. *IEEE Transactions on Information Theory*, 39(1):239–242, 1993.

[75] Salil P. Vadhan. *A Study of Statistical Zero-Knowledge Proofs*. PhD thesis, Massachusetts Institute of Technology, 1999.