From Single-Bit to Multi-Bit Public-Key Encryption via Non-Malleable Codes

Sandro Coretti[1], Ueli Maurer[1], Björn Tackmann[2,*], and Daniele Venturi[3]

[1] ETH Zurich, {corettis,maurer}@inf.ethz.ch
[2] UC San Diego, btackmann@eng.ucsd.edu
[3] 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 chosen-ciphertext secure—the resulting scheme is *malleable*. This paper analyzes 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. An attacker's ability to make multiple decryption queries requires that the underlying code be *continuously* non-malleable (Faust *et al.*, TCC '14). This flavor of non-malleable codes can only be achieved if the decoder is allowed to "self-destruct" when it processes an invalid encoding. The resulting PKE scheme inherits this property and therefore only achieves a weaker variant of chosen-ciphertext security, where the decryption becomes dysfunctional once the attacker submits an invalid ciphertext.

We first show that the above approach based on non-malleable codes indeed yields a solution to the problem of domain extension for public-key encryption where the decryption may self-destruct, provided that the underlying code is continuously non-malleable against a *reduced* form of bit-wise tampering. This statement is shown by combining a simple information-theoretic argument with the constructive cryptography perspective on PKE (Coretti *et al.*, Asiacrypt '13). Then, we prove that the code of Dziembowski *et al.* is actually already continuously non-malleable against (full) bit-wise tampering; this constitutes the first *information-theoretically* secure continuously non-malleable code, a technical contribution that we believe is of independent interest. Compared to the previous approaches to PKE domain extension, our scheme is more efficient and intuitive, at the cost of not achieving full CCA security. Our result is also one of the first applications of non-malleable codes in a context other than memory tampering.

[*] Work done while author was at ETH Zurich.

1 Introduction

1.1 Overview

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 [22], the first formal definition of public-key encryption has been provided by Goldwasser and Micali [32], and to date numerous instantiations of this concept have been proposed, e.g., [50,25,17,29,33,36,51,49], for different security properties and based on various different computational assumptions.

One natural approach towards developing public-key encryption schemes based on weak and credible assumptions is to build "stronger" or more general schemes generically from "weaker" or less general ones. While the "holy grail"—generically building a chosen-ciphertext secure scheme based on any chosen-plaintext secure one—has so far remained out of reach, and despite negative results [31], various interesting positive results have been shown. For instance, Cramer *et al.* [16] build *bounded-query* chosen-ciphertext secure schemes from chosen-plaintext secure ones, Choi *et al.* [10] *non-malleable* schemes from chosen-plaintext secure ones, and Lin and Tessaro [38] show how the security of weakly chosen-ciphertext secure schemes can be amplified. A line of work started by Myers, Sergi, and shelat [47] and continued by Dachman-Soled [18] 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 [48] and Hohenberger, Lewko, and Waters [34], who generically build a multi-bit chosen-ciphertext secure scheme from a single-bit chosen-ciphertext secure one.

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, this simple approach does not yield chosen-ciphertext security. The reason is that the above scheme is *malleable*: 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[4] (a concept introduced by Dziembowski *et al.* [24]) 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 [24] are not sufficient: Since they are only secure against a single tampering, the security of the

[4] 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.

resulting scheme would only hold with respect to a single decryption. *Continuously* non-malleable codes (Faust *et al.* [26]) 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.

The resulting scheme achieves a notion weaker than full CCA, which we term *self-destruct chosen-ciphertext security (SD-CCA)*. Roughly, SD-CCA security is CCA security with the twist that the decryption oracle stops working once the adversary submits an invalid ciphertext.

Our paper consists of two main parts: First, we prove that the above approach allows to build multi-bit SD-CCA-secure PKE from single-bit SD-CCA-secure PKE, provided that the underlying code is continuously non-malleable against a *reduced* form of bit-wise tampering. This proof is greatly facilitated by rephrasing the problem using the paradigm of constructive cryptography [40], since it follows almost immediately from the composition theorem. For comparison, the full version of this paper [14] also contains a purely game-based proof. Second, we show that a simplified variant of the code by Dziembowski *et al.* [24] is already continuously non-malleable against the aforementioned reduced bit-wise tampering and that the full variant of said code achieves continuous non-malleability against *full* bit-wise tampering. This constitutes the first *information-theoretically* secure continuously non-malleable code, a contribution that we believe is of independent interest, and forms the technical core of this paper.

1.2 Techniques and Contributions

Constructive cryptography [40]. Statements about the security of cryptographic schemes can be stated as *constructions* of a "stronger" or more useful desired resource from a "weaker" or more restricted assumed one. Two such construction steps can be composed, i.e., if a protocol π constructs a resource S from an assumed resource R, denoted by $R \overset{\pi}{\Longmapsto} S$, and, additionally, a protocol ψ assumes resource S and constructs a resource T, then the composition theorem of constructive cryptography (see Section 2.4) states that the composed protocol, denoted $\psi \circ \pi$, constructs resource T from R. The resources considered in this work are different types of communication channels between two parties A and B; a channel is a resource that involves three entities: the sender, the receiver, and a (potential) attacker E.

We use and extend the notation by Maurer and Schmid [44], denoting different types of channels by different arrow symbols. A *confidential* channel (later denoted $-\!\diamond\!\!\twoheadrightarrow\!\bullet$) hides the messages sent by A from the attacker E but potentially allows her to inject *independent* messages; an *authenticated* channel (later denoted $\bullet\!\!-\!\diamond\!\!\twoheadrightarrow$) is dual to the confidential channel in that it potentially leaks the message to the attacker but prevents modifications and injections; an insecure channel (later denoted $-\,\twoheadrightarrow$) protects neither the confidentiality nor the authenticity. In all cases, the double arrow head indicates that the channel can be used to transmit multiple messages. A single arrow head, instead, means that channels are single-use. All channels used within this work are described formally in Section 2.5.

Warm-up: dealing with the malleability of the one-time pad. To illustrate the intuition behind our approach, consider the following simple example: The one-time pad allows to encrypt an n-bit message m using an n-bit shared key κ by computing the ciphertext $e = m \oplus \kappa$. If e is sent via an insecure channel, an attacker can replace it by a different ciphertext e', in which case the receiver will compute $m' = e' \oplus \kappa = m \oplus (e \oplus e')$. This can be seen, as described in previous work by Maurer *et al.* [43], as constructing from an insecure channel and a shared secret n-bit key an "XOR-malleable" channel, denoted $-\!\!\oplus\!\!\rightarrow\!\bullet$, which is confidential but allows the attacker to specify a mask $\delta \in \{0,1\}^n$ $(= e \oplus e')$ to be XORed to the transmitted message.

Non-malleable codes can be used to deal with the XOR-malleability. To transmit a k-bit message m, we encode m with a (k,n)-bit non-malleable code, obtaining an n-bit codeword c, which we transmit via the XOR-malleable channel $-\!\!\oplus\!\!\rightarrow\!\bullet$. Since by XORing a mask δ to a codeword transmitted via $-\!\!\oplus\!\!\rightarrow\!\bullet$ the attacker can influence the value of each bit of the codeword only independently, a code that is non-malleable w.r.t. the function class $\mathcal{F}_{\mathsf{bit}}$, which (in particular) allows to either "keep" or "flip" each bit of a codeword only individually, is sufficient. Indeed, the non-malleability of the code implies that the decoded message will be either the original message or a completely unrelated value, which is the same guarantee as formulated by the single-message confidential channel (denoted $\longrightarrow\!\bullet$), and hence using the code, one achieves the construction

$$-\!\!\oplus\!\!\rightarrow\!\bullet \qquad \Longmapsto \qquad \longrightarrow\!\bullet.$$

A more detailed treatment and a formalization of this example appears in the full version of this paper [14]; suitable non-malleable codes are described in [15,24,9].

Dealing with the malleability of multiple single-bit encryptions. Intuitively, CCA encryption guarantees that an attacker, by modifying a particular ciphertext, can either leave the message contained therein intact or replace it by an independently created one. This intuition is formally captured by the confidential channel $-\!\!\diamond\!\!\twoheadrightarrow\!\bullet$: at the attacker interface E, it allows to either forward messages sent by A or to inject *independent* messages. In [13], it is shown how CCA-secure encryption can be used to construct a confidential channel $-\!\!\diamond\!\!\twoheadrightarrow\!\bullet$ from A to B from an authenticated channel $\longleftarrow\!\bullet$ from B to A and an insecure channel $-\,-\!\!\twoheadrightarrow$ from A to B. As shown in Section 3, this and the composition theorem imply that using n independent single-bit PKE schemes, one can construct n (independent) instances of the single-bit confidential channel $\overset{\text{1-bit}}{-\!\!\diamond\!\!\twoheadrightarrow\!\bullet}$, written $[\overset{\text{1-bit}}{-\!\!\diamond\!\!\twoheadrightarrow\!\bullet}]^n$.

The remaining step is showing how to achieve the construction

$$[\overset{\text{1-bit}}{-\!\!\diamond\!\!\twoheadrightarrow\!\bullet}]^n \qquad \Longmapsto \qquad \overset{k\text{-bit}}{-\!\!\diamond\!\!\twoheadrightarrow\!\bullet} \tag{1}$$

for some $k > 1$. Then, by the composition theorem, plugging these two steps together yields a protocol $\mathsf{m\text{-}pke}$ that constructs a k-bit confidential channel from an authenticated channel and an insecure channel.

To achieve construction (1), we use non-malleable codes. The fact that the channels are multiple-use leads to two important differences to the one-time-pad example above: First, the attacker can fabricate *multiple* codewords, which are then decoded. Second, each bit of such a codeword can be created by combining *any* of the bits sent by A over the corresponding channel. These capabilities can be formally captured by a particular class $\mathcal{F}_{\mathsf{copy}}$ of tampering functions. We prove in Section 3 that any code that is *continuously non-malleable* w.r.t. $\mathcal{F}_{\mathsf{copy}}$ can be used to achieve (1).

Unfortunately, we show in Section 5 that any code, in order to satisfy the above type of non-malleability, has to "self-destruct" in the event of a decoding error. For the application in the setting of public-key encryption, this means that the decryption algorithm of the receiver B also has to deny processing any further ciphertext once the code self-destructs.

Self-destruct CCA security. In Section 3 we show how the protocol $\mathsf{m\text{-}pke}$ can be seen as a PKE scheme Π that achieves *self-destruct CCA security (SD-CCA)* and show that the single-bit confidential channel $\overset{\text{1-bit}}{-\!\!\diamond\!\!\twoheadrightarrow\!\bullet}$ can also be constructed using a single-bit SD-CCA scheme (instead of a CCA-secure one). Thus, overall we obtain a way to transform 1-bit SD-CCA-secure PKE into multi-bit SD-CCA-secure PKE. For comparison, the full version of this paper [14] also contains a direct, entirely game-based proof that combining a single-bit SD-CCA PKE scheme with a non-malleable code as above yields a multi-bit SD-CCA scheme.[5]

SD-CCA is a (weaker) CCA variant that allows the scheme to self-destruct in case it detects an invalid ciphertext. The standard CCA game can easily be extended to include the self-destruct mode of the decryption: the decryption oracle keeps answering decryption queries as long as no invalid ciphertext (i.e., a ciphertext upon which the decryption algorithm outputs an error symbol) is received; after such an event occurs, no further decryption query is answered.

The guarantees of SD-CCA are perhaps best understood if compared to the q-bounded CCA notion by [10]. While q-CCA allows an *a priori determined* number q of decryption queries, SD-CCA allows an *arbitrary* number of *valid* decryption queries and one invalid query. From a practical viewpoint, an attacker can efficiently violate the availability with a scheme of either notion. However, as long as no invalid ciphertexts are received, an SD-CCA scheme can run indefinitely, whereas a q-CCA scheme has to necessarily stop after q decryptions.

Subsequent work [12] shows that SD-CCA security can in fact be achieved from CPA security only, by generalizing a technique by Choi *et al.* [10]. The resulting scheme, however, is considerably less efficient than the one we provide in this paper (under reasonable assumptions about the plaintext size). In [12],

[5] In fact, PKE scheme Π is only *replayable* SD-CCA secure; cf. Section 3.4.

the authors also study the relation between SD-CCA and other standard security notions and discuss possible applications.

Continuous non-malleability w.r.t. $\mathcal{F}_{\mathsf{copy}}$. The class $\mathcal{F}_{\mathsf{copy}}$ can be seen as a multi-encoding version of the function class $\mathcal{F}_{\mathsf{set}}$, which consists of functions that tamper with every bit of an encoding individually and may either leave it unchanged or replace it by a fixed value. In Section 4 we build a continuously non-malleable code w.r.t. $\mathcal{F}_{\mathsf{copy}}$; the code consists of a linear error-correcting secret sharing (LECSS) scheme and can be seen as a simplified version of the code in [24]. The security proof of the code proceeds in two steps: First, we prove that it is continuously non-malleable w.r.t. $\mathcal{F}_{\mathsf{set}}$ against tampering with a single encoding; the main challenge in this proof is showing that by repeatedly tampering with an encoding, an attacker cannot infer (too much) useful information about it. Then, we show that if a code is continuously non-malleable w.r.t. $\mathcal{F}_{\mathsf{set}}$ against tampering with a single encoding, then it is also *adaptively* continuously non-malleable w.r.t. $\mathcal{F}_{\mathsf{copy}}$, i.e., against tampering with many encodings simultaneously. In addition, in the full version of this paper [14], we also show that the full version of the code by [24] is non-malleable against *full* bit-wise tampering (i.e., when additionally the tamper function is allowed to flip bits of an encoding). These are the main technical contributions of this work.

1.3 More Details on Related Work

The work of Hohenberger *et al.* [34]—building on the work of Myers and shelat [48]—describes a multi-bit CCA-secure encrytion scheme from a single-bit CCA-secure one, a CPA-secure one, and a 1-query-bounded CCA-secure one. Their scheme is rather sophisticated and has a somewhat circular structure, requiring a complex security proof. The public key is of the form $\mathsf{pk} = (\mathsf{pk}_{in}, \mathsf{pk}_A, \mathsf{pk}_B)$, where the "inner" public key pk_{in} is the public key of a so-called *detectable CCA (DCCA)* PKE scheme, which is built from the single-bit CCA-secure scheme, and the "outer" public keys pk_A and pk_B are, respectively, the public key of a 1-bounded CCA and a CPA secure PKE scheme. To encrypt a k-bit message m one first encrypts a tuple (r_A, r_B, m), using the "inner" public key pk_{in}, obtaining a ciphertext e_{in}, where r_A and r_B are thought as being the randomness for the "outer" encryption scheme. Next, one has to encrypt the inner ciphertext e_{in} under the "outer" public key pk_A (resp. pk_B) using randomness r_A (resp. r_B) and thus obtaining a ciphertext e_A (resp. e_B). The output ciphertext is $e = (e_A, e_B)$.

To use the above scheme, we have to instantiate the DCCA, 1-bounded CCA and CPA components. As argued in [34], all schemes can be instantiated using a single-bit CCA-secure PKE scheme yielding a fully black-box construction of a multi-bit CCA-secure PKE from a single-bit CCA-secure PKE. Let us denote with l_p (resp., l_e) the bit-length of the public key (resp., the ciphertext) for the single-bit CCA-secure PKE scheme. When we refer to the construction of [16] for the 1-bounded CCA component, we get a public key of size roughly $(3+16s)\cdot l_p$ for the public key and $(k+2s)\cdot 4s\cdot l_e^2$ for the ciphertext, for security parameter s.[6]

In contrast, our scheme instantiated with the information-theoretic LECSS scheme of [24] has a ciphertext of length $\approx 5k\cdot l_e$ and a public key of length $k\cdot l_p$. Note that the length of the public key depends on the length of the message, as we need independent public keys for each encrypted bit (whereas the DCCA scheme can use always the same public key). However, we observe that when k is not too large, e.g. in case the PKE scheme is used as a key encapsulation mechanism, we would have $k\approx s$ yielding public keys of comparable size. On the negative side, recall that our construction needs to self-destruct in case an invalid ciphertext is processed, which is not required in [34], and thus our construction only achieves SD-CCA security and not full-blown CCA security.

Non-malleable codes. Beyond the constructions of [24,9,26], non-malleable codes exists against block-wise tampering [11], against bit-wise tampering and permutations [5,4], against split-state tampering—both information-theoretic [23,2,7,3,1] and computational [39,19]—and in a setting where the computational complexity of the tampering functions is limited [8,28,35]. We stress that the typical application of non-malleable codes is to protect cryptographic schemes against memory tampering (see, e.g., [30,24,20,21]). A further application of non-malleable codes has been shown by Agrawal *et al.* [4] (in concurrent and independent work). They show that one can obtain a non-malleable multi-bit commitment scheme from a non-malleable single-bit commitment scheme by encoding the value with a (specific) non-malleable code and then committing to the codeword bits. Despite the similarity of the approaches, the techniques applied in their paper differ heavily from ours. The class of tampering functions the code has to protect against is different, and we additionally need continuous non-malleability to handle multiple decryption queries (this is not required for the commitment case).

2 Preliminaries

2.1 Systems: Resources, Converters, Distinguishers, and Reductions

Resources and converters. We use the concepts and terminology of abstract [42] and constructive cryptography [40]. The *resources* we consider are different types of communication channels, which are systems with three interfaces labeled by A, B, and E. A *converter* is a two-interface system which is directed in that it has an *inside* and an *outside* interface. Converters model protocol engines that are used by the parties, and using a protocol is modeled by connecting the party's interface of the resource to the inside interface of the converter (which hides those two interfaces) and using the outside interface of the converter instead. We generally use upper-case, bold-face letters (e.g., $\mathbf{R}$,

[6] For simplicity, we assumed that the random strings r_A, r_B are computed by stretching the seed (of length s) of a pseudo-random generator.

$\mathbf{S}$) or channel symbols (e.g., $\bullet\!\!-\!\!\diamond\!\!\twoheadrightarrow$) to denote resources or single-interface systems and lower-case Greek letters (e.g., α, β) or sans-serif fonts (e.g., enc, dec) for converters. We denote by Φ the set of all resources and by Σ the set of all converters.

For $I \in \{A, B, E\}$, a resource $\mathbf{R} \in \Phi$, and a converter $\alpha \in \Sigma$, the expression $\alpha^I\mathbf{R}$ denotes the composite system obtained by connecting the inside interface of α to interface I of $\mathbf{R}$; the outside interface of α becomes the I-interface of the composite system. The system $\alpha^I\mathbf{R}$ is again a resource (cf. Figure 5 on page 16). For two resources $\mathbf{R}$ and $\mathbf{S}$, $[\mathbf{R}, \mathbf{S}]$ denotes the parallel composition of $\mathbf{R}$ and $\mathbf{S}$. For each $I \in \{A, B, E\}$, the I-interfaces of $\mathbf{R}$ and $\mathbf{S}$ are merged and become the *sub-interfaces* of the I-interface of $[\mathbf{R}, \mathbf{S}]$.

Two converters α and β can be composed serially by connecting the inside interface of β to the outside interface of α, written $\beta \circ \alpha$, with the effect that $(\beta \circ \alpha)^I\mathbf{R} = \beta^I\alpha^I\mathbf{R}$. Moreover, converters can also be taken in parallel, denoted by $[\alpha, \beta]$, with the effect that $[\alpha, \beta]^I[\mathbf{R}, \mathbf{S}] = [\alpha^I\mathbf{R}, \beta^I\mathbf{S}]$. We assume the existence of an identity converter $\mathsf{id} \in \Sigma$ with $\mathsf{id}^I\mathbf{R} = \mathbf{R}$ for all resources $\mathbf{R} \in \Phi$ and interfaces $I \in \{A, B, E\}$ and of a special converter $\bot \in \Sigma$ with an inactive outside interface.

Distinguishers. A *distinguisher* $\mathbf{D}$ connects to all interfaces of a resource $\mathbf{U}$ and outputs a single bit at the end of its interaction with $\mathbf{U}$. The expression $\mathbf{DU}$ defines a binary random variable corresponding to the output of $\mathbf{D}$ when interacting with $\mathbf{U}$, and the *distinguishing advantage of a distinguisher* $\mathbf{D}$ *on two systems* $\mathbf{U}$ *and* $\mathbf{V}$ is defined as

$$\Delta^{\mathbf{D}}(\mathbf{U}, \mathbf{V}) := |\mathsf{P}[\mathbf{DU} = 1] - \mathsf{P}[\mathbf{DV} = 1]|.$$

The distinguishing advantage measures how much the output distribution of $\mathbf{D}$ differs when it is connected to either $\mathbf{U}$ or $\mathbf{V}$. Note that the distinguishing advantage is a pseudo-metric.[7]

Reductions. When relating two distinguishing problems, it is convenient to use a special type of system $\mathbf{C}$ that translates one setting into the other. Formally, $\mathbf{C}$ is a converter that has an *inside* and an *outside* interface. When it is connected to a system $\mathbf{S}$, which is denoted by $\mathbf{CS}$, the inside interface of $\mathbf{C}$ connects to the (merged) interface(s) of $\mathbf{S}$ and the outside interface of $\mathbf{C}$ is the interface of the composed system. $\mathbf{C}$ is called a *reduction system* (or simply *reduction*).

To reduce distinguishing two systems $\mathbf{S}, \mathbf{T}$ to distinguishing two systems $\mathbf{U}, \mathbf{V}$, one exhibits a reduction $\mathbf{C}$ such that $\mathbf{CS} \equiv \mathbf{U}$ and $\mathbf{CT} \equiv \mathbf{V}$. Then, for all distinguishers $\mathbf{D}$, we have $\Delta^{\mathbf{D}}(\mathbf{U}, \mathbf{V}) = \Delta^{\mathbf{D}}(\mathbf{CS}, \mathbf{CT}) = \Delta^{\mathbf{DC}}(\mathbf{S}, \mathbf{T})$. The last equality follows from the fact that $\mathbf{C}$ can also be thought of as being part of the distinguisher (which follows from the *composition-order independence* [42]).

[7] That is, for any $\mathbf{D}$, it is symmetric, satisfies the triangle inequality, and $\Delta^{\mathbf{D}}(\mathbf{R}, \mathbf{R}) = 0$ for all $\mathbf{R}$.

2.2 Discrete Systems

The behavior of systems can be formalized by random systems as in [46,41]: A random system $\mathbf{S}$ is a sequence $(\mathsf{p}^{\mathbf{S}}_{Y^i|X^i})_{i\geq 1}$ of conditional probability distributions, where $\mathsf{p}^{\mathbf{S}}_{Y^i|X^i}(y^i, x^i)$ is the probability of observing the outputs $y^i = (y_1, \ldots, y_i)$ given the inputs $x^i = (x_1, \ldots, x_i)$. If for two systems $\mathbf{R}$ and $\mathbf{S}$,

$$\mathsf{p}^{\mathbf{R}}_{Y^i|X^i} = \mathsf{p}^{\mathbf{S}}_{Y^i|X^i}$$

for all i and for all parameters where both are defined, they are called *equivalent*, denoted by $\mathbf{R} \equiv \mathbf{S}$. In that case, $\Delta^{\mathbf{D}}(\mathbf{R}, \mathbf{S}) = 0$ for all distinguishers $\mathbf{D}$.

A system $\mathbf{S}$ can be extended by a so-called *monotone binary output* (or *MBO*) $\mathcal{B}$, which is an additional one-bit output $B_1, B_2, \ldots$ with the property that $B_i = 1$ implies $B_{i+1} = 1$ for all i.[8] The enhanced system is denoted by $\hat{\mathbf{S}}$, and its behavior is described by the sequence $(\mathsf{p}^{\hat{\mathbf{S}}}_{Y^i,B_i|X^i})_{i\geq 1}$. If for two systems $\hat{\mathbf{R}}$ and $\hat{\mathbf{S}}$ with MBOs,

$$\mathsf{p}^{\hat{\mathbf{R}}}_{Y^i,B_i=0|X^i} = \mathsf{p}^{\hat{\mathbf{S}}}_{Y^i,B_i=0|X^i}$$

for all i, they are called *game equivalent*, which is denoted by $\hat{\mathbf{R}} \overset{g}{\equiv} \hat{\mathbf{S}}$. In such a case, $\Delta^{\mathbf{D}}(\mathbf{R}, \mathbf{S}) \leq \Gamma^{\mathbf{D}}(\hat{\mathbf{R}}) = \Gamma^{\mathbf{D}}(\hat{\mathbf{S}})$, where $\Gamma^{\mathbf{D}}(\hat{\mathbf{R}})$ denotes the probability that $\mathbf{D}$ provokes the MBO. For more details and a proof of this fact, consult [41].[9]

2.3 The Notion of Construction

We formalize the security of protocols via the notion of *construction*, introduced in [42]:

Definition 1. *Let Φ and Σ be as above, and let ε_1 and ε_2 be two functions mapping each distinguisher $\mathbf{D}$ to a real number in $[0, 1]$. A protocol $\pi = (\pi_1, \pi_2) \in \Sigma^2$ constructs resource $\mathbf{S} \in \Phi$ from resource $\mathbf{R} \in \Phi$ with distance $(\varepsilon_1, \varepsilon_2)$ and with respect the simulator $\sigma \in \Sigma$, denoted*[10]

$$\mathbf{R} \overset{\pi,\sigma,(\varepsilon_1,\varepsilon_2)}{\Longmapsto} \mathbf{S},$$

if for all distinguishers $\mathbf{D}$,

$$\begin{cases} \Delta^{\mathbf{D}}({\pi_1}^A {\pi_2}^B \bot^E \mathbf{R}, \bot^E \mathbf{S}) \leq \varepsilon_1(\mathbf{D}) & (\textit{availability}) \\ \Delta^{\mathbf{D}}({\pi_1}^A {\pi_2}^B \mathbf{R}, \sigma^E \mathbf{S}) \leq \varepsilon_2(\mathbf{D}) & (\textit{security}). \end{cases}$$

The availability condition captures that a protocol must correctly implement the functionality of the constructed resource in the absence of the attacker. The security condition models the requirement that everything the attacker can achieve in the setting with the assumed resource and the protocol, she can also accomplish in the setting with the constructed resource (using the simulator to translate the behavior).

[8] In other words, once the MBO is 1, it cannot return to 0.

[9] Intuitively, this means that in order to distinguish the two systems, $\mathbf{D}$ has to provoke the MBO.

[10] In less formal contexts, we sometimes drop the superscripts on $\Longmapsto$.

2.4 The Composition Theorem

The above construction notion composes in the following two ways:[11] First, if one (lower-level) protocol constructs the resource that is assumed by the other (higher-level) protocol, then the composition of those two protocols constructs the same resource as the higher-level protocol, but from the resources assumed by the lower-level protocol, under the assumptions that occur in (at least) one of the individual security statements. Second, the security of constructions is maintained in the presence of arbitrary resources taken in parallel.

To state the theorem, we make use of the special converter id (cf. Section 2.1). Furthermore, we assume the operation $[\cdot, \ldots, \cdot]$ to be left-associative; in this way we can simply express multiple resources using the single variable $\mathbf{U}$.

Theorem 1. *Let $\mathbf{R}, \mathbf{S}, \mathbf{T}, \mathbf{U} \in \Phi$ be resources. Let $\pi = (\pi_1, \pi_2)$ and $\psi = (\psi_1, \psi_2)$ be protocols, σ_π and σ_ψ be simulators, and $(\varepsilon_\pi^1, \varepsilon_\pi^2)$, $(\varepsilon_\psi^1, \varepsilon_\psi^2)$ such that*

$$\mathbf{R} \quad \overset{(\pi,\sigma_\pi,(\varepsilon_\pi^1,\varepsilon_\pi^2))}{\Longmapsto} \quad \mathbf{S} \qquad \textit{and} \qquad \mathbf{S} \quad \overset{(\psi,\sigma_\psi,(\varepsilon_\psi^1,\varepsilon_\psi^2))}{\Longmapsto} \quad \mathbf{T}.$$

Then

$$\mathbf{R} \quad \overset{(\alpha,\sigma_\alpha,(\varepsilon_\alpha^1,\varepsilon_\alpha^2))}{\Longmapsto} \quad \mathbf{T}$$

with $\alpha = (\psi_1 \circ \pi_1, \psi_2 \circ \pi_2)$, $\sigma_\alpha = \sigma_\pi \circ \sigma_\psi$, and $\varepsilon_\alpha^i(\mathbf{D}) = \varepsilon_\pi^i(\mathbf{D}\sigma_\psi^E) + \varepsilon_\psi^i(\mathbf{D}\pi_1^A\pi_2^B)$, where $\mathbf{D}\sigma_\psi^E$ and $\mathbf{D}\pi_1^A\pi_2^B$ mean that $\mathbf{D}$ applies the converters at the respective interfaces. Moreover

$$[\mathbf{R}, \mathbf{U}] \quad \overset{([\pi,(\mathsf{id},\mathsf{id})],[\sigma_\pi,\mathsf{id}],(\bar{\varepsilon}_\pi^1,\bar{\varepsilon}_\pi^2))}{\Longmapsto} \quad [\mathbf{S}, \mathbf{U}],$$

with $\bar{\varepsilon}_\pi^i(\mathbf{D}) = \varepsilon_\pi^i(\mathbf{D}[\cdot, \mathbf{U}])$, where $\mathbf{D}[\cdot, \mathbf{U}]$ means that the distinguisher emulates $\mathbf{U}$ in parallel. (The analogous statement holds with respect to $[\mathbf{U}, \mathbf{R}]$ and $[\mathbf{U}, \mathbf{S}]$.)

2.5 Channel Resources

From the perspective of constructive cryptography, the purpose of a public-key encryption scheme is to construct a confidential channel from non-confidential channels. A channel is a resource that involves a sender A, a receiver B, and—to model channels with different levels of security—an attacker E. The main types of channels relevant to this work are defined below with respect to interface set $\{A, B, E\}$. All channels are parametrized by a message space $\mathcal{M} \subseteq \{0,1\}^*$, which is only made explicit in the confidential channel (see below), however.

[11] The composition theorem was first explicitly stated in [45], but the statement there was restricted to asymptotic settings. Later, in [37], the theorem was stated in a way that also allows to capture concrete security statements. The proof, however, still follows the same steps as the one in [45].

Insecure multiple-use channel. The insecure channel $- \twoheadrightarrow$ transmits multiple messages $m \in \mathcal{M}$ and corresponds to, for instance, communication via the Internet. If no attacker is present (i.e., in case $\perp^E - \twoheadrightarrow$), then all messages are transmitted from A to B faithfully. Otherwise (for $- \twoheadrightarrow$), the communication can be controlled via the E-interface, i.e., the attacker learns all messages input at the A-interface and chooses the messages to be output at the B-interface. The channel is described in more detail in Figure 1.

Channel $\perp^E - \twoheadrightarrow$	Channel $- \twoheadrightarrow$	
on m *at* A | **output** m at B	**on** m *at* A | **output** (msg, m) at E	**on** (inj, m) *at* E | **output** m at B

Fig. 1. Insecure, multiple-use communication channel from A to B.

Authenticated (unreliable) single-use channel. The (single-use) authenticated channel $\bullet\!\!\longrightarrow$, described in Figure 2, allows the sender A to transmit a single message to the receiver B authentically. That means, while the attacker (at the E-interface) can still read the transmitted message, the only influence allowed is delaying the message (arbitrarily, i.e., there is no guarantee that the message will ever be delivered). The channel guarantees that *if* a message is delivered to B, *then* this message was input by A before. There are different constructions that result in the channel $\bullet\!\!\longrightarrow$, based on, for instance, MACs or signature schemes.

Channel $\perp^E \bullet\!\!\longrightarrow$	Channel $\bullet\!\!\longrightarrow$	
on first m *at* A | **output** m at B	**on first** m *at* A | **output** (msg, m) at E	**on first** dlv *at* E | **output** m at B (if defined)

Fig. 2. Authenticated, single-use communication channel from A to B.

Confidential multiple-use channel. The k-bit confidential channel $\overset{k\text{-bit}}{-\!\diamond\!\!\twoheadrightarrow\!\bullet}$ allows to transmit multiple messages $m \in \{0,1\}^k$. If no attacker is present (i.e., in case $\perp^E \overset{k\text{-bit}}{-\!\diamond\!\!\twoheadrightarrow\!\bullet}$), then all messages are transmitted from A to B faithfully. Otherwise (for $\overset{k\text{-bit}}{-\!\diamond\!\!\twoheadrightarrow\!\bullet}$), all messages $m \in \{0,1\}^k$ input at the A-interface are stored in a buffer $\mathcal{B}$.[12] The attacker can then choose messages from the buffer $\mathcal{B}$ (by using an index) to be delivered at the B-interface, or inject messages from $\{0,1\}^k$ which are then also output at the B-interface. Note that E cannot inject mes-

[12] The $\diamond$ in the symbol $\overset{k\text{-bit}}{-\!\diamond\!\!\twoheadrightarrow\!\bullet}$ is to suggest the presence of $\mathcal{B}$.

sages that depend on those in $\mathcal{B}$, i.e., the confidential channel is non-malleable. It is described in more detail in Figure 3.

Channel $\bot^E \overset{k\text{-bit}}{-\!\!\diamond\!\!\twoheadrightarrow\!\bullet}$

```
on $m \in \{0,1\}^k$ at $A$
    output $m$ at $B$
```

Channel $\overset{k\text{-bit}}{-\!\!\diamond\!\!\twoheadrightarrow\!\bullet}$

```
init
    $i \leftarrow 0$
    $\mathcal{B} \leftarrow \emptyset$

on $m \in \{0,1\}^k$ at $A$
    $i \leftarrow i + 1$
    $\mathcal{B} \leftarrow \mathcal{B} \cup \{(i, m)\}$
    output $(\mathsf{msg}, i)$ at $E$

on $(\mathsf{dlv}, i')$ at $A$
    if $\exists m : (i', m) \in \mathcal{B}$
        output $m$ at $B$

on $(\mathsf{inj}, m')$ at $A$
    output $m'$ at $E$
```

Fig. 3. Confidential, multiple-use k-bit channel from A to B.

2.6 Public-Key Encryption Schemes

A public-key encryption (PKE) scheme with message space $\mathcal{M} \subseteq \{0,1\}^*$ and ciphertext space $\mathcal{E}$ is defined as three algorithms $\Pi = (K, E, D)$, where the key-generation algorithm K 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 $e \in \mathcal{E}$ 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 $\diamond$, 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 K.

We introduce security notions for PKE schemes as we need them.

2.7 Continuously Non-Malleable Codes

Non-malleable codes, introduced in [24], are coding schemes that protect the encoded messages against certain classes of adversarially chosen modifications, in the sense that the decoding will result either in the original message or in an unrelated value.

Definition 2 (Coding scheme). *A* (k, n)-coding scheme $(\mathrm{Enc}, \mathrm{Dec})$ *consists of a randomized encoding function* $\mathrm{Enc} : \{0,1\}^k \to \{0,1\}^n$ *and a deterministic decoding function* $\mathrm{Dec} : \{0,1\}^n \to \{0,1\}^k \cup \{\diamond\}$ *such that* $\mathrm{Dec}(\mathrm{Enc}(x)) = x$ *(with probability* 1 *over the randomness of the encoding function) for each* $x \in \{0,1\}^k$*. The special symbol* $\diamond$ *indicates an* invalid codeword*.*

In the original definition [24], the adversary is allowed to modify the codeword via a function of a specified class $\mathcal{F}$ only once. Continuous non-malleability, introduced in [26], extends this guarantee to the case where the adversary is allowed to perform multiple such modifications for a fixed target codeword. The notion of *adaptive* continuous non-malleability considered here is an extension of the one

System $\mathbf{S}^{\mathsf{real}}_{\mathcal{F}}$

```
init
    $i \leftarrow 0$

on $(\mathsf{encode}, x)$
    $i \leftarrow i + 1$
    $c^{(i)} \leftarrow\!\!\$\ \mathrm{Enc}(x)$

on $(\mathsf{tamper}, f)$ with $f \in \mathcal{F}^{(i)}$
    $c' \leftarrow f(c^{(1)}, \dots, c^{(i)})$
    $x' \leftarrow \mathrm{Dec}(c')$
    if $x' = \diamond$
        self-destruct
    out $x'$
```

System $\mathbf{S}^{\mathsf{simu}}_{\mathcal{F},\tau}$

```
init
    $i \leftarrow 0$

on $(\mathsf{encode}, x)$
    $i \leftarrow i + 1$
    $x^{(i)} \leftarrow\!\!\$\ x$

on $(\mathsf{tamper}, f)$ with $f \in \mathcal{F}^{(i)}$
    $x' \leftarrow\!\!\$\ \tau(i, f)$
    if $x' = \diamond$
        self-destruct
    if $x' = (\mathsf{same}, j)$
        $x' \leftarrow x^{(j)}$
    out $x'$
```

Fig. 4. Systems $\mathbf{S}^{\mathsf{real}}_{\mathcal{F}}$ and $\mathbf{S}^{\mathsf{simu}}_{\mathcal{F},\tau}$ defining adaptive continuous non-malleability of (Enc, Dec). The command **self-destruct** has the effect that $\diamond$ is output and all future queries are answered by $\diamond$.

in [26] in that the adversary is allowed to adaptively specify messages and the functions may depend on multiple codewords. That is, the class $\mathcal{F}$ is actually a sequence $(\mathcal{F}^{(i)})_{i \geq 1}$ of function families with $\mathcal{F}^{(i)} \subseteq \{f \mid f : (\{0,1\}^n)^i \to \{0,1\}^n\}$, and after encoding i messages, the adversary chooses functions from $\mathcal{F}^{(i)}$. A similar adaptive notion has been already considered for continuous strong non-malleability in the split-state model [27].

Formally, adaptive continuous non-malleability w.r.t. $\mathcal{F}$ is defined by comparing the two random systems $\mathbf{S}^{\mathsf{real}}_{\mathcal{F}}$ and $\mathbf{S}^{\mathsf{simu}}_{\mathcal{F},\tau}$ defined in Figure 4. Both systems process encode and tamper queries from a distinguisher $\mathbf{D}$, whose objective is to tell the two systems apart.

System $\mathbf{S}^{\mathsf{real}}_{\mathcal{F}}$ produces a random encoding $c^{(i)}$ of each message $x^{(i)}$ specified by $\mathbf{D}$ and allows $\mathbf{D}$ to repeatedly issue tampering functions $f \in \mathcal{F}^{(i)}$. For each such query, $\mathbf{S}^{\mathsf{real}}_{\mathcal{F}}$ computes the modified codeword $c' = f(c^{(1)}, \dots, c^{(i)})$ and outputs $\mathrm{Dec}(c')$. Whenever $\mathrm{Dec}(c') = \diamond$, the system enters a self-destruct mode, in which all further queries are replied to by $\diamond$.

The second random system, $\mathbf{S}^{\mathsf{simu}}_{\mathcal{F},\tau}$, features a simulator τ, which is allowed to keep state. The simulator repeatedly takes a tampering function and outputs either a message x', (same, v) for $v \in \{1, \dots, i\}$, or $\diamond$, where (same, v) is used by τ to indicate that (it believes that) the tampering results in an n-bit string that decodes to the v^{th} message encoded. System $\mathbf{S}^{\mathsf{simu}}_{\mathcal{F},\tau}$ outputs whatever τ outputs, except that (same, v) is replaced by the v^{th} message $x^{(v)}$ specified by $\mathbf{D}$. Moreover, in case of $\diamond$, $\mathbf{S}^{\mathsf{simu}}_{\mathcal{F},\tau}$ self-destructs.

For $\ell, q \in \mathbb{N}$, $\mathbf{S}^{\mathsf{real}}_{\mathcal{F},\ell,q}$ is the system that behaves as $\mathbf{S}^{\mathsf{real}}_{\mathcal{F}}$ except that only the first ℓ encode-queries and the first q tamper-queries are handled (and similarly for $\mathbf{S}^{\mathsf{simu}}_{\mathcal{F},\tau,\ell,q}$ and $\mathbf{S}^{\mathsf{simu}}_{\mathcal{F},\tau}$). Note that by setting $\ell = 1$, one recovers continuous non-malleability as defined in [26],[13] and by additionally setting $q = 1$ the original definition of non-malleability.

Definition 3 (Continuous non-malleability). *Let* $\mathcal{F} = (\mathcal{F}^{(i)})_{i \geq 1}$ *be a sequence of function families* $\mathcal{F}^{(i)} \subseteq \{f \mid f : (\{0,1\}^n)^i \to \{0,1\}^n\}$ *and let* $\ell, q \in \mathbb{N}$. *A coding scheme* (Enc, Dec) *is* adaptively continuously $(\mathcal{F}, \varepsilon, \ell, q)$-non-malleable

[13] Being based on *strong* non-malleability [24], the notion of [26] is actually stronger than ours.

(or simply $(\mathcal{F}, \varepsilon, \ell, q)$-non-malleable*) if there exists a simulator* τ *such that* $\Delta^{\mathbf{D}}(\mathbf{S}^{\mathsf{real}}_{\mathcal{F},\ell,q}, \mathbf{S}^{\mathsf{simu}}_{\mathcal{F},\tau,\ell,q}) \leq \varepsilon$ *for all distinguishers* $\mathbf{D}$.

3 From Single-Bit to Multi-Bit Channels

In this section we examine the issue of domain extension for chosen-ciphertext-secure (CCA) public-key encryption (PKE). To that end, we employ the constructive cryptography paradigm and proceed in two main steps: First, we reuse a result by [13] saying that CCA-secure PKE can be used to construct a confidential channel from an authenticated and an insecure channel. This implies that using n independent copies of a single-bit CCA-secure PKE scheme, one obtains n parallel instances of a single-bit confidential channel. Second, using continuously non-malleable codes, we tie the n single-bit channels together and obtain a k-bit confidential channel. Combining these two steps using the composition theorem results in a protocol that constructs a k-bit confidential channel from an authenticated and an insecure channel.

The combined protocol can be seen as the following simple PKE scheme: first encode a k-bit message using a continuously non-malleable (k, n)-code to protect its integrity, obtaining an n-bit codeword c; then encrypt c bit-wise using n independent public keys for a single-bit CCA-secure PKE. As we show, continuously non-malleable codes only exist if the decoder is allowed to "self-destruct" once it processes an invalid codeword. This property translates to the resulting PKE scheme, which therefore only achieves a weaker form of CCA security, called *self-destruct CCA security (SD-CCA)*, where the decryption oracle stops working after the attacker submits an invalid ciphertext. Noting that SD-CCA security suffices to construct the single-bit confidential channels yields a domain-extension technique for SD-CCA-secure PKE schemes.

We stress that the need for self-destruct is not a limitation of the security proof of our code (cf. Section 4), as continuous non-malleability for the class of tampering functions required for the above transformation to work is impossible without the self-destruct property (cf. Section 5 for details).

3.1 Single-Bit PKE Viewed Constructively

Following the proof of [13, Theorem 2], one can show that a 1-bit SD-CCA-secure PKE scheme can be used to design a protocol that achieves the construction

$$[\longleftarrow\!\!\bullet, -\!-\!\!\twoheadrightarrow] \quad \Longmapsto \quad \overset{\text{1-bit}}{-\!\diamond\!\!\twoheadrightarrow\!\!\bullet}, \tag{2}$$

where, in a nutshell, the receiver's protocol converter is responsible for key generation, decryption, as well as self-destructing, the sender's protocol converter for encryption, and where the authenticated channel $\longleftarrow\!\!\bullet$ is used for the transmission of the public key and the insecure channel $-\!-\!\!\twoheadrightarrow$ for sending ciphertexts. The constructed single-bit confidential channel $\overset{\text{1-bit}}{-\!\diamond\!\!\twoheadrightarrow\!\!\bullet}$ hides all messages sent by the sender from the attacker and allows the attacker to either deliver already sent messages or to inject *independent* messages. This captures the intuitive

(SD-)CCA guarantee that an attacker, by modifying a particular ciphertext, can either leave the message contained therein intact or replace it by an independently created one.

Using n independent copies of the single-bit scheme in parallel yields a protocol $\mathsf{1\text{-}pke}$ that achieves:

$$[\longleftarrow\!\!\bullet,\ -\ \twoheadrightarrow] \quad \overset{\mathsf{1\text{-}pke}}{\Longmapsto} \quad [\overset{\text{1-bit}}{-\!\diamond\!\!\twoheadrightarrow\!\bullet}]^n, \tag{3}$$

which follows almost directly from the composition theorem. More details can be found in the full version of this paper [14].

3.2 Tying the Channels Together

We now show how to construct, using an adaptive continuously non-malleable (k,n)-code (cf. Section 2.7), a (single) k-bit confidential channel from the n independent single-bit confidential channels constructed in the previous section. This is achieved by having the sender encode the message with the non-malleable code and sending the resulting codeword over the 1-bit channels (bit-by-bit), while the receiver decodes all n-bit strings received via these channels. Additionally, due to the self-destruct property of continuously non-malleable codes, the receiver must stop decoding once an invalid codeword has been received.

More precisely, let $(\mathrm{Enc}, \mathrm{Dec})$ be a (k,n)-coding scheme and consider the following protocol $\mathsf{nmc} = (\mathsf{encode}, \mathsf{decode})$: Converter encode encodes every message $m \in \{0,1\}^k$ input at its outside interface with fresh randomness, resulting in an n-bit encoding $c = c_1 \cdots c_n \leftarrow \mathrm{Enc}(m)$. Then, for $i = 1, \ldots, n$, it outputs bit c_i to the i^{th} channel at the inside interface. Converter decode, whenever it receives an n-bit string $c' = c'_1 \cdots c'_n$ (where the i^{th} bit c'_i was received on the i^{th} channel), it computes $m' \leftarrow \mathrm{Dec}(c')$ and outputs m' at the outside interface. If $m' = \diamond$, it implements the self-destruct mode, i.e., it answers all future encodings received at the inside interface by outputting $\diamond$ at the outside interface.

The goal is now to show that protocol nmc achieves the construction

$$[\overset{\text{1-bit}}{-\!\diamond\!\!\twoheadrightarrow\!\bullet}]^n \quad \overset{\mathsf{nmc}}{\Longmapsto} \quad \overset{k\text{-bit}}{-\!\diamond\!\!\twoheadrightarrow\!\bullet}\ . \tag{4}$$

The required non-malleability. By inspecting both sides of Figure 5, it becomes immediately apparent why adaptive continuously non-malleable codes are the proper choice to achieve construction (4): On the left-hand side, the distinguisher can repeatedly input messages $m^{(i)}$ at interface A, which results in encodings $c^{(i)}$ being input (bit-by-bit) into the single-bit channels. Using the E-interfaces of these channels, the distinguisher can repeatedly see the decoding of an n-bit string $c' = c'_1 \cdots c'_n$ at interface B, where each bit c'_j results from either forwarding one of the bits already in the j^{th} channel or from injecting a fresh bit that is either 0 or 1.

Put differently, the distinguisher can effectively launch tampering attacks via functions from $\mathcal{F}_{\mathsf{copy}} := (\mathcal{F}^{(i)}_{\mathsf{copy}})_{i \geq 1}$, where $\mathcal{F}^{(i)}_{\mathsf{copy}} \subseteq \{f \mid f : (\{0,1\}^n)^i \to \{0,1\}^n\}$

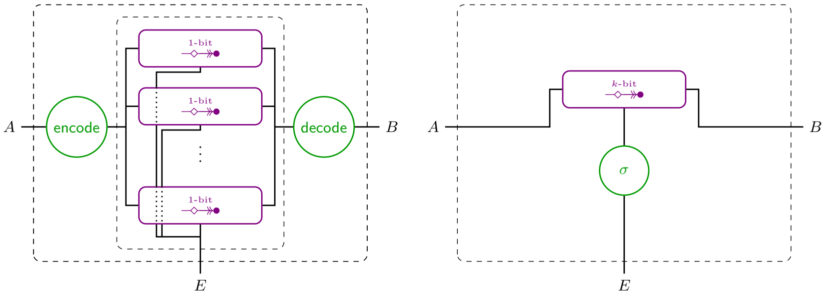

Fig. 5. Left: The assumed resource $[\overset{\text{1-bit}}{-\!\!\diamond\!\!\twoheadrightarrow\!\!\bullet}]^n$ with protocol converters encode and decode attached to interfaces A and B, denoted $\mathsf{encode}^A\mathsf{decode}^B[\overset{\text{1-bit}}{-\!\!\diamond\!\!\twoheadrightarrow\!\!\bullet}]^n$. Right: The constructed resource $\overset{k\text{-bit}}{-\!\!\diamond\!\!\twoheadrightarrow\!\!\bullet}$ with simulator σ attached to the E-interface, denoted $\sigma^E \overset{k\text{-bit}}{-\!\!\diamond\!\!\twoheadrightarrow\!\!\bullet}$. In particular, σ must simulate the E-interfaces of $[\overset{\text{1-bit}}{-\!\!\diamond\!\!\twoheadrightarrow\!\!\bullet}]^n$. The protocol is secure if the two systems are indistinguishable.

and each function $f \in \mathcal{F}^{(i)}_{\mathsf{copy}}$ is characterized by a vector $\chi(f) = (f_1, \ldots, f_n)$ where $f_j \in \{\mathsf{zero}, \mathsf{one}, \mathsf{copy}_1, \ldots, \mathsf{copy}_i\}$, with the meaning that f takes as input i codewords $(c^{(1)}, \ldots, c^{(i)})$ and outputs an n-bit string $c' = c'_1 \cdots c'_n$ in which each bit c'_j is either set to 0 (zero), set to 1 (one), or copied from the j^{th} bit in a codeword $c^{(v)}$ (copy_v) for $v \in \{1, \ldots, i\}$.

On the right-hand side, the distinguisher may again input messages $m^{(i)}$ at interface A, to the k-bit confidential channel. At interface E, this channel only allows to either deliver entire k-bit messages already sent by A or to inject independent messages. The simulator σ required to prove (4) needs to simulate the E-interfaces of the single-bit confidential channels at its outside interface and, based solely on what is input at these interfaces, decide whether to forward or inject a message, which corresponds exactly to the task of the simulator τ in the non-malleability experiment (cf. Section 2.7).

Theorem 2 below formalizes this correspondence; its proof is essentially a technicality: one merely needs to "translate" between the channel settings and the non-malleability experiment.

Theorem 2. *For any $\ell, q \in \mathbb{N}$, if* $(\mathrm{Enc}, \mathrm{Dec})$ *is* $(\mathcal{F}_{\mathsf{copy}}, \varepsilon, \ell, q)$*-continuously non-malleable, there exists a simulator σ such that*

$$\left[\overset{\text{1-bit},\ell,q}{-\!\!\diamond\!\!\twoheadrightarrow\!\!\bullet}\right]^n \quad \overset{(\mathsf{nmc},\sigma,(0,\varepsilon))}{\Longmapsto} \quad \overset{k\text{-bit},\ell,q}{-\!\!\diamond\!\!\twoheadrightarrow\!\!\bullet},$$

where the additional superscripts ℓ, q on a channel mean that it only processes the first ℓ queries at the A-interface and only the first q queries at the E-interface.

Proof. The availability condition holds by the correctness of the code.

Let $\mathcal{F} := \mathcal{F}_{\mathsf{copy}}$, $\mathbf{S}^{\mathsf{real}}_{\mathcal{F}} := \mathbf{S}^{\mathsf{real}}_{\mathcal{F},\ell,q}$, and $\mathbf{S}^{\mathsf{simu}}_{\mathcal{F},\tau} := \mathbf{S}^{\mathsf{simu}}_{\mathcal{F},\tau,\ell,q}$ where τ is the simulator guaranteed to exist by Definition 3.

Consider the following simulator σ (based on τ), which simulates the E-sub-interfaces of the 1-bit confidential channels at its outside interface: When

(msg, i) is received at the inside interface, it outputs (msg, i) at each outside sub-interface corresponding to a 1-bit confidential channel. Whenever σ receives one instruction to either deliver[14] $((\mathsf{dlv}, i')$ for $i' \in \mathbb{N})$ or inject $((\mathsf{inj}, m')$ for $m' \in \{0,1\})$ a bit at each outside sub-interface corresponding to one of the confidential channels, it assembles these to a function f with $\chi(f) = (f_1, \ldots, f_n)$ as follows: For all $j = 1, \ldots, n$,

$$f_j := \begin{cases} \mathsf{zero} & \text{if the instruction on the } j^{\text{th}} \text{ sub-interface is } (\mathsf{inj}, 0), \\ \mathsf{one} & \text{if the instruction on the } j^{\text{th}} \text{ sub-interface is } (\mathsf{inj}, 1), \\ \mathsf{copy}_{i'} & \text{if the instruction on the } j^{\text{th}} \text{ sub-interface is } (\mathsf{dlv}, i'). \end{cases}$$

Then, σ invokes τ to obtain $x' \leftarrow\!\!\$\; \tau(i, f)$, where i is the number of instructions (msg, i) received at the inside interface so far. If $x' = (\mathsf{same}, j)$, σ outputs (dlv, j) at the inside interface. Otherwise, it outputs (inj, x'). If $x' = \diamond$, σ outputs $(\mathsf{inj}, \diamond)$ at the inside interface and implements the self-destruct mode, i.e., outputs $(\mathsf{inj}, \diamond)$ at the inside interface for all future inputs to the simulated interfaces of the single-bit channels.

Consider the following reduction $\mathbf{C}$, which provides interfaces A, B, and E on the outside and expects to connect to either $\mathbf{S}^{\mathsf{real}}_{\mathcal{F}}$ or $\mathbf{S}^{\mathsf{simu}}_{\mathcal{F},\tau}$ on the inside. When a message m is input at the A-interface, $\mathbf{C}$ outputs (encode, m) on the inside. Similarly to σ, it repeatedly collects instructions input at the E-sub-interfaces and uses them to form a tamper function f, which it outputs on the inside as (tamper, f). Then, it outputs the answer x' received on the inside at the B-interface. Additionally, if $x' = \diamond$, $\mathbf{C}$ implements the self-destruct mode, i.e., subsequently only outputs $\diamond$ at interface B.

One observes that

$$\mathbf{C}\mathbf{S}^{\mathsf{real}}_{\mathcal{F}} \equiv \mathsf{encode}^{A}\mathsf{decode}^{B}[\overset{\text{1-bit}}{-\!\!\diamond\!\!\twoheadrightarrow\!\bullet}]^{n} \qquad \text{and} \qquad \mathbf{C}\mathbf{S}^{\mathsf{simu}}_{\mathcal{F},\tau} \equiv \sigma^{E}\ \overset{k\text{-bit},\ell,q}{-\!\!\diamond\!\!\twoheadrightarrow\!\bullet}\ .$$

Thus, for all distinguishers $\mathbf{D}$,

$$\begin{aligned} \Delta^{\mathbf{D}}(\mathsf{encode}^{A}\mathsf{decode}^{B}[\overset{\text{1-bit}}{-\!\!\diamond\!\!\twoheadrightarrow\!\bullet}]^{n}, \sigma^{E}\ \overset{k\text{-bit},\ell,q}{-\!\!\diamond\!\!\twoheadrightarrow\!\bullet}) &= \Delta^{\mathbf{D}}(\mathbf{C}\mathbf{S}^{\mathsf{real}}_{\mathcal{F}}, \mathbf{C}\mathbf{S}^{\mathsf{simu}}_{\mathcal{F},\tau}) \\ &= \Delta^{\mathbf{DC}}(\mathbf{S}^{\mathsf{real}}_{\mathcal{F}}, \mathbf{S}^{\mathsf{simu}}_{\mathcal{F},\tau}) \leq \varepsilon. \end{aligned}$$

□

3.3 Plugging It Together

The composition theorem of constructive cryptography (cf. Section 2.4) implies that the protocol $\mathsf{m\text{-}pke} = \mathsf{nmc} \circ \mathsf{1\text{-}pke}$ resulting from composing the protocols $\mathsf{1\text{-}pke}$ and nmc for transformations (3) and (4), respectively, achieves

$$[\longleftarrow\!\bullet, -\!\!\twoheadrightarrow] \quad \overset{\mathsf{m\text{-}pke}}{\Longmapsto} \quad \overset{k\text{-bit}}{-\!\!\diamond\!\!\twoheadrightarrow\!\bullet}\ . \tag{5}$$

[14] For simplicity, assume that no deliver instruction (dlv, i') for some i' greater than the largest number i received via (msg, i) at the inside interface so far is input.

System $\mathbf{G}_b^{\textsf{sd-cca}}$

```
init
    $(\textsf{pk}, \textsf{sk}) \leftarrow K$
    output $\textsf{pk}$

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

on $(\textsf{dec}, e')$
    $m' \leftarrow D_{\textsf{sk}}(e')$
    if $m' = \diamond$
        self-destruct
    else if $e' = e$
        output $\textsf{test}$
    else
        output $m'$
```

Fig. 6. System $\mathbf{G}_b^{\textsf{sd-cca}}$, where $b \in \{0, 1\}$, defining SD-CCA security of a PKE scheme $\Pi = (K, E, D)$. The command **self-destruct** causes the system to output $\diamond$ and to answer all future decryption queries by $\diamond$.

3.4 SD-CCA Security

In this section we formally define the notion of SD-CCA security and show how protocol m-pke can be seen as a PKE scheme Π that achieves SD-CCA security;[15] a proof can be found in the full version of this paper [14]. There we also provide a direct game-based proof of the fact that combining single-bit SD-CCA-secure PKE with a non-malleable code as shown above yields a multi-bit SD-CCA-secure PKE scheme. That proof is a hybrid argument and is obtained by "unwrapping" the concatenation of the statements in this section. The modular nature and the intuitive simplicity of the proofs are lost, however.

Definition of SD-CCA. The only difference between the SD-CCA game and the standard game used to define CCA is that the decryption oracle self-destructs, i.e., it stops processing further queries once an invalid ciphertext is queried. Note that the self-destruct feature only affects the decryption oracle; the adversary is still allowed to get the challenge ciphertext after provoking a self-destruct. The game is phrased as a distinguishing problem between the two systems $\mathbf{G}_0^{\textsf{sd-cca}}$ and $\mathbf{G}_1^{\textsf{sd-cca}}$ described in Figure 6.

Definition 4. *A PKE scheme $\Pi = (K, E, D)$ is (t, q, ε)-SD-CCA secure if*

$$\Delta^{\mathbf{D}}(\mathbf{G}_0^{\textsf{sd-cca}}, \mathbf{G}_1^{\textsf{sd-cca}}) \leq \varepsilon$$

for all distinguishers $\mathbf{D}$ with running time at most t and making at most q decryption queries.

The PKE scheme. The PKE scheme $\Pi = (K, E, D)$ corresponding to our protocol m-pke can be obtained as follows. The key generation algorithm K generates n independent key pairs of the 1-bit scheme. The encryption algorithm E first encodes a message using a non-malleable code and then encrypts each bit of the resulting encoding independently and outputs the n resulting ciphertexts. The decryption algorithm D first decrypts the n ciphertexts, decodes the resulting bitstring, and outputs the decoded message or the symbol $\diamond$, indicating an

[15] Actually, Π is only replayable SD-CCA secure; see below for details.

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

```
Key Generation $K'$
  for $i \leftarrow 1$ to $n$
    $(\mathsf{pk}_i, \mathsf{sk}_i) \leftarrow\$ K$
  $\mathsf{pk} \leftarrow (\mathsf{pk}_1, \dots, \mathsf{pk}_n)$
  $\mathsf{sk} \leftarrow (\mathsf{sk}_1, \dots, \mathsf{sk}_n)$
  return $(\mathsf{pk}, \mathsf{sk})$
```

```
Encryption $E'_{\mathsf{pk}}(m)$
  $c = c_1 \cdots c_n \leftarrow \mathrm{Enc}(m)$
  for $i \leftarrow 1$ to $n$
    $e_i \leftarrow\$ E_{\mathsf{pk}_i}(c_i)$
  return $e = (e_1, \dots, e_n)$
```

```
Decryption $D'_{\mathsf{sk}}(e)$
  for $i \leftarrow 1$ to $n$
    $c_i \leftarrow\$ D_{\mathsf{sk}_i}(e_i)$
    if $c_i = \diamond$
      return $\diamond$
  $m \leftarrow \mathrm{Dec}(c_1 \cdots c_n)$
  return $m$
```

Fig. 7. The k-bit PKE scheme $\Pi' = (K', E', D')$ built from a 1-bit PKE scheme $\Pi = (K, E, D)$ and a (k, n)-coding scheme (Enc, Dec).

invalid ciphertext, if any of these steps fails. The scheme is described in more detail in Figure 7.

Security of Π. In the full version of this paper [14], we show that Π is *replayable* SD-CCA secure. The notion of *replayable* CCA security (RCCA) in general was introduced by Canetti *et al.* [6] to deal with the artificial strictness of full CCA security. Roughly, RCCA security weakens full CCA security by potentially allowing an attacker to maul a ciphertext into one that decrypts to the identical message. The SD-CCA game $\mathbf{G}_b^{\textsf{sd-cca}}$ can be easily modified to a new game $\mathbf{G}_b^{\textsf{sd-rcca}}$, which behaves as $\mathbf{G}_b^{\textsf{sd-cca}}$, except that it outputs test whenever $D_{\mathsf{sk}}(e') \in \{m_0, m_1\}$ for a decryption query e'.

The reason that Π achieves only replayable SD-CCA security is that given any ciphertext e, an attacker can replace the first component of e by a fresh encryption of a randomly chosen bit and thereby obtain, with probability 1/2, a ciphertext $e' \neq e$ that decrypts to the same message as e. In [6], the authors provide generic ways to achieve full CCA security from replayable CCA security. As shown in subsequent work [12] to this paper, these techniques can also be applied in the context of SD-CCA security.[16]

4 Continuous Non-Malleability against $\mathcal{F}_{\mathsf{copy}}$

In this section, we describe a code that is adaptively continuously non-malleable w.r.t. $\mathcal{F}_{\mathsf{copy}}$. In the full version of this paper [14], we also provide a code secure w.r.t. to an extension $\mathcal{F}'_{\mathsf{copy}}$ of $\mathcal{F}_{\mathsf{copy}}$ that allows bit-flips as well.

The transition from continuous to *adaptive* continuous non-malleability w.r.t. $\mathcal{F}_{\mathsf{copy}}$ is achieved generically:

Theorem 3. *If a coding scheme* (Enc, Dec) *is continuously* $(\mathcal{F}_{\mathsf{copy}}, \varepsilon, 1, q)$*-non-malleable, it is also continuously* $(\mathcal{F}_{\mathsf{copy}}, 2\ell\varepsilon + \frac{q\ell}{2^k}, \ell, q)$*-non-malleable, for all* $\ell, q \in \mathbb{N}$.

The proof of Theorem 3 also appears in the full version of this paper. It remains to construct a continuously non-malleable code that is secure against tampering with a single encoding, which we do below.

[16] SD-CCA is called IND-SDA security in [12].

Continuous non-malleability for single encoding. The code is based on a linear error-correcting secret-sharing (LECSS). The use of a LECSS is inspired by the work of [24], who proposed a (non-continuous) non-malleable code against bit-wise tampering based on a LECSS and, additionally, a so-called AMD-code, which essentially handles bit-flips. As we do not need to provide non-malleability against bit-flips, using only the LECSS is sufficient for our purposes. The following definition is taken from [24]:[17]

Definition 5 (LECSS code). *A* (k, n)*-coding scheme* $(\mathrm{Enc}, \mathrm{Dec})$ *is a* (d, t)*-*linear error-correcting secret-sharing (LECSS) code *if the following properties hold:*

- Linearity: *For all* $c \in \{0,1\}^n$ *such that* $\mathrm{Dec}(c) \neq \bot$*, all* $\delta \in \{0,1\}^n$*, we have*
$$\mathrm{Dec}(c \oplus \delta) = \begin{cases} \bot & \textit{if } \mathrm{Dec}(\delta) = \bot \\ \mathrm{Dec}(c) \oplus \mathrm{Dec}(\delta) & \textit{otherwise.} \end{cases}$$
- Distance d: *For all* $c' \in \{0,1\}^n$ *with Hamming weight* $0 < w_H(c') < d$*, we have* $\mathrm{Dec}(c') = \bot$*.*
- Secrecy t: *For any fixed* $x \in \{0,1\}^k$*, the bits of* $\mathrm{Enc}(x)$ *are individually uniform and* t*-wise independent (over the randomness in the encoding).*

It turns out that a LECSS code is already continuously non-malleable with respect to $\mathcal{F}_{\mathsf{copy}}$:

Theorem 4. *Assume that* $(\mathrm{Enc}, \mathrm{Dec})$ *is a* (t, d)*-LECSS* (k, n)*-code for* $d > n/4$ *and* $d > t$*. Then* $(\mathrm{Enc}, \mathrm{Dec})$ *is* $(\mathcal{F}_{\mathsf{copy}}, \varepsilon, 1, q)$*-continuously non-malleable for all* $q \in \mathbb{N}$ *and*
$$\varepsilon = 2^{-(t-1)} + \left(\frac{t}{n(d/n - 1/4)^2}\right)^{t/2}.$$

For brevity, we write $\mathcal{F}_{\mathsf{set}}$ for $\mathcal{F}^{(1)}_{\mathsf{copy}}$ below, with the idea that the tampering functions in $\mathcal{F}^{(1)}_{\mathsf{copy}}$ only allow to keep a bit or to set it to 0 or to 1. More formally, a function $f \in \mathcal{F}_{\mathsf{set}}$ can be characterized by a vector $\chi(f) = (f_1, \ldots, f_n)$ where $f_i \in \{\mathsf{zero}, \mathsf{one}, \mathsf{keep}\}$, with the meaning that f takes as input a codeword c and outputs a codeword $c' = c'_1 \cdots c'_n$ in which each bit is either set to 0 (zero), set to 1 (one), or left unchanged (keep).

For the proof of Theorem 4, fix $q \in \mathbb{N}$ and some distinguisher $\mathbf{D}$. For the remainder of this section, let $\mathcal{F} := \mathcal{F}_{\mathsf{set}}$, $\mathbf{S}^{\mathsf{real}}_{\mathcal{F}} := \mathbf{S}^{\mathsf{real}}_{\mathcal{F},1,q}$ and $\mathbf{S}^{\mathsf{simu}}_{\mathcal{F},\tau} := \mathbf{S}^{\mathsf{simu}}_{\mathcal{F},\tau,1,q}$ (for a simulator τ to be determined). For a tamper query $f \in \mathcal{F}$ with $\chi(f) = (f_1, \ldots, f_n)$ issued by $\mathbf{D}$, let $A(f) := \{i \mid f_i \in \{\mathsf{zero}, \mathsf{one}\}\}$, $B(f) := \{i \mid f_i \in \{\mathsf{keep}\}\}$, and $a(f) := |A(f)|$. Moreover, let $\mathsf{val}(\mathsf{zero}) := 0$ and $\mathsf{val}(\mathsf{one}) := 1$. Queries f with $0 \leq a(f) \leq t$, $t < a(f) < n - t$, and $n - t \leq a(f) \leq n$ are called *low queries*, *middle queries*, and *high queries*, respectively.

[17] The operator $\oplus$ denotes the bit-wise XOR.

Handling Middle Queries. Consider the hybrid system $\mathbf{H}$ that proceeds as $\mathbf{S}_{\mathcal{F}}^{\mathsf{real}}$, except that as soon as $\mathbf{D}$ specifies a middle query f, $\mathbf{H}$ self-destructs, i.e., answers f and all subsequent queries by $\diamond$.

Lemma 1. $\Delta^{\mathbf{D}}(\mathbf{S}_{\mathcal{F}}^{\mathsf{real}}, \mathbf{H}) \leq \frac{1}{2^t} + \left(\frac{t}{n(d/n-1/4)^2}\right)^{t/2}$.

Proof. Define a *successful* middle query to be a middle query that does not decode to $\diamond$. On both systems $\mathbf{S}_{\mathcal{F}}^{\mathsf{real}}$ and $\mathbf{H}$, one can define an MBO $\mathcal{B}$ (cf. Section 2.2) that is provoked if and only if the *first* middle query is successful and the self-destruct has not been provoked up to that point.

Clearly, $\mathbf{S}_{\mathcal{F}}^{\mathsf{real}}$ and $\mathbf{H}$ behave identically until MBO $\mathcal{B}$ is provoked, thus $\hat{\mathbf{S}}_{\mathcal{F}}^{\mathsf{real}} \overset{g}{\equiv} \hat{\mathbf{H}}$, and

$$\Delta^{\mathbf{D}}(\mathbf{S}_{\mathcal{F}}^{\mathsf{real}}, \mathbf{H}) \leq \Gamma^{\mathbf{D}}(\hat{\mathbf{S}}_{\mathcal{F}}^{\mathsf{real}}).$$

Towards bounding $\Gamma^{\mathbf{D}}(\hat{\mathbf{S}}_{\mathcal{F}}^{\mathsf{real}})$, note first that adaptivity does not help in provoking $\mathcal{B}$: For any distinguisher $\mathbf{D}$, there exists a *non-adaptive* distinguisher $\mathbf{D}'$ with

$$\Gamma^{\mathbf{D}}(\hat{\mathbf{S}}_{\mathcal{F}}^{\mathsf{real}}) \leq \Gamma^{\mathbf{D}'}(\hat{\mathbf{S}}_{\mathcal{F}}^{\mathsf{real}}). \tag{6}$$

$\mathbf{D}'$ proceeds as follows: First, it (internally) interacts with $\mathbf{D}$ only. Initially, it stores the message x output by $\mathbf{D}$ internally. Whenever $\mathbf{D}$ outputs a low query, $\mathbf{D}'$ answers with x. Whenever $\mathbf{D}$ outputs a high query $f = (f_1, \ldots, f_n)$, $\mathbf{D}'$ checks whether there exists a codeword c^* that agrees with f in positions i where $f_i \in \{\mathsf{zero}, \mathsf{one}\}$. If it exists, it answers with $\mathrm{Dec}(c^*)$, otherwise with $\diamond$. As soon as $\mathbf{D}$ specifies a middle query, $\mathbf{D}'$ stops its interaction with $\mathbf{D}$ and sends x and all the queries to $\hat{\mathbf{S}}_{\mathcal{F}}^{\mathsf{real}}$.

To prove (6), fix all randomness in experiment $\mathbf{D}'\mathbf{S}_{\mathcal{F}}^{\mathsf{real}}$, i.e., the coins of $\mathbf{D}$ (inside $\mathbf{D}'$) and the randomness of the encoding (inside $\mathbf{S}_{\mathcal{F}}^{\mathsf{real}}$). Suppose $\mathbf{D}$ would provoke $\mathcal{B}$ in the direct interaction with $\mathbf{S}_{\mathcal{F}}^{\mathsf{real}}$. In that case all the answers by $\mathbf{D}'$ are equal to the answers by $\mathbf{S}_{\mathcal{F}}^{\mathsf{real}}$. This is due to the fact that the distance of the LECSS is $d > t$; a successful low query must therefore result in the original message x and a successful high query in $\mathrm{Dec}(c^*)$. Thus, whenever $\mathbf{D}$ provokes $\mathcal{B}$, $\mathbf{D}'$ provokes it as well.

It remains to analyze the success probability of non-adaptive distinguishers $\mathbf{D}'$. Fix the coins of $\mathbf{D}'$; this determines the tamper queries. Suppose there is at least one middle case, as otherwise $\mathcal{B}$ is trivially not provoked. The middle case's success probability can be analyzed as in [24, Theorem 4.1], which leads to

$$\Gamma^{\mathbf{D}'}(\hat{\mathbf{S}}_{\mathcal{F}}^{\mathsf{real}}) \leq \frac{1}{2^t} + \left(\frac{t}{n(d/n-1/4)^2}\right)^{t/2}$$

(recall that the MBO cannot be provoked after an unsuccessful first middle query). □

Simulator. The final step of the proof consists of exhibiting a simulator τ such that $\Delta^{\mathbf{D}}(\mathbf{H}, \mathbf{S}^{\mathsf{simu}}_{\mathcal{F},\tau})$ is small. The indistinguishability proof is facilitated by defining two hardly distinguishable systems $\mathbf{B}$ and $\mathbf{B}'$ and a wrapper system $\mathbf{W}$ such that $\mathbf{WB} \equiv \mathbf{H}$ and $\mathbf{WB}' \equiv \mathbf{S}^{\mathsf{simu}}_{\mathcal{F},\tau}$.

System $\mathbf{B}$ works as follows: Initially, it takes a value $x \in \{0,1\}^k$, computes an encoding $c_1 \cdots c_n \leftarrow\!\$\ \mathrm{Enc}(x)$ of it, and outputs λ (where the symbol λ indicates an empty output). 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, $\mathbf{B}$ returns $a_i = 1$. Otherwise, it outputs $a_i = \diamond$ and self-destructs (i.e., all future answers are $\diamond$). The system $\mathbf{B}'$ behaves as $\mathbf{B}$ except that the initial input x is ignored and the $c_1, \ldots, c_n$ are chosen uniformly at random and independently.

The behavior of $\mathbf{B}$ (and similarly the behavior of $\mathbf{B}'$) is described by a sequence $(\mathsf{p}^{\mathbf{B}}_{A^i|G^i})_{i \geq 0}$ of conditional probability distributions (cf. Section 2.2), where $\mathsf{p}^{\mathbf{B}}_{A^i|G^i}(a^i, g^i)$ is the probability of observing the outputs $a^i = (\lambda, a_1, \ldots, a_i)$ given the inputs $g^i = (x, g_1, \ldots, g_i)$. For simplicity, assume below that g^i is such that no position is guessed twice (a generalization is straight-forward) and that a^i is of the form $\{\lambda\}\{1\}^*\{\diamond\}^*$ (as otherwise it has probability 0 anyway).

For system $\mathbf{B}$, all i, and any g^i, $\mathsf{p}^{\mathbf{B}}_{A^i|G^i}(a^i, g^i) = 2^{-(s+1)}$ if a^i has $s < \min(i,t)$ leading 1's; this follows from the t-wise independence of the bits of $\mathrm{Enc}(x)$. All remaining output vectors a^i, i.e., those with at least $\min(i,t)$ preceding 1's, share a probability mass of $2^{-\min(i,t)}$, in a way that depends on the code in use and on x. (It is easily verified that this yields a valid probability distribution.) The behavior of $\mathbf{B}'$ is obvious given the above (simply replace "t" by "n" in the above description).

Lemma 2. $\Delta^{\mathbf{D}}(\mathbf{B}, \mathbf{B}') \leq 2^{-t}$.

Proof. On both systems $\mathbf{B}$ and $\mathbf{B}'$, one can define an MBO $\mathcal{B}$ that is zero as long as *less* than t positions have been guessed correctly. In the following, $\hat{\mathbf{B}}$ and $\hat{\mathbf{B}}'$ denote $\mathbf{B}$ and $\mathbf{B}'$ with the MBO, respectively.

Analogously to the above, the behavior of $\hat{\mathbf{B}}$ (and similarly that of $\hat{\mathbf{B}}'$) is described by a sequence $(\mathsf{p}^{\hat{\mathbf{B}}}_{A^i,B_i=0|G^i})_{i \geq 0}$ of conditional probability distributions, where $\mathsf{p}^{\hat{\mathbf{B}}}_{A^i,B_i=0|G^i}(a^i, g^i)$ is the probability of observing the outputs $a^i = (\lambda, a_1, \ldots, a_i)$ and $b_0 = b_1 = \ldots = b_i = 0$ given the inputs $g^i = (x, g_1, \ldots, g_i)$. One observes that due to the t-wise independence of $\mathrm{Enc}(x)$'s bits, for $i < t$,

$$\mathsf{p}^{\hat{\mathbf{B}}}_{A^i,B_i=0|G^i}(a^i, g^i) = \mathsf{p}^{\hat{\mathbf{B}}'}_{A^i,B_i=0|G^i}(a^i, g^i) = \begin{cases} 2^{-(s+1)} & \text{if } a^i \text{ has } s < i \text{ leading 1's,} \\ 2^{-i} & \text{if } a^i \text{ has } i \text{ leading 1's, and} \\ 0 & \text{otherwise,} \end{cases}$$

and for $i \geq t$,

$$\mathsf{p}^{\hat{\mathbf{B}}}_{A^i,B_i=0|G^i}(a^i, g^i) = \mathsf{p}^{\hat{\mathbf{B}}'}_{A^i,B_i=0|G^i}(a^i, g^i) = \begin{cases} 2^{-(s+1)} & \text{if } a^i \text{ has } s < t \text{ leading 1's,} \\ 0 & \text{otherwise.} \end{cases}$$

```
System $\mathbf{W}$

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

on first $(\mathsf{encode}, x)$ at $\mathtt{o}$
    out $x$ at $\mathtt{i}$

on $(\mathsf{tamper}, f)$ with $0 \leq a(f) \leq t$ at $\mathtt{o}$
    for $i$ where $f_i \in A(f)$
        $g \leftarrow \mathsf{val}(f_i)$
        if $c_i = \varnothing$
            out $(i, g)$ at $\mathtt{i}$
            get $a \in \{\diamond, 1\}$ at $\mathtt{i}$
            if $a = \diamond$
                self-destruct
            $c_i \leftarrow g$
        else
            if $c_i \neq g$
                self-destruct
    out $x$ at $\mathtt{out}$

on $(\mathsf{tamper}, f)$ with $t < a(f) < n - t$ at $\mathtt{o}$
    self-destruct

on $(\mathsf{tamper}, f)$ with $n - t \leq a(f) \leq n$ at $\mathtt{o}$
    for $i$ where $f_i \in A(f)$
        $c'_i \leftarrow \mathsf{val}(f_i)$
    if $\exists$ codeword $c^* : \forall i \in A(f) : c'_i = c^*_i$
        for $i$ where $f_i \in B(f)$
            $g \leftarrow c^*_i$
            if $c_i = \varnothing$
                out $(i, g)$ at $\mathtt{i}$
                get $a \in \{\diamond, 1\}$ at $\mathtt{i}$
                if $a = \diamond$
                    self-destruct
                $c_i \leftarrow g$
            else
                if $c_i \neq g$
                    self-destruct
    else
        self-destruct
    out $\mathrm{Dec}(c^*)$ at $\mathtt{out}$
```

Fig. 8. The wrapper system $\mathbf{W}$. The command **self-destruct** causes $\mathbf{W}$ to output $\diamond$ at $\mathtt{o}$ and to answer all future queries by $\diamond$. The symbol $\varnothing$ stands for "undefined."

Therefore, $\hat{\mathbf{B}} \overset{g}{\equiv} \hat{\mathbf{B}}'$ and $\Delta^{\mathbf{D}}(\mathbf{B}, \mathbf{B}') \leq \Gamma^{\mathbf{D}}(\hat{\mathbf{B}}')$. Observe that by an argument similar to the one above, adaptivity does not help in provoking the MBO of $\hat{\mathbf{B}}'$. Thus, $\Gamma^{\mathbf{D}}(\hat{\mathbf{B}}') \leq 2^{-t}$, since an optimal non-adaptive strategy simply tries to guess distinct positions. □

Recall that the purpose of the wrapper system $\mathbf{W}$ is to emulate $\mathbf{H}$ and $\mathbf{S}^{\mathsf{simu}}_{\mathcal{F},\tau}$ using $\mathbf{B}$ and $\mathbf{B}'$, respectively. The key point is to note that low queries f can be answered knowing only the positions $A(f)$ of $\mathrm{Enc}(x)$, high queries knowing only the positions in $B(f)$, and middle queries can always be rejected. A full description of $\mathbf{W}$ can be found in Figure 8. It has an outside interface $\mathtt{o}$ and an inside interface $\mathtt{i}$; at the latter interface, $\mathbf{W}$ expects to be connected to either $\mathbf{B}$ or $\mathbf{B}'$.

Lemma 3. $\mathbf{WB} \equiv \mathbf{H}$.

Proof. Since the distance of the LECSS is $d > t$, the following holds: A low query results in same if all injected positions match the corresponding bits of the encoding, and in $\diamond$ otherwise. Similarly, for a high query, there can be at most one codeword that matches the injected positions. If such a codeword c^* exists, the outcome is $\mathrm{Dec}(c^*)$ if the bits in the keep-positions match c^*, and otherwise $\diamond$. By inspection, it can be seen that $\mathbf{W}$ acts accordingly. □

Consider now the system $\mathbf{WB}'$. Due to the nature of $\mathbf{B}'$, the behavior of $\mathbf{WB}'$ is independent of the value x that is initially encoded. This allows to easily design a simulator τ as required by Definition 3. A full description of τ can be found in Figure 9.

```
Simulator $\tau$

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

on $(1, f)$ with $0 \le a(f) \le t$
    if $\forall i \in A(f) : \mathsf{val}(f_i) = c_i$
        return same
    else
        return $\diamond$

on $(1, f)$ with $t < a(f) < n - t$
    return $\diamond$

on $(1, f)$ with $n - t \le a(f) \le n$
    for $i$ where $f_i \in A(f)$
        $c'_i \leftarrow \mathsf{val}(f_i)$
    for $i$ where $f_i \in B(f)$
        $c'_i \leftarrow c_i$
    $c' \leftarrow c'_1 \cdots c'_n$
    return $\mathrm{Dec}(c')$
```

Fig. 9. The simulator τ.

Lemma 4. *The simulator τ of Figure 9 satisfies $\mathbf{WB}' \equiv \mathbf{S}^{\mathsf{simu}}_{\mathcal{F},\tau}$.*

Proof. Consider the systems $\mathbf{WB}'$ and $\mathbf{S}^{\mathsf{simu}}_{\mathcal{F},\tau}$. Both internally choose uniform and independent bits $c_1, \ldots, c_n$. System $\mathbf{WB}'$ answers low queries with the value x initially encoded if all injected positions match the corresponding random bits and with $\diamond$ otherwise. Simulator τ returns same in the former case, which $\mathbf{S}^{\mathsf{simu}}_{\mathcal{F},\tau}$ replaces by x, and $\diamond$ in the latter case.

Note that the answer by $\mathbf{WB}'$ to a high query f always matches $\mathrm{Dec}(c'_1 \cdots c'_n)$, where for $i \in A(f)$, $c'_i = \mathsf{val}(f_i)$, and for $i \in B(f)$, $c'_i = c_i$: If no codeword c^* matching the injected positions exists, then $\mathrm{Dec}(c'_1 \cdots c'_n) = \diamond$, which is also what $\mathbf{WB}'$ outputs. If such c^* exists and $c^*_i = c_i$ for all $i \in B(f)$, the output of $\mathbf{WB}'$ is $\mathrm{Dec}(c'_1 \cdots c'_n)$. If there exists an $i \in B(f)$ with $c^*_i \neq c_i$, $\mathbf{WB}'$ outputs $\diamond$, and in this case $\mathrm{Dec}(c'_1 \cdots c'_n) = \diamond$ since the distance of the LECSS is $d > t$. □

The proof of Theorem 4 now follows from a simple triangle inequality.

Proof (of Theorem 4). From Lemmas 1, 2, 3, and 4, one obtains that for all distinguishers $\mathbf{D}$,

$$\begin{aligned}
\Delta^{\mathbf{D}}(\mathbf{S}^{\mathsf{real}}_{\mathcal{F}}, \mathbf{S}^{\mathsf{simu}}_{\mathcal{F},\tau}) &\le \Delta^{\mathbf{D}}(\mathbf{S}^{\mathsf{real}}_{\mathcal{F}}, \mathbf{H}) + \underbrace{\Delta^{\mathbf{D}}(\mathbf{H}, \mathbf{WB})}_{=0} \\
&\quad + \underbrace{\Delta^{\mathbf{D}}(\mathbf{WB}, \mathbf{WB}')}_{=\Delta^{\mathbf{DW}}(\mathbf{B},\mathbf{B}')} + \underbrace{\Delta^{\mathbf{D}}(\mathbf{WB}', \mathbf{S}^{\mathsf{simu}}_{\mathcal{F},\tau})}_{=0} \\
&\le 2^{-t} + \left(\frac{t}{n(d/n - 1/4)^2}\right)^{t/2} + 2^{-t} \\
&\le 2^{-(t-1)} + \left(\frac{t}{n(d/n - 1/4)^2}\right)^{t/2}.
\end{aligned}$$

□

5 On the Necessity of Self-Destruct

In this section we show that no (k, n)-coding scheme $(\mathrm{Enc}, \mathrm{Dec})$ can achieve (even non-adaptive, i.e. for $\ell = 1$) continuous non-malleability against $\mathcal{F}_{\mathsf{copy}}$

without self-destruct. This fact is reminiscent of the negative result by Gennaro *et al.* [30], and was already observed by Faust *et al.* [26] (without a proof) for the easier case of *strong* continuous non-malleability. The impossibility proof in this section assumes that Dec is deterministic and that $\mathrm{Dec}(\mathrm{Enc}(x)) = x$ with probability 1 for all $x \in \{0,1\}^k$ (cf. Definition 2). The distinguisher $\mathbf{D}$ provided by Theorem 5 is universal, i.e., it breaks any coding scheme (if given oracle access to its decoding algorithm).

For the remainder of this section, let $\mathcal{F} := \mathcal{F}_{\mathsf{set}}$ (as defined in Section 4), $\mathbf{S}^{\mathsf{real}}_{\mathcal{F}} := \mathbf{S}^{\mathsf{real}}_{\mathcal{F},1,n}$, and $\mathbf{S}^{\mathsf{simu}}_{\mathcal{F},\tau} := \mathbf{S}^{\mathsf{simu}}_{\mathcal{F},\tau,1,n}$ (with some simulator τ). Moreover, both $\mathbf{S}^{\mathsf{real}}_{\mathcal{F}}$ and $\mathbf{S}^{\mathsf{simu}}_{\mathcal{F},\tau}$ are stripped of the self-destruct mode.

Theorem 5. *There exists a distinguisher* $\mathbf{D}$ *such that for all coding schemes* $(\mathrm{Enc}, \mathrm{Dec})$ *and all simulators* τ*,*

$$\Delta^{\mathbf{D}}(\mathbf{S}^{\mathsf{real}}_{\mathcal{F}}, \mathbf{S}^{\mathsf{simu}}_{\mathcal{F},\tau}) \geq 1 - \frac{n+1}{2^k}.$$

The corollary below states no pair of converters $(\mathsf{encode}, \mathsf{decode})$ can achieve the constructive statement corresponding to Theorem 2 without relying on the self-destruct feature.

Corollary 1. *For any protocol* $\mathsf{nmc} := (\mathsf{encode}, \mathsf{decode})$ *and all simulators* σ*, if both converters are stateless and*

$$[\overset{\text{1-bit}}{-\!\diamond\!\!\twoheadrightarrow\!\bullet}]^n \quad \overset{((\mathsf{encode},\mathsf{decode}),\sigma,(0,\varepsilon))}{\Longmapsto} \quad \overset{k\text{-bit}}{-\!\diamond\!\!\twoheadrightarrow\!\bullet},$$

then,

$$\varepsilon \geq 1 - \frac{n+1}{2^k}.$$

Proof. Note that the protocol achieves perfect availability and thus constitutes a perfectly correct (k, n)-coding scheme (since the converters are stateless and with perfect correctness, decode can w.l.o.g. be assumed to be deterministic). Consider an arbitrary simulator σ. It can be converted into a simulator τ as required by Definition 3 in a straight-forward manner. Similarly, there exists a straight-forward reduction $\mathbf{C}$ such that

$$\mathbf{C}(\mathsf{encode}^A \mathsf{decode}^B [\overset{\text{1-bit},1,n}{-\!\diamond\!\!\twoheadrightarrow\!\bullet}]^n) \equiv \mathbf{S}^{\mathsf{real}}_{\mathcal{F}} \qquad \text{and} \qquad \mathbf{C}(\sigma^E \overset{k\text{-bit},1,n}{-\!\diamond\!\!\twoheadrightarrow\!\bullet}) \equiv \mathbf{S}^{\mathsf{simu}}_{\mathcal{F},\tau}.$$

Thus, $\mathbf{DC}$ achieves advantage $1 - \frac{n+1}{2^k}$. □

5.1 Proof of Theorem 5

Distinguisher $\mathbf{D} := \mathbf{D}_{\mathsf{Ext}}$ uses an algorithm Ext that always extracts the encoded message when interacting with system $\mathbf{S}^{\mathsf{real}}_{\mathcal{F}}$ and does so with small probability only when interacting with system $\mathbf{S}^{\mathsf{simu}}_{\mathcal{F},\tau}$ (for any simulator).

The Extraction Algorithm. Consider the following algorithm Ext, which repeatedly issues tamper queries (tamper, f) with $f \in \mathcal{F}_{\mathsf{set}}$, expects an answer in $\{0,1\}^k \cup \{\diamond, \mathsf{same}\}$, and eventually outputs a value $x' \in \{0,1\}^k$: Initially, it initializes variables $f_1, \ldots, f_n \leftarrow \varnothing$ (where the value $\varnothing$ stands for "undefined"). Then, for $i = 1, \ldots, n$ it proceeds as follows: It queries (tamper, f) with $\chi(f) = (f_1, \ldots, f_{i-1}, \mathsf{zero}, \mathsf{keep}, \ldots, \mathsf{keep})$. If the answer is same, it sets $f_i \leftarrow \mathsf{zero}$ and otherwise $f_i \leftarrow \mathsf{one}$. In the end Ext outputs $x' \leftarrow \mathrm{Dec}(\mathsf{val}(f_1) \cdots \mathsf{val}(f_n))$.

The Distinguisher. Consider the following distinguisher $\mathbf{D}_{\mathsf{Ext}}$: Initially, it chooses $x \leftarrow \{0,1\}^k$ and outputs (encode, x) to the system it is connected to. Then, it lets Ext interact with that system, replacing an answer by same whenever it is x. When Ext terminates and outputs a value x', $\mathbf{D}_{\mathsf{Ext}}$ outputs 1 if $x' = x$ and 0 otherwise.

Lemma 5. $\mathsf{P}[\mathbf{D}_{\mathsf{Ext}}\mathbf{S}^{\mathsf{real}}_{\mathcal{F}} = 1] = 1$.

Proof. Assume that before the i^{th} iteration of Ext, asking the query (tamper, f) with $\chi(f) = (f_1, \ldots, f_{i-1}, \mathsf{keep}, \mathsf{keep}, \ldots, \mathsf{keep})$ to $\mathbf{S}^{\mathsf{real}}_{\mathcal{F}}$ yields the answer x. From this it follows that either $(f_1, \ldots, f_{i-1}, \mathsf{zero}, \mathsf{keep}, \ldots, \mathsf{keep})$ or $(f_1, \ldots, f_{i-1}, \mathsf{one}, \mathsf{keep}, \ldots, \mathsf{keep})$ leads to the answer x; Ext sets f_i appropriately (the fact that the answer x is replaced by same plays no role here). Thus, in the end, computing $\mathrm{Dec}(\mathsf{val}(f_1) \cdots \mathsf{val}(f_n))$ yields x. □

In other words, Lemma 5 means that Ext always succeeds at recovering the value x chosen by $\mathbf{D}$. Showing that this happens only with small probability when $\mathbf{D}_{\mathsf{Ext}}$ interacts with $\mathbf{S}^{\mathsf{simu}}_{\mathcal{F},\tau}$ completes the proof.

Lemma 6. $\mathsf{P}[\mathbf{D}_{\mathsf{Ext}}\mathbf{S}^{\mathsf{simu}}_{\mathcal{F},\tau} = 1] \leq \frac{n+1}{2^k}$.

Proof. Consider the following modified distinguisher $\hat{\mathbf{D}}_{\mathsf{Ext}}$ that works as $\mathbf{D}_{\mathsf{Ext}}$ except that it does *not* modify the answers received by the system it is connected to. Moreover, let $\hat{\mathbf{S}}^{\mathrm{simu}}_{\mathcal{F},\tau}$ be the the system that ignores all encode-queries and handles queries (tamper, f) by invoking $\tau(1, f)$ and outputting τ's answer.

Note that in both experiments, Ext's view is identical unless it causes τ to output x (the value encoded by $\mathbf{D}$), which happens with probability at most $\frac{n}{2^k}$. Thus,

$$|\mathsf{P}^{\mathbf{D}_{\mathsf{Ext}}\mathbf{S}^{\mathsf{simu}}_{\mathcal{F},\tau}}[\mathsf{Ext} \text{ outputs } x] - \mathsf{P}^{\hat{\mathbf{D}}_{\mathsf{Ext}}\hat{\mathbf{S}}^{\mathrm{simu}}_{\mathcal{F},\tau}}[\mathsf{Ext} \text{ outputs } x]| \leq \frac{n}{2^k}.$$

Furthermore, in experiment $\hat{\mathbf{D}}_{\mathsf{Ext}}\hat{\mathbf{S}}^{\mathrm{simu}}_{\mathcal{F},\tau}$, Ext's view is independent of x, and therefore, x is output by Ext with probability $\frac{1}{2^k}$. The claim follows. □

Acknowledgments. We thank Yevgeniy Dodis for insightful discussions and for suggesting many improvements to the paper. We thank Joël Alwen and Daniel Tschudi for helpful discussions, in particular on the impossibility proof in Section 5. The work was supported by the Swiss National Science Foundation (SNF), project no. 200020-132794. Björn Tackmann is supported by the Swiss National Science Foundation (SNF).

References

1. Divesh Aggarwal, Yevgeniy Dodis, Tomasz Kazana, and Maciej Obremski. Non-malleable reductions and applications. Cryptology ePrint Archive, Report 2014/821, 2014. `http://eprint.iacr.org/`.
2. Divesh Aggarwal, Yevgeniy Dodis, and Shachar Lovett. Non-malleable codes from additive combinatorics. In *STOC*, pages 774–783, 2014.
3. Divesh Aggarwal, Stefan Dziembowski, Tomasz Kazana, and Maciej Obremski. Leakage-resilient non-malleable codes. Cryptology ePrint Archive, Report 2014/807, 2014. `http://eprint.iacr.org/`. To appear in TCC 2015.
4. Shashank Agrawal, Divya Gupta, Hemanta K. Maji, Omkant Pandey, and Manoj Prabhakaran. Explicit non-malleable codes resistant to permutations and perturbations. Cryptology ePrint Archive, Report 2014/841, 2014. `http://eprint.iacr.org/`.
5. Shashank Agrawal, Divya Gupta, Hemanta K. Maji, Omkant Pandey, and Manoj Prabhakaran. Explicit optimal-rate non-malleable codes against bit-wise tampering and permutations. Cryptology ePrint Archive, Report 2014/842, 2014. `http://eprint.iacr.org/`. To appear in TCC 2015.
6. Ran Canetti, Hugo Krawczyk, and Jesper Buus Nielsen. Relaxing chosen-ciphertext security. In *CRYPTO*, pages 565–582, 2003.
7. Eshan Chattopadhyay and David Zuckerman. Non-malleable codes against constant split-state tampering. In *FOCS*, pages 306–315, 2014.
8. Mahdi Cheraghchi and Venkatesan Guruswami. Capacity of non-malleable codes. In *ITCS*, pages 155–168, 2014.
9. Mahdi Cheraghchi and Venkatesan Guruswami. Non-malleable coding against bit-wise and split-state tampering. In *TCC*, pages 440–464, 2014.
10. Seung Geol Choi, Dana Dachman-Soled, Tal Malkin, and Hoeteck Wee. Black-box construction of a non-malleable encryption scheme from any semantically secure one. In *TCC*, pages 427–444, 2008.
11. Seung Geol Choi, Aggelos Kiayias, and Tal Malkin. BiTR: Built-in tamper resilience. In *ASIACRYPT*, pages 740–758, 2011.
12. Sandro Coretti, Yevgeniy Dodis, Björn Tackmann, and Daniele Venturi. Self-destruct non-malleability. *IACR Cryptology ePrint Archive*, 2014:866, 2014.
13. Sandro Coretti, Ueli Maurer, and Björn Tackmann. Constructing confidential channels from authenticated channels - public-key encryption revisited. In *ASIACRYPT (1)*, pages 134–153, 2013.
14. Sandro Coretti, Ueli Maurer, Björn Tackmann, and Daniele Venturi. From single-bit to multi-bit public-key encryption via non-malleable codes. *IACR Cryptology ePrint Archive*, 2014:324, 2014.
15. Ronald Cramer, Yevgeniy Dodis, Serge Fehr, Carles Padró, and Daniel Wichs. Detection of algebraic manipulation with applications to robust secret sharing and fuzzy extractors. In *EUROCRYPT*, pages 471–488, 2008.
16. 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.
17. Ronald Cramer and Victor Shoup. A practical public key cryptosystem provably secure against adaptive chosen ciphertext attack. In Hugo Krawczyk, editor, *CRYPTO 1998*, volume 1462 of *LNCS*, pages 13–25, Heidelberg, 1998. Springer.
18. 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.

19. Dana Dachman-Soled, Feng-Hao Liu, Elaine Shi, and Hong-Sheng Zhou. Locally decodable and updatable non-malleable codes and their applications. Cryptology ePrint Archive, Report 2014/663, 2014. `http://eprint.iacr.org/`. To appear in TCC 2015.
20. Ivan Damgård, Sebastian Faust, Pratyay Mukherjee, and Daniele Venturi. Bounded tamper resilience: How to go beyond the algebraic barrier. In *ASIACRYPT (2)*, pages 140–160, 2013.
21. Ivan Damgård, Sebastian Faust, Pratyay Mukherjee, and Daniele Venturi. The chaining lemma and its application. Cryptology ePrint Archive, Report 2014/979, 2014. `http://eprint.iacr.org/`.
22. Whitfield Diffie and Martin E. Hellman. New directions in cryptography. *IEEE Transactions on Information Theory*, 22(6):644–654, 1976.
23. Stefan Dziembowski, Tomasz Kazana, and Maciej Obremski. Non-malleable codes from two-source extractors. In *CRYPTO (2)*, pages 239–257, 2013.
24. Stefan Dziembowski, Krzysztof Pietrzak, and Daniel Wichs. Non-malleable codes. In *ICS*, pages 434–452, 2010.
25. Taher ElGamal. A public key cryptosystem and a signature scheme based on discrete logarithms. In *CRYPTO*, pages 10–18, 1984.
26. Sebastian Faust, Pratyay Mukherjee, Jesper Buus Nielsen, and Daniele Venturi. Continuous non-malleable codes. In *TCC*, pages 465–488, 2014.
27. Sebastian Faust, Pratyay Mukherjee, Jesper Buus Nielsen, and Daniele Venturi. A tamper and leakage resilient von Neumann architecture. *IACR Cryptology ePrint Archive*, 2014:338, 2014. To appear in PKC 2015.
28. 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.
29. Eiichiro Fujisaki, Tatsuaki Okamoto, David Pointcheval, and Jacques Stern. RSA-OAEP is secure under the RSA assumption. In Joe Kilian, editor, *CRYPTO 2001*, volume 2139 of *LNCS*, pages 260–274, Heidelberg, 2001. Springer.
30. 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.
31. 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.
32. Shafi Goldwasser and Silvio Micali. Probabilistic encryption. *J. Comput. Syst. Sci.*, 28(2):270–299, 1984.
33. Dennis Hofheinz and Eike Kiltz. Practical chosen ciphertext secure encryption from factoring. In Antoine Joux, editor, *EUROCRYPT 2009*, volume 5479 of *LNCS*, pages 313–332, Heidelberg, 2009. Springer.
34. Susan Hohenberger, Allison B. Lewko, and Brent Waters. Detecting dangerous queries: A new approach for chosen ciphertext security. In *EUROCRYPT*, pages 663–681, 2012.
35. Zahra Jafargholi and Daniel Wichs. Tamper detection and continuous non-malleable codes. Cryptology ePrint Archive, Report 2014/956, 2014. `http://eprint.iacr.org/`. To appear in TCC 2015.
36. Eike Kiltz, Krzysztof Pietrzak, Martijn Stam, and Moti Yung. A new randomness extraction paradigm for hybrid encryption. In Antoine Joux, editor, *EUROCRYPT 2009*, volume 5479 of *LNCS*, pages 590–609, Heidelberg, 2009. Springer.
37. Markulf Kohlweiss, Ueli Maurer, Cristina Onete, Björn Tackmann, and Daniele Venturi. Anonymity-preserving public-key encryption: A constructive approach. In *Privacy Enhancing Technologies*, pages 19–39, 2013.

38. Huijia Lin and Stefano Tessaro. Amplification of chosen-ciphertext security. In *EUROCRYPT*, pages 503–519, 2013.
39. Feng-Hao Liu and Anna Lysyanskaya. Tamper and leakage resilience in the split-state model. In *CRYPTO*, pages 517–532, 2012.
40. Ueli Maurer. Constructive cryptography - a new paradigm for security definitions and proofs. In *TOSCA*, pages 33–56, 2011.
41. Ueli Maurer. Conditional equivalence of random systems and indistinguishability proofs. In *2013 IEEE International Symposium on Information Theory Proceedings (ISIT)*, pages 3150–3154, 2013.
42. Ueli Maurer and Renato Renner. Abstract cryptography. In *ICS*, pages 1–21, 2011.
43. Ueli Maurer, Andreas Rüedlinger, and Björn Tackmann. Confidentiality and integrity: A constructive perspective. In *TCC*, pages 209–229, 2012.
44. Ueli Maurer and Pierre Schmid. A calculus for security bootstrapping in distributed systems. *Journal of Computer Security*, 4(1):55–80, 1996.
45. Ueli Maurer and Björn Tackmann. On the soundness of authenticate-then-encrypt: formalizing the malleability of symmetric encryption. In *ACM Conference on Computer and Communications Security*, pages 505–515, 2010.
46. Ueli M. Maurer. Indistinguishability of random systems. In *EUROCRYPT*, pages 110–132, 2002.
47. 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.
48. Steven Myers and Abhi Shelat. Bit encryption is complete. In *FOCS*, pages 607–616, 2009.
49. Chris Peikert and Brent Waters. Lossy trapdoor functions and their applications. *SIAM J. Comput.*, 40(6):1803–1844, 2011.
50. 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.
51. Alon Rosen and Gil Segev. Chosen-ciphertext security via correlated products. *SIAM J. Comput.*, 39(7):3058–3088, 2010.