Non-Malleable Codes for Partial Functions with Manipulation Detection

Aggelos Kiayias[1], Feng-Hao Liu[2], and Yiannis Tselekounis[1(✉)]

[1] University of Edinburgh, Edinburgh, UK
akiayias@inf.ed.ac.uk, ytselekounis@ed.ac.uk
[2] Florida Atlantic University, Boca Raton, USA
fenghao.liu@fau.edu

Abstract. Non-malleable codes were introduced by Dziembowski, Pietrzak and Wichs (ICS '10) and its main application is the protection of cryptographic devices against tampering attacks on memory. In this work, we initiate a comprehensive study on non-malleable codes for the class of partial functions, that read/write on an arbitrary subset of codeword bits with specific cardinality. Our constructions are efficient in terms of information rate, while allowing the attacker to access asymptotically almost the entire codeword. In addition, they satisfy a notion which is stronger than non-malleability, that we call non-malleability with manipulation detection, guaranteeing that any modified codeword decodes to either the original message or to $\perp$. Finally, our primitive implies All-Or-Nothing Transforms (AONTs) and as a result our constructions yield efficient AONTs under standard assumptions (only one-way functions), which, to the best of our knowledge, was an open question until now. In addition to this, we present a number of additional applications of our primitive in tamper resilience.

1 Introduction

Non-malleable codes (NMC) were introduced by Dziembowski, Pietrzak and Wichs [27] as a relaxation of error correction and error detection codes, aiming to provide strong privacy but relaxed correctness. Informally, non-malleability guarantees that any modified codeword decodes either to the original message or to a completely unrelated one, with overwhelming probability. The definition of non-malleability is simulation-based, stating that for any tampering function f, there exists a simulator that simulates the tampering effect by only accessing f, i.e., without making any assumptions on the distribution of the encoded message.

The main application of non-malleable codes that motivated the seminal work by Dziembowski et al. [27] is the protection of cryptographic implementations

A. Kiayias—Research partly supported by the H2020 project FENTEC (# 780108).
F.-H. Liu—Research supported by the NSF Award #CNS-1657040.
Y. Tselekounis—Research partly supported by the H2020 project PANORAMIX (# 653497).

H. Shacham and A. Boldyreva (Eds.): CRYPTO 2018, LNCS 10993, pp. 577–607, 2018.
https://doi.org/10.1007/978-3-319-96878-0_20

from *active physical attacks* against memory, known as *tampering attacks*. In this setting, the adversary modifies the memory of the cryptographic device, receives the output of the computation, and tries to extract sensitive information related to the private memory. Security against such types of attacks can be achieved by encoding the private memory of the device using non-malleable codes. Besides that, various applications of non-malleable codes have been proposed in subsequent works, such as CCA secure encryption schemes [20] and non-malleable commitments [4].

Due to their important applications, constructing non-malleable codes has received a lot of attention over recent years. As non-malleability against general functions is impossible [27], various subclasses of tampering functions have been considered, such as split-state functions [1–3,26,27,36,37], bit-wise tampering and permutations [4,5,27], bounded-size function classes [32], bounded depth/fan-in circuits [6], space-bounded tampering [29], and others (cf. Sect. 1.4). One characteristic shared by those function classes is that they allow *full access* to the codeword, while imposing structural or computational restrictions to the way the function computes over the input. In this work we initiate a comprehensive study on non-malleability for functions that receive *partial access* over the codeword, which is an important yet overlooked class, as we elaborate below.

The class of partial functions. The class of *partial functions* contains all functions that read/write on an arbitrary subset of codeword bits with specific cardinality. Concretely, let c be a codeword with length ν. For $\alpha \in [0, 1)$, the function class $\mathcal{F}^{\alpha\nu}$ (or $\mathcal{F}^{\alpha}$ for brevity) consists of all functions that operate over any subset of bits of c with cardinality at most $\alpha\nu$, while leaving the remaining bits intact. The work of Cheraghchi and Guruswami [18] explicitly defines this class and uses a subclass (the one containing functions that always touch the first $\alpha\nu$ bits of the codeword) in a negative way, namely as the tool for deriving capacity lower bounds for *information-theoretic* non-malleable codes against split-state functions. Partial functions were also studied implicitly by Faust et al. [32], while aiming for non-malleability against bounded-size circuits.[1]

Even though capacity lower bounds for partial functions have been derived (cf. [18]), our understanding about *explicit* constructions is still limited. Existential results can be derived by the probabilistic method, as shown in prior works [18,27][2], but they do not yield explicit constructions. On the other hand, the capacity bounds do not apply to the computational setting, which could potentially allow more practical solutions. We believe that this is a direction that needs to be explored, as besides the theoretical interest, partial functions is

[1] Specifically, in [32], the authors consider a model where a common reference string (CRS) is available, with length roughly logarithmic in the size of the tampering function class; as a consequence, the tampering function is allowed to read/write the whole codeword while having only partial information over the CRS.

[2] Informally, prior works [18,27] showed existence of non-malleable codes for classes of certain bounded cardinalities. The results cover the class of partial functions.

a natural model that complies with existing attacks that require partial access to the registers of the cryptographic implementation [8,10–12,44].[3]

Besides the importance of partial functions in the active setting, i.e., when the function is allowed to partially *read*/*write* the codeword, the passive analogue of the class, i.e., when the function is only given *read access* over the codeword, matches the model considered by All-Or-Nothing Transforms (AONTs), which is a notion originally introduced by Rivest [41], providing security guarantees similar to those of leakage resilience: reading an arbitrary subset (up to some bounded cardinality) of locations of the codeword does not reveal the underlying message. As non-malleable codes provide privacy, non-malleability for partial functions is the active analogue of (and in fact implies) AONTs, that find numerous applications [13,14,40,41,43].

Plausibility. At a first glance one might think that partial functions better comply with the framework of error-correction/detection codes (ECC/EDC), as they do not touch the whole codeword. However, if we allow the adversary to access asymptotically almost the entire codeword, it is conceivable it can use this generous *access rate*, i.e., the fraction of the codeword that can be accessed (see below), to create correlated encodings, thus we believe solving non-malleability in this setting is a natural question. Additionally, non-malleability provides simulation based security, which is not considered by ECC/EDC.

We illustrate the separation between the notions using the following example. Consider the set of partial functions that operate either on the right or on the left half of the codeword (the function chooses if it is going to be left or right), and the trivial encoding scheme that on input message s outputs (s, s). The decoder, on input (s, s'), checks if $s = s'$, in which case it outputs s, otherwise it outputs $\perp$. This scheme is clearly an EDC against the aforementioned function class,[4] as the output of the decoder is in $\{s, \perp\}$, with probability 1; however, it is malleable since the tampering function can create encodings whose validity depends on the message. On the other hand, an ECC would provide a trivial solution in this setting, however it requires restriction of the adversarial access fraction to 1/2 (of the codeword); by accessing more than this fraction, the attacker can possibly create invalid encodings depending on the message, as general ECCs do not provide privacy. Thus, the ECC/EDC setting is inapt when aiming for simulation based security in the presence of attackers that access almost the entire codeword. Later in this section, we provide an extensive discussion on challenges of non-malleability for partial functions.

Besides the plausibility and the lack of a comprehensive study, partial functions can potentially allow stronger primitives, as constant functions are excluded from the class. This is similar to the path followed by Jafargholi and Wichs [34], aiming to achieve *tamper detection* (cf. Sect. 1.4) against a class of

[3] The attacks by [8,11,12] require the modification of a single (random) memory bit, while in [10] a single error per each round of the computation suffices. In [44], the attack requires a single faulty byte.

[4] It is not an ECC as the decoder does not know which side has been modified by the tampering function.

functions that implicitly excludes constant functions and the identity function. In this work we prove that this intuition holds, by showing that partial functions allow a stronger primitive that we define as *non-malleability* with *manipulation detection* (MD-NMC), which in addition to simulation based security, it also guarantees that any tampered codeword will either decode to the original message or to $\perp$. Again, and as in the case of ECC/EDC, we stress out that manipulation/tamper-detection codes do not imply MD-NMC, as they do not provide simulation based security (cf. Sect. 1.4).[5]

Given the above, we believe that partial functions is an interesting and well-motivated model. The goal of this work is to answer the following (informally stated) question:

> *Is it possible to construct efficient (high information rate) non-malleable codes for partial functions, while allowing the attacker to access almost the entire codeword?*

We answer the above question in the affirmative. Before presenting our results (cf. Sect. 1.1) and the high level ideas behind our techniques (cf. Sect. 1.2), we identify the several challenges that are involved in tackling the problem.

Challenges. We first define some useful notions used throughout the paper.

- *Information rate*: the ratio of message to codeword length, as the message length goes to infinity.
- *Access rate*: the fraction of the number of bits that the attacker is allowed to access over the total codeword length, as the message length goes to infinity.

The access rate measures the effectiveness of a non-malleable code in the partial function setting and reflects the level of adversarial access to the codeword. In this work, we aim at constructing non-malleable codes for partial functions with high *information rate* and high *access rate*, i.e., both rates should approach 1 simultaneously. Before discussing the challenges posed by this requirement, we first review some known impossibility results. First, non-malleability for partial functions with concrete access rate 1 is impossible, as the function can fully decode the codeword and then re-encode a related message [27]. Second, information-theoretic non-malleable codes with constant information rate (e.g., 0.5) are not possible against partial functions with constant access rate [18][6], and consequently, solutions in the information-theoretic settings such as ECC and Robust Secret Sharing (RSS) do not solve our problem. Based on these facts, in order to achieve our goal, the only path is to explore the computational setting, aiming for access rate at most $1 - \epsilon$, for some $\epsilon > 0$.

At a first glance one might think that non-malleability for partial functions is easier to achieve, compared to other function classes, as partial functions

[5] Clearly, MD-NMC imply manipulation/error-detection codes.

[6] Informally, in [18] (Theorem 5.3) the authors showed that any information-theoretic non-malleable code with a constant access rate and a constant information rate must have a constant distinguishing probability.

cannot touch the whole codeword. Having that in mind, it would be tempting to conclude that existing designs/techniques with minor modifications are sufficient to achieve our goal. However, we will show that this intuition is misleading, by pointing out why prior approaches fail to provide security against partial functions with high access rate.

The current state of the art in the computational setting considers tools such as (Authenticated) Encryption [1,22,24,28,36,37], *non-interactive zero-knowledge* (NIZK) proofs [22,28,30,37], and ℓ-more extractable collision resistant hashes (ECRH) [36], where others use KEM/DEM techniques [1,24]. Those constructions share a common structure, incorporating a short secret key sk (or a short encoding of it), as well as a long ciphertext, e, and a proof π (or a hash value). Now, consider the partial function f that gets full access to the secret key sk and a constant number of bits of the ciphertext e, partially decrypts e and modifies the codeword depending on those bits. Then, it is not hard to see that non-malleability falls apart as the security of the encryption no longer holds. The attack requires access rate only $O((|sk|)/(|sk| + |e| + |\pi|))$, for [22,28,37] and $O(\text{poly}(k)/|s|)$ for [1,24,36]. A similar attack applies to [30], which is in the continual setting.

One possible route to tackle the above challenges, is to use an encoding scheme over the ciphertext, such that partial access over it does not reveal the underlying message.[7] The guarantees that we need from such a primitive resemble the properties of AONTs, however this primitive does not provide security against active, i.e., tampering, attacks. Another approach would be to use Reconstructable Probabilistic Encodings [6], which provide error-correcting guarantees, yet still it is unknown whether we can achieve information rate 1 for such a primitive. In addition, the techniques and tools for protecting the secret key can be used to achieve optimal information rate as they are independent of the underlying message, yet at the same time, they become the weakest point against partial functions with high access rate. Thus, the question is how to overcome the above challenges, allowing access to almost the entire codeword.

In this paper we solve the challenges presented above based on the following observation: in existing solutions the structure of the codeword is fixed and known to the attacker, and independently of the primitives that we use, the only way to resolve the above issues is by hiding the structure via randomization. This requires a structure recovering mechanism that can either be implemented by an "external" source, or otherwise the structure needs to be reflected in the codeword in some way that the attacker cannot exploit. In the present work we implement this mechanism in both ways, by first proposing a construction in the *common reference string* (CRS) model, and then we show how to remove the CRS using slightly bigger alphabets. Refer to Sect. 1.2 for a technical overview.

[7] In the presence of NIZKs we can have attacks with low access rate that read sk, e, and constant number of bits from the proof.

1.1 Our Results

We initiate the study of *non-malleable codes* with *manipulation-detection* (MD-NMC), and we present the first (to our knowledge) construction for this type of codes. We focus on achieving simultaneously high *information rate* and high *access rate*, in the partial functions setting, which by the results of [18], it can be achieved only in the computational setting.

Our contribution is threefold. First, we construct an information rate 1 non-malleable code in the CRS model, with access rate $1 - 1/\Omega(\log k)$, where k denotes the security parameter. Our construction combines Authenticated Encryption together with an inner code that protects the key of the encryption scheme (cf. Sect. 1.2). The result is informally summarized in the following theorem.

Theorem 1.1 (Informal)**.** *Assuming one-way functions, there exists an explicit computationally secure* MD-NMC *in the* CRS *model, with information rate* 1 *and access rate* $1 - 1/\Omega(\log k)$*, where* k *is the security parameter.*

Our scheme, in order to achieve security with error $2^{-\Omega(k)}$, produces codewords of length $|s| + O(k^2 \log k)$, where $|s|$ denotes the length of the message, and uses a CRS of length $O(k^2 \log k \log(|s| + k))$. We note that our construction does not require the CRS to be fully tamper-proof and we refer the reader to Sect. 1.2 for a discussion on the topic.

In our second result we show how to remove the CRS by slightly increasing the size of the alphabet. Our result is a computationally secure MD-NMC in the standard model, achieving information and access rate $1 - 1/\Omega(\log k)$. Our construction is proven secure by a reduction to the security of the scheme presented in Theorem 1.1. Below, we informally state our result.

Theorem 1.2 (Informal)**.** *Assuming one-way functions, there exists an explicit, computationally secure* MD-NMC *in the standard model, with alphabet length* $O(\log k)$*, information rate* $1 - 1/\Omega(\log k)$ *and access rate* $1 - 1/\Omega(\log k)$*, where* k *is the security parameter.*

Our scheme produces codewords of length $|s|(1 + 1/O(\log k)) + O(k^2 \log^2 k)$.

In Sect. 1.2, we consider security against continuous attacks. We show how to achieve a weaker notion of continuous security, while avoiding the use of a self-destruct mechanism, which was originally achieved by [28]. Our notion is weaker than full continuous security [30], since the codewords need to be updated. Nevertheless, our update operation is deterministic and avoids the full re-encoding process [27,37]; it uses only shuffling and refreshing operations, i.e., we avoid cryptographic computations such as group operations and NIZKs. We call such an update mechanism a "light update." Informally, we prove the following result.

Theorem 1.3 (Informal)**.** *One-way functions imply continuous non-malleable codes with deterministic light updates and without self-destruct, in the standard model, with alphabet length* $O(\log k)$*, information rate* $1 - 1/\Omega(\log k)$ *and access rate* $1 - 1/\Omega(\log k)$*, where* k *is the security parameter.*

As we have already stated, non-malleable codes against partial functions imply AONTs [41]. The first AONT was presented by Boyko [13] in the random oracle model, and then Canetti et al. [14] consider AONTs with public/private parts as well as a secret-only part, which is the full notion. Canetti et al. [14] provide efficient constructions for both settings, yet the fully secure AONT (called "secret-only" in that paper) is based on non-standard assumptions.[8]

Assuming one-way functions, our results yield efficient, fully secure AONTs, in the standard model. This resolves, the open question left in [14], where the problem of constructing AONT under standard assumptions was posed. Our result is presented in the following theorem.

Theorem 1.4 (Informal)**.** *Assuming one-way functions, there exists an explicit secret-only* AONT *in the standard model, with information rate* 1 *and access rate* $1 - 1/\Omega(\log k)$*, where* k *is the security parameter.*

The above theorem is derived by the Informal Theorem 1.1 yielding an AONT whose output consists of both the CRS and the codeword produced by the NMC scheme in the CRS model. A similar theorem can be derived with respect to the Informal Theorem 1.2. Finally, and in connection to AONTs that provide leakage resilience, our results imply leakage-resilient codes [37] for partial functions.

In the full version of the paper we provide concrete instantiations of our constructions, using textbook instantiations [35] for the underlying authenticated encryption scheme. For completeness, we also provide information theoretic variants of our constructions that maintain high access rate and thus necessarily sacrifice information rate.

1.2 Technical Overview

On the manipulation detection property. In the present work we exploit the fact that the class of partial functions does not include constant functions and we achieve a notion that is stronger than non-malleability, which we call *non-malleability* with *manipulation detection*. We formalize this notion as a strengthening of non-malleability and we show that our constructions achieve this stronger notion. Informally, manipulation detection ensures that any tampered codeword will either decode to the original message or to $\perp$.

A MD-NMC **in the CRS model.** For the exposition of our ideas, we start with a naive scheme (which does not work), and then show how we resolve all the challenges. Let $(\mathsf{KGen}, \mathsf{E}, \mathsf{D})$ be a (symmetric) authenticated encryption scheme and consider the following encoding scheme: to encode a message s, the encoder computes $(sk||e)$, where $e \leftarrow \mathsf{E}_{sk}(s)$ is the ciphertext and $sk \leftarrow \mathsf{KGen}(1^k)$, is the secret key. We observe that the scheme is secure if the tampering function can only read/write on the ciphertext, e, assuming the authenticity property

[8] In [43] the authors present a deterministic AONT construction that provides weaker security.

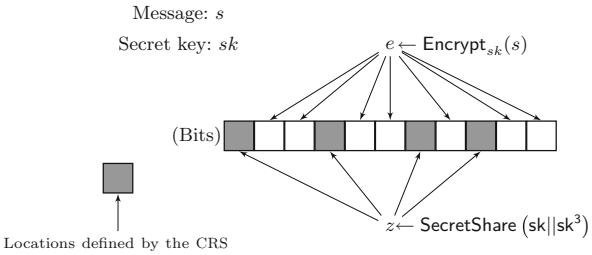

Fig. 1. Description of the scheme in the CRS model.

of the encryption scheme, however, restricting access to sk, which is short, is unnatural and makes the problem trivial. On the other hand, even partial access to sk, compromises the authenticity property of the scheme, and even if there is no explicit attack against the non-malleability property, there is no hope for proving security based on the properties of $(\mathsf{KGen}, \mathsf{E}, \mathsf{D})$, in black-box way.

A solution to the above problems would be to protect the secret key using an inner encoding, yet the amount of tampering is now restricted by the capabilities of the inner scheme, as the attacker knows the exact locations of the "*sensitive*" codeword bits, i.e., the non-ciphertext bits. In our construction, we manage to protect the secret key while avoiding the bottleneck on the access rate by designing an inner encoding scheme that provides limited security guarantees when used standalone, still when it is used in conjunction with a *shuffling technique* that permutes the inner encoding and ciphertext bit locations, it guarantees that any attack against the secret key will create an invalid encoding with overwhelming probability, even when allowing access to *almost the entire* codeword.

Our scheme is depicted in Fig. 1 and works as follows: on input message s, the encoder (i) encrypts the message by computing $sk \leftarrow \mathsf{KGen}(1^k)$ and $e \leftarrow \mathsf{E}_{sk}(s)$, (ii) computes an m-out-of-m secret sharing z of $(sk||sk^3)$ (interpreting both sk and sk^3 as elements in some finite field),[9] and outputs a random shuffling of $(z||e)$, denoted as $P_\Sigma(z||e)$, according to the common reference string Σ. Decoding proceeds as follows: on input c, the decoder (i) inverts the shuffling operation by computing $(z||e) \leftarrow P_\Sigma^{-1}(c)$, (ii) reconstructs $(sk||sk')$, and (iii) if $sk^3 = sk'$, outputs $\mathsf{D}_{sk}(e)$, otherwise, it outputs $\perp$.

In Sect. 3 we present the intuition behind our construction and a formal security analysis. Our instantiation yields a rate 1 computationally secure MD-NMC in the CRS model, with access rate $1 - 1/\Omega(\log k)$ and codewords of length $|s| + O(k^2 \log k)$, under mild assumptions (e.g., one way functions).

On the CRS. In our work, the tampering function, and consequently the codeword locations that the function is given access to, are fixed before sampling the

[9] In general, any polynomial of small degree, e.g., sk^c, would suffice, depending on the choice of the underlying finite field. Using sk^3 suffices when working over fields of characteristic 2. We could also use sk^2 over fields of characteristic 3.

CRS and this is critical for achieving security. However, proving security in this setting is non-trivial. In addition, the tampering function receives full access to the CRS when tampering with the codeword. This is in contrast to the work by Faust et al. [32] in the information-theoretic setting, where the (internal) tampering function receives partial information over the CRS.

In addition, our results tolerate adaptive selection of the codeword locations, with respect to the CRS, in the following way: each time the attacker requests access to a location, he also learns if it corresponds to a bit of z or e, together with the index of that bit in the original string. In this way, the CRS is gradually disclosed to the adversary while picking codeword locations.

Finally, our CRS sustains a substantial amount of tampering that depends on the codeword locations chosen by the attacker: an attacker that gets access to a sensitive codeword bit is allowed to modify the part of the CRS that defines the location of that bit in the codeword. The attacker is allowed to modify all but $O(k \log(|s| + k))$ bits of the CRS, that is of length $O(k^2 \log k \log(|s| + k))$. To our knowledge, this is the first construction that tolerates, even partial modification of the CRS. In contrast, existing constructions in the CRS model are either using NIZKs [22,28,30,37], or they are based on the *knowledge of exponent assumption* [36], thus tampering access to the CRS might compromise security.

Removing the CRS. A first approach would be to store the CRS inside the codeword together with $P_\Sigma(z||e)$, and give to the attacker read/write access to it. However, the tampering function, besides getting direct (partial) access to the encoding of sk, it also gets indirect access to it by (partially) controlling the CRS. Then, it can modify the CRS in way such that, during decoding, ciphertext locations of its choice will be treated as bits of the inner encoding, z, increasing the tampering rate against z significantly. This makes the task of protecting sk hard, if not impossible (unless we restrict the access rate significantly).

To handle this challenge, we embed the structure recovering mechanism inside the codeword and we emulate the CRS effect by increasing the size of the alphabet, giving rise to a block-wise structure.[10] Notice that, non-malleable codes with large alphabet size (i.e., $\text{poly}(k) + |s|$ bits) might be easy to construct, as we can embed in each codeword block the verification key of a signature scheme together with a secret share of the message, as well as a signature over the share. In this way, partial access over the codeword does not compromise the security of the signature scheme while the message remains private, and the simulation is straightforward. This approach however, comes with a large overhead, decreasing the information rate and access rate of the scheme significantly. In general, and similar to error correcting codes, we prefer smaller alphabet sizes – the larger the size is, the more coarse access structure is required, i.e., in order to access individual bits we need to access the blocks that contain them. In this work, we aim at minimizing this restriction by using small alphabets, as we describe below.

[10] Bigger alphabets have been also considered in the context of error-correcting codes, in which the codeword consists of symbols.

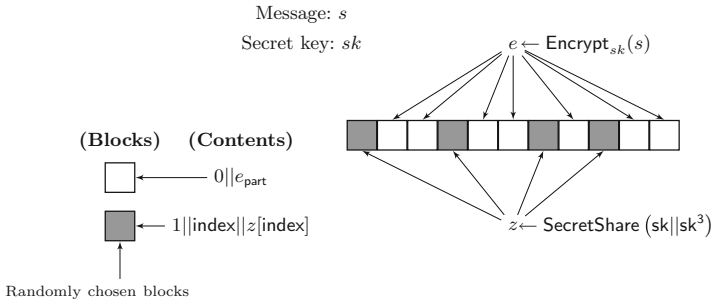

Fig. 2. Description of the scheme in the standard model.

Our approach on the problem is the following. We increase the alphabet size to $O(\log k)$ bits, and we consider two types of blocks: (i) *sensitive blocks*, in which we store the inner encoding, z, of the secret key, sk, and (ii) *non-sensitive* blocks, in which we store the ciphertext, e, that is fragmented into blocks of size $O(\log k)$. The first bit of each block indicates whether it is a sensitive block, i.e., we set it to 1 for sensitive blocks and to 0, otherwise. Our encoder works as follows: on input message s, it computes z, e, as in the previous scheme and then uses rejection sampling to sample the indices, $\rho_1, \ldots, \rho_{|z|}$, for the sensitive blocks. Then, for every $i \in \{1, \ldots, |z|\}$, ρ_i is a sensitive block, with contents $(1||i||z[i])$, while the remaining blocks keep ciphertext pieces of size $O(\log k)$. Decoding proceeds as follows: on input codeword $C = (C_1, \ldots, C_{\mathsf{bn}})$, for each $i \in [\mathsf{bn}]$, if C_i is a non-sensitive block, its data will be part of e, otherwise, the last bit of C_i will be part of z, as it is dictated by the index stored in C_i. If the number of sensitive blocks is not the expected, the decoder outputs $\perp$, otherwise, z, e, have been fully recovered and decoding proceeds as in the previous scheme. Our scheme is depicted in Fig. 2.

The security of our construction is based on the fact that, due to our shuffling technique, the position mapping will not be completely overwritten by the attacker, and as we prove in Sect. 4, this suffices for protecting the inner encoding over sk. We prove security of the current scheme (cf. Theorem 4.4) by a reduction to the security of the scheme in the CRS model. Our instantiation yields a rate $1 - 1/\Omega(\log k)$ MD-NMC in the standard model, with access rate $1-1/\Omega(\log k)$ and codewords of length $|s|(1+1/O(\log k))+O(k^2 \log^2 k)$, assuming one-way functions.

It is worth pointing out that the idea of permuting blocks containing sensitive and non-sensitive data was also considered by [42] in the context of list-decodable codes, however the similarity is only in the fact that a permutation is being used at some point in the encoding process, and our objective, construction and proof are different.

Continuously non-malleable codes with light updates. We observe that the codewords of the block-wise scheme can be updated efficiently, using shuffling and refreshing operations. Based on this observation, we prove that our code is

secure against continuous attacks, for a notion of security that is weaker than the original one [30], as we need to update our codeword. However, our update mechanism is using cheap operations, avoiding the full decoding and re-encoding of the message, which is the standard way to achieve continuous security [27,37]. In addition, our solution avoids the usage of a self-destruction mechanism that produces $\perp$ in all subsequent rounds after the first round in which the attacker creates an invalid codeword, which was originally achieved by [28], and makes an important step towards practicality.

The update mechanism works as follows: in each round, it randomly shuffles the blocks and refreshes the randomness of the inner encoding of sk. The idea here is that, due to the continual shuffling and refreshing of the inner encoding scheme, in each round the attacker learns nothing about the secret key, and every attempt to modify the inner encoding, results to an invalid key, with overwhelming probability. Our update mechanism can be made deterministic if we further encode a seed of a PRG together with the secret key, which is similar to the technique presented in [37].

Our results are presented in Sect. 5 (cf. Theorem 5.3), and the rates for the current scheme match those of the one-time secure, block-wise code.

1.3 Applications

Security against passive attackers - AONTs. Regarding the passive setting, our model and constructions find useful application in all settings where AONTs are useful (cf. [13,14,40,41]), e.g., for increasing the security of encryption without increasing the key-size, for improving the efficiency of block ciphers and constructing remotely keyed encryption [13,41], and also for constructing computationally secure secret sharing [40]. Other uses of AONTs are related to optimal asymmetric encryption padding [13].

Security against memory tampering - (Binary alphabets, Logarithmic length CRS). As with every NMC, the most notable application of the proposed model and constructions is when aiming for protecting cryptographic devices against memory tampering. Using our CRS based construction we can protect a large tamperable memory with a small (logarithmic in the message length) tamperproof memory, that holds the CRS.

The construction is as follows. Consider any device performing cryptographic operations, e.g., a smart card, whose memory is initialized when the card is being issued. Each card is initialized with an independent CRS, which is stored in a tamper-proof memory, while the codeword is stored in a tamperable memory. Due to the independency of the CRS values, it is plausible to assume that the adversary is not given access to the CRS prior to tampering with the card; the full CRS is given to the tampering function while it tampers with the codeword during computation. This idea is along the lines of the *only computation leaks information* model [38], where data can only be leaked during computation, i.e., the attacker learns the CRS when the devices performs computations that depend on it. We note that in this work we allow the tampering function to read

the full CRS, in contrast to [32], in which the tampering function receives partial information over it (our CRS can also be tampered, cf. the above discussion). In subsequent rounds the CRS and the codeword are being updated by the device, which is the standard way to achieve security in multiple rounds while using a one-time NMC [27].

Security against memory tampering - (Logarithmic length alphabets, no CRS). In modern architectures data is stored and transmitted in chunks, thus our block-wise encoding scheme can provide tamper-resilience in all these settings. For instance, consider the case of arithmetic circuits, having memory consisting of consecutive blocks storing integers. Considering adversaries that access the memory of such circuits in a block-wise manner, is a plausible scenario. In terms of modeling, this is similar to tamper-resilience for arithmetic circuits [33], in which the attacker, instead of accessing individual circuit wires carrying bits, it accesses wires carrying integers. The case is similar for RAM computation where the CPU operates over 32 or 64 bit registers (securing RAM programs using NMC was also considered by [22–24,31]). We note that the memory segments in which the codeword blocks are stored do not have to be physically separated, as partial functions output values that depend on the whole input in which they receive access to. This is in contrast to the split-state setting in which the tampering function tampers with each state independently, and thus the states need to be physically separated.

Security against adversarial channels. In Wiretap Channels [9,39,45] the goal is to communicate data privately against eavesdroppers, under the assumption that the channel between the sender and the adversary is "noisier" than the channel between the sender and the receiver. The model that we propose and our block-wise construction can be applied in this setting to provide privacy against a wiretap adversary under the assumption that due to the gap of noise there is a small (of rate $o(1)$) fraction of symbols that are delivered intact to the receiver and dropped from the transmission to the adversary. This enables private, key-less communication between the parties, guaranteeing that the receiver will either receive the original message, or $\perp$. In this way, the communication will be non-malleable in the sense that the receiver cannot be lead to output $\perp$ depending on any property of the plaintext. Our model allows the noise in the receiver side to depend on the transmission to the wiretap adversary, that tampers with a large (of rate $1-o(1)$) fraction of symbols, leading to an "active" variant of the wiretap model.

1.4 Related Work

Manipulation detection has been considered independently of the notion of non-malleability, in the seminal paper by Cramer et al. [21], who introduced the notion of *algebraic manipulation detection* (AMD) codes, providing security against additive attacks over the codeword. A similar notion was considered by Jafargholi and Wichs [34], called *tamper detection*, aiming to detect malicious modifications over the codeword, independently of how those affect the

output of the decoder. Tamper detection ensures that the application of any (admissible) function to the codeword leads to an invalid decoding.

Non-malleable codes for other function classes have been extensively studied, such as constant split-state functions [17,25], block-wise tampering [15,19], while the work of [2] develops beautiful connections among various function classes. In addition, other variants of non-malleable codes have been proposed, such as continuous non-malleable codes [30], augmented non-malleable codes [1], locally decodable/updatable non-malleable codes [16,22–24,31], and non-malleable codes with split-state refresh [28]. In [7] the authors consider AC0 circuits, bounded-depth decision trees and streaming, space-bounded adversaries. Leakage resilience was also considered as an additional feature, e.g., by [16,24,28,37].

2 Preliminaries

In this section we present basic definitions and notation that will be used throughout the paper.

Definition 2.1 (Notation)**.** *Let t, i, j, be non-negative integers. Then, $[t]$ is the set $\{1,\ldots,t\}$. For bit-strings x, y, $x||y$, is the concatenation of x, y, $|x|$ denotes the length of x, for $i \in [|x|]$, $x[i]$ is the i-th bit of x, $||_{j=1}^{t} x_j := x_1||\ldots||x_t$, and for $i \leq j$, $x[i:j] = x[i]||\ldots||x[j]$. For a set I, $|I|$, $\mathcal{P}(I)$, are the cardinality and power set of I, respectively, and for $I \subseteq [|x|]$, $x_{|_I}$ is the projection of the bits of x with respect to I. For a string variable c and value v, $c \leftarrow v$ denotes the assignment of v to c, and $c[I] \leftarrow v$, denotes an assignment such that $c_{|_I}$ equals v. For a distribution D over a set $\mathcal{X}$, $x \leftarrow D$, denotes sampling an element $x \in \mathcal{X}$, according to D, $x \leftarrow \mathcal{X}$ denotes sampling a uniform element x from $\mathcal{X}$, $U_{\mathcal{X}}$ denotes the uniform distribution over $\mathcal{X}$ and $x_1,\ldots,x_t \xleftarrow{\text{rs}} \mathcal{X}$ denotes sampling a uniform subset of $\mathcal{X}$ with t distinct elements, using rejection sampling. The statistical distance between two random variables X, Y, is denoted by $\Delta(X,Y)$, "$\approx$" and "$\approx_c$", denote statistical and computational indistinguishability, respectively, and $\mathsf{negl}(k)$ denotes an unspecified, negligible function, in k.*

Below, we define coding schemes, based on the definitions of [27,37].

Definition 2.2 (Coding scheme [27])**.** *A (κ,ν)-coding scheme, $\kappa,\nu \in \mathbb{N}$, is a pair of algorithms $(\mathsf{Enc},\mathsf{Dec})$ such that: $\mathsf{Enc} : \{0,1\}^{\kappa} \to \{0,1\}^{\nu}$ is an encoding algorithm, $\mathsf{Dec} : \{0,1\}^{\nu} \to \{0,1\}^{\kappa} \cup \{\perp\}$ is a decoding algorithm, and for every $s \in \{0,1\}^{\kappa}$, $\Pr[\mathsf{Dec}(\mathsf{Enc}(s)) = s] = 1$, where the probability runs over the randomness used by $(\mathsf{Enc},\mathsf{Dec})$.*

We can easily generalize the above definition for larger alphabets, i.e., by considering $\mathsf{Enc} : \{0,1\}^{\kappa} \to \Gamma^{\nu}$ and $\mathsf{Dec} : \Gamma^{\nu} \to \{0,1\}^{\kappa} \cup \{\perp\}$, for some alphabet Γ.

Definition 2.3 (Coding scheme in the Common Reference String (CRS) Model [37])**.** *A (κ,ν)-coding scheme in the CRS model, $\kappa,\nu \in \mathbb{N}$,*

is a triple of algorithms $(\mathsf{Init}, \mathsf{Enc}, \mathsf{Dec})$ *such that:* Init *is a randomized algorithm which receives* 1^k*, where* k *denotes the security parameter, and produces a common reference string* $\Sigma \in \{0,1\}^{\mathrm{poly}(k)}$*, and* $(\mathsf{Enc}(1^k, \Sigma, \cdot), \mathsf{Dec}(1^k, \Sigma, \cdot))$ *is a* (κ, ν)*-coding scheme,* $\kappa, \nu = \mathrm{poly}(k)$*.*

For brevity, 1^k will be omitted from the inputs of Enc and Dec.

Below we define *non-malleable codes* with *manipulation detection*, which is a stronger notion than the one presented in [27], in the sense that the tampered codeword will always decode to the original message or to $\perp$. Our definition is with respect to alphabets, as in Sect. 4 we consider alphabets of size $O(\log k)$.

Definition 2.4 (Non-Malleability with Manipulation Detection (MD-NMC))**.** *Let* Γ *be an alphabet, let* $(\mathsf{Init}, \mathsf{Enc}, \mathsf{Dec})$ *be a* (κ, ν)*-coding scheme in the common reference string model, and* $\mathcal{F}$ *be a family of functions* $f : \Gamma^\nu \to \Gamma^\nu$*. For any* $f \in \mathcal{F}$ *and* $s \in \{0,1\}^\kappa$*, define the tampering experiment*

$$\mathsf{Tamper}_s^f := \left\{ \begin{array}{c} \Sigma \leftarrow \mathsf{Init}(1^k), c \leftarrow \mathsf{Enc}(\Sigma, s), \tilde{c} \leftarrow f_\Sigma(c), \tilde{s} \leftarrow \mathsf{Dec}(\Sigma, \tilde{c}) \\ \mathit{Output} : \tilde{s}. \end{array} \right\}$$

which is a random variable over the randomness of Enc*,* Dec *and* Init*. The coding scheme* $(\mathsf{Init}, \mathsf{Enc}, \mathsf{Dec})$ *is non-malleable with manipulation detection with respect to the function family* $\mathcal{F}$*, if for all, sufficiently large* k *and for all* $f \in \mathcal{F}$*, there exists a distribution* $D_{(\Sigma,f)}$ *over* $\{0,1\}^\kappa \cup \{\perp, \mathsf{same}^*\}$*, such that for all* $s \in \{0,1\}^\kappa$*, we have:*

$$\left\{\mathsf{Tamper}_s^f\right\}_{k\in\mathbb{N}} \approx \left\{ \begin{array}{c} \tilde{s} \leftarrow D_{(\Sigma,f)} \\ \mathit{Output}\ s\ \mathit{if}\ \tilde{s} = \mathsf{same}^*,\ \mathit{and}\ \perp\ \mathit{otherwise} \end{array} \right\}_{k\in\mathbb{N}}$$

where $\Sigma \leftarrow \mathsf{Init}(1^k)$ *and* $D_{(\Sigma,f)}$ *is efficiently samplable given access to* f*,* Σ*. Here, "*$\approx$*" may refer to statistical, or computational, indistinguishability.*

In the above definition, f is parameterized by Σ to differentiate tamper-proof input, i.e., Σ, from tamperable input, i.e., c.

Below we define the tampering function class that will be used throughout the paper.

Definition 2.5 (The class of partial functions $\mathcal{F}_\Gamma^{\alpha\nu}$ (or $\mathcal{F}^\alpha$))**.** *Let* Γ *be an alphabet,* $\alpha \in [0,1)$ *and* $\nu \in \mathbb{N}$*. Any* $f \in \mathcal{F}_\Gamma^{\alpha\nu}$*,* $f : \Gamma^\nu \to \Gamma^\nu$*, is indexed by a set* $I \subseteq [\nu]$*,* $|I| \leq \alpha\nu$*, and a function* $f' : \Gamma^{\alpha\nu} \to \Gamma^{\alpha\nu}$*, such that for any* $x \in \Gamma^\nu$*,* $(f(x))_{|_I} = f'\left(x_{|_I}\right)$ *and* $(f(x))_{|_{I^c}} = x_{|_{I^c}}$*, where* $I^c := [\nu] \backslash I$*.*

For simplicity, in the rest of the text we will use the notation $f(x)$ and $f(x_{|_I})$ (instead of $f'\left(x_{|_I}\right)$). Also, the length of the codeword, ν, according to Γ, will be omitted from the notation and whenever Γ is omitted we assume that $\Gamma = \{0,1\}$. In Sect. 3, we consider $\Gamma = \{0,1\}$, while in Sect. 4, $\Gamma = \{0,1\}^{O(\log k)}$, i.e., the tampering function operates over blocks of size $O(\log k)$. When considering the CRS model, the functions are parameterized by the common reference string.

The following lemma is useful for proving security throughout the paper.

Lemma 2.6. *Let* $(\mathsf{Enc}, \mathsf{Dec})$ *be a* (κ, ν)*-coding scheme and* $\mathcal{F}$ *be a family of functions. For every* $f \in \mathcal{F}$ *and* $s \in \{0,1\}^\kappa$*, define the tampering experiment*

$$\mathsf{Tamper}_s^f := \left\{ \begin{array}{c} c \leftarrow \mathsf{Enc}(s), \tilde{c} \leftarrow f(c), \tilde{s} \leftarrow \mathsf{Dec}(\tilde{c}) \\ \textit{Output } \mathsf{same}^* \textit{ if } \tilde{s} = s, \textit{ and } \tilde{s} \textit{ otherwise.} \end{array} \right\}$$

which is a random variable over the randomness of Enc *and* Dec*.* $(\mathsf{Enc}, \mathsf{Dec})$ *is an* MD-NMC *with respect to* $\mathcal{F}$*, if for any* $f \in \mathcal{F}$ *and all sufficiently large* k*: (i) for any pair of messages* s_0*,* $s_1 \in \{0,1\}^\kappa$*,* $\left\{\mathsf{Tamper}_{s_0}^f\right\}_{k\in\mathbb{N}} \approx \left\{\mathsf{Tamper}_{s_1}^f\right\}_{k\in\mathbb{N}}$*, and (ii) for any* s*,* $\Pr\left[\mathsf{Tamper}_s^f \notin \{\bot, s\}\right] \leq \mathsf{negl}(k)$*. Here, "*$\approx$*" may refer to statistical, or computational, indistinguishability.*

The proof of the above lemma is provided in the full version of the paper. For coding schemes in the CRS model the above lemma is similar, and Tamper_s^f internally samples $\Sigma \leftarrow \mathsf{Init}(1^k)$.

3 An MD-NMC for Partial Functions, in the CRS Model

In this section we consider $\Gamma = \{0,1\}$ and we construct a rate 1 MD-NMC for $\mathcal{F}^\alpha$, with access rate $\alpha = 1 - 1/\Omega(\log k)$. Our construction is defined below and depicted in Fig. 1.

Construction 3.1. *Let* k*,* $m \in \mathbb{N}$*, let* $(\mathsf{KGen}, \mathsf{E}, \mathsf{D})$ *be a symmetric encryption scheme,* $(\mathsf{SS}_m, \mathsf{Rec}_m)$ *be an* m*-out-of-*m *secret sharing scheme, and let* $l \leftarrow 2m|sk|$*, where* sk *follows* $\mathsf{KGen}(1^k)$*. We define an encoding scheme* $(\mathsf{Init}, \mathsf{Enc}, \mathsf{Dec})$*, that outputs* $\nu = l + |e|$ *bits,* $e \leftarrow \mathsf{E}_{sk}(s)$*, as follows:*

- $\mathsf{Init}(1^k)$*: Sample* $r_1, \ldots, r_l \xleftarrow{\mathsf{rs}} \{0,1\}^{\log(\nu)}$*, and output* $\Sigma = (r_1, \ldots, r_l)$*.*
- $\mathsf{Enc}(\Sigma, \cdot)$*: for input message* s*, sample* $sk \leftarrow \mathsf{KGen}(1^k)$*,* $e \leftarrow \mathsf{E}_{sk}(s)$*.*
  - **(Secret share)** *Sample* $z \leftarrow \mathsf{SS}_m(sk||sk^3)$*, where* $z = \|_{i=1}^{2|sk|} z_i$*,* $z \in \{0,1\}^{2m|sk|}$*, and for* $i \in [|sk|]$*,* z_i *(resp.* $z_{|sk|+i}$*) is an* m*-out-of-*m *secret sharing of* $sk[i]$ *(resp.* $sk^3[i]$*).*
  - **(Shuffle)** *Compute* $c \leftarrow P_\Sigma(z||e)$ *as follows:*
    1. **(Sensitive bits)***: Set* $c \leftarrow 0^\nu$*. For* $i \in [l]$*,* $c[r_i] \leftarrow z[i]$*.*
    2. **(Ciphertext bits)***: Set* $i \leftarrow 1$*. For* $j \in [l + |e|]$*, if* $j \notin \{r_p \mid p \in [l]\}$*,* $c[j] \leftarrow e[i]$*,* i++*.*

  Output c*.*
- $\mathsf{Dec}(\Sigma, \cdot)$*: on input* c*, compute* $(z||e) \leftarrow P_\Sigma^{-1}(c)$*,* $(sk||sk') \leftarrow \mathsf{Rec}_m(z)$*, and if* $sk^3 = sk'$*, output* $\mathsf{D}_{sk}(e)$*, otherwise output* $\bot$*.*

The set of indices of z_i *in the codeword will be denoted by* Z_i*.*

In the above we consider all values as elements over $\mathbf{GF}(2^{\mathrm{poly}(k)})$.

Our construction combines authenticated encryption with an inner encoding that works as follows. It interprets sk as an element in the finite field $\mathbf{GF}(2^{|sk|})$ and computes sk^3 as a field element. Then, for each bit of $(sk||sk^3)$, it computes

an m-out-of-m secret sharing of the bit, for some parameter m (we note that elements in $\mathbf{GF}(2^{|sk|})$ can be interpreted as bit strings). Then, by combining the inner encoding with the shuffling technique, we get a encoding scheme whose security follows from the observations that we briefly present below:

- For any tampering function which does not have access to all m shares of a single bit of $(sk||sk^3)$, the tampering effect on the secret key can be expressed essentially as a linear shift, i.e., as $((sk+\delta)||(sk^3+\eta))$ for some $(\delta,\eta) \in \mathbf{GF}(2^{|sk|}) \times \mathbf{GF}(2^{|sk|})$, independent of sk.
- By permuting the locations of the inner encoding and the ciphertext bits, we have that with overwhelming probability any tampering function who reads/writes on a $(1-o(1))$ fraction of codeword bits, will not learn any single bit of $(sk||sk^3)$.
- With overwhelming probability over the randomness of sk and CRS, for non-zero η and δ, $(sk+\delta)^3 \neq sk^3+\eta$, and this property enables us to design a consistency check mechanism whose output is simulatable, without accessing sk.
- The security of the final encoding scheme follows by composing the security of the inner encoding scheme with the authenticity property of the encryption scheme.

Below we present the formal security proof of the above intuitions.

Theorem 3.2. *Let k, $m \in \mathbb{N}$ and $\alpha \in [0,1)$. Assuming $(\mathsf{SS}_m, \mathsf{Rec}_m)$ is an m-out-of-m secret sharing scheme and $(\mathsf{KGen}, \mathsf{E}, \mathsf{D})$ is* 1-IND-CPA[11] *secure, authenticated encryption scheme, the code of Construction 3.1 is a* MD-NMC *against $\mathcal{F}^{\alpha}$, for any α, m, such that $(1-\alpha)m = \omega(\log(k))$.*

Proof. Let I be the set of indices chosen by the attacker and $I^{\mathsf{c}} = [\nu] \backslash I$, where $\nu = 2m|sk| + |e|$. The tampered components of the codeword will be denoted using the character "˜" on top of the original symbol, i.e., we have $\tilde{c} \leftarrow f(c)$, the tampered secret key sk (resp. sk^3) that we get after executing $\mathsf{Rec}_m(\tilde{z})$ will be denoted by $\tilde{sk}$ (resp. $\tilde{sk}'$). Also the tampered ciphertext will be $\tilde{e}$. We prove the needed using a series of hybrid experiments that are depicted in Fig. 3. Below, we describe the hybrids.

- $\mathsf{Exp}_0^{\Sigma,f,s}$: We prove security of our code using Lemma 2.6, i.e., by showing that (i) for any s_0, s_1, $\mathsf{Tamper}_{s_0}^f \approx \mathsf{Tamper}_{s_1}^f$, and (ii) for any s, $\Pr\left[\mathsf{Tamper}_s^f \notin \{\perp, s\}\right] \leq \mathsf{negl}(k)$, where Tamper_s^f is defined in Lemma 2.6. For any f, s, $\Sigma \leftarrow \mathsf{Init}(1^k)$, the first experiment, $\mathsf{Exp}_0^{\Sigma,f,s}$, matches the experiment Tamper_s^f in the CRS model, i.e., Σ is sampled inside Tamper_s^f.

[11] This is an abbreviations for indistinguishability under chosen plaintext attack, for a single pre-challenge query to the encryption oracle.

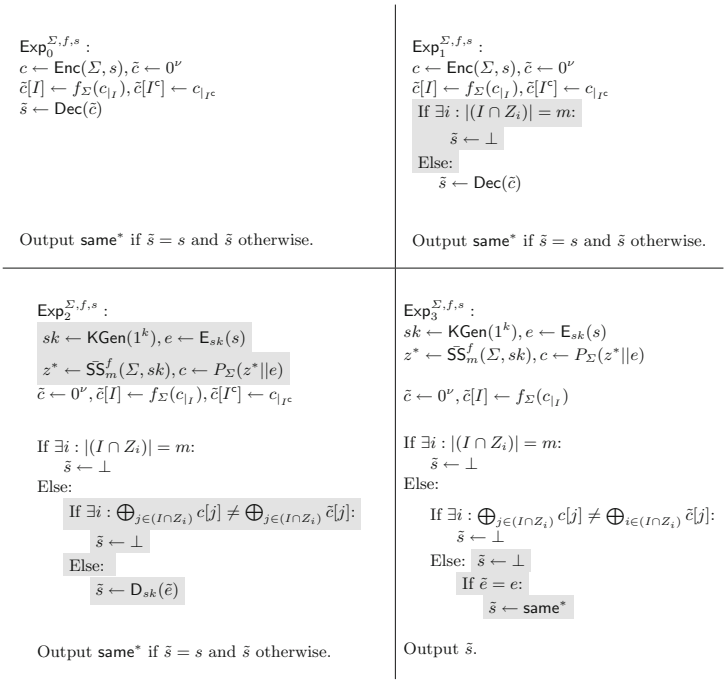

Fig. 3. The hybrid experiments for the proof of Theorem 3.2.

- $\mathsf{Exp}_1^{\Sigma,f,s}$: In the second experiment we define Z_i, $i \in [2|sk|]$, to be the set of codeword indices in which the secret sharing z_i is stored, $|Z_i| = m$. The main difference from the previous experiment is that the current one outputs $\perp$, if there exists a bit of sk or sk^3 for which the tampering function reads all the shares of it, while accessing at most $\alpha\nu$ bits of the codeword. Intuitively, and as we prove in Claim 3.3, by permuting the location indices of $z||e$, this event happens with probability negligible in k, and the attacker does not learn any bit of sk and sk^3, even if he is given access to $(1-o(1))\nu$ bits of the codeword.
- $\mathsf{Exp}_2^{\Sigma,f,s}$: By the previous hybrid we have that for all $i \in [2|sk|]$, the tampering function will not access all bits of z_i, with overwhelming probability. In the third experiment we unfold the encoding procedure, and in addition, we substitute the secret sharing procedure SS_m with $\bar{\mathsf{SS}}_m^f$ that computes shares z_i^* that reveal no information about $sk||sk^3$; for each i, $\bar{\mathsf{SS}}_m^f$ simply "drops" the bit of z_i with the largest index that is not being accessed by f. We formally define $\bar{\mathsf{SS}}_m^f$ below.

$\bar{\mathsf{SS}}_m^f(\Sigma, sk)$:

1. Sample $(z_1, \ldots, z_{2|sk|}) \leftarrow \mathsf{SS}_m\left(sk||sk^3\right)$ and set $z_i^* \leftarrow z_i$, $i \in [2|sk|]$.
2. For $i \in [2|sk|]$, let $l_i := \max_d \{d \in [m] \wedge \mathsf{Ind}(z_i[d]) \notin I)\}$, where Ind returns the index of $z_i[d]$ in c, i.e., l_i is the largest index in $[m]$ such that $z_i[l_i]$ is not accessed by f.
3. (**Output**): For all i set $z_i^*[l_i] = *$, and output $z^* := \|_{i=1}^{2|sk|} z_i^*$.

In $\mathsf{Exp}_1^{\Sigma,f,s}$, $z = \|_{i=1}^{2|sk|} z_i$, and each z_i is an m-out-of-m secret sharing for a bit of sk or sk^3. From Claim 3.3, we have that for all i, $|I \cap Z_i| < m$ with overwhelming probability, and we can observe that the current experiment is identical to the previous one up to the point of computing $f(c_{|_I})$, as $c_{|_I}$ and $f(c_{|_I})$ depend only on z^*, that carries no information about sk and sk^3.

Another difference between the two experiments is in the external "Else" branch: $\mathsf{Exp}_1^{\Sigma,f,s}$ makes a call on the decoder while $\mathsf{Exp}_2^{\Sigma,f,s}$, before calling $\mathsf{D}_{sk}(\tilde{e})$, checks if the tampering function has modified the shares in a way such that the reconstruction procedure $((\tilde{sk}, \tilde{sk}') \leftarrow \mathsf{Rec}_m(\tilde{z}))$ will give $\tilde{sk} \neq sk$ or $\tilde{sk}' \neq sk'$. This check is done by the statement "If $\exists i : \bigoplus_{j \in (I \cap Z_i)} c[j] \neq \bigoplus_{j \in (I \cap Z_i)} \tilde{c}[j]$", without touching sk or sk^3.[12] In case modification is detected the current experiments outputs $\perp$. The intuition is that an attacker that partially modifies the shares of sk and sk^3, creates shares of $\tilde{sk}$ and $\tilde{sk}'$, such that $\tilde{sk}^3 = \tilde{sk}'$, with negligible probability in k. We prove this by a reduction to the 1-IND-CPA security of the encryption scheme: any valid modification over the inner encoding of the secret key gives us method to compute the original secret key sk, with non-negligible probability. The ideas are presented formally in Claim 3.4.

– $\mathsf{Exp}_3^{\Sigma,f,s}$: The difference between the current experiment and the previous one is that instead of calling the decryption $\mathsf{D}_{sk}(\tilde{e})$, we first check if the attacker has modified the ciphertext, in which case the current experiment outputs $\perp$, otherwise it outputs same^*. By the previous hybrid, we reach this newly introduced "Else" branch of $\mathsf{Exp}_3^{\Sigma,f,s}$, only if the tampering function didn't modify the secret key. Thus, the indistinguishability between the two experiments follows from the authenticity property of the encryption scheme in the presence of z^*: given that $\tilde{sk} = sk$ and $\tilde{sk}' = sk'$, we have that if the attacker modifies the ciphertext, then with overwhelming probability $\mathsf{D}_{sk}(\tilde{e}) = \perp$, otherwise, $\mathsf{D}_{sk}(\tilde{e}) = s$, and the current experiment correctly outputs same^* or $\perp$ (cf. Claim 3.5).
– Finally, we prove that for any $f \in \mathcal{F}^\alpha$, and message s, $\mathsf{Exp}_3^{\Sigma,f,s}$ is indistinguishable from $\mathsf{Exp}_3^{\Sigma,f,0}$, where 0 denotes the zero-message. This follows by the semantic security of the encryption scheme, and gives us the indistinguishability property of Lemma 2.6. The manipulation detection property is derived by the indistinguishability between the hybrids and the fact that the output of $\mathsf{Exp}_3^{\Sigma,f,s}$ is in the set $\{\mathsf{same}^*, \perp\}$.

[12] Recall that our operations are over $\mathbf{GF}(2^{\mathrm{poly}(k)})$.

In what follows, we prove indistinguishability between the hybrids using a series of claims.

Claim 3.3. *For $k, m \in \mathbb{N}$, assume $(1-\alpha)m = \omega(\log(k))$. Then, for any $f \in \mathcal{F}^{\alpha}$ and any message s, we have $\mathsf{Exp}_0^{\Sigma,f,s} \approx \mathsf{Exp}_1^{\Sigma,f,s}$, where the probability runs over the randomness used by* Init, Enc.

Proof. The difference between the two experiments is that $\mathsf{Exp}_1^{\Sigma,f,s}$ outputs $\perp$ when the attacker learns all shares of some bit of sk or sk^3, otherwise it produces output as $\mathsf{Exp}_0^{\Sigma,f,s}$ does. Let E the event "$\exists i : |(I \cap Z_i)| = m$". Clearly, $\mathsf{Exp}_0^{\Sigma,f,s} = \mathsf{Exp}_1^{\Sigma,f,s}$ conditioned on $\neg E$, thus the statistical distance between the two experiments is bounded by $\Pr[E]$. In the following we show that $\Pr[E] \leq \mathsf{negl}(k)$. We define by E_i the event in which f learns the entire z_i. Assuming the attacker reads n bits of the codeword, we have that for all $i \in [2|sk|]$,

$$\Pr_{\Sigma}[E_i] = \Pr_{\Sigma}[\ |I \cap Z_i| = m\ ] = \prod_{j=0}^{m-1} \frac{n-j}{\nu-j} \leq \left(\frac{n}{\nu}\right)^m.$$

We have $n = \alpha\nu$ and assuming $\alpha = 1-\epsilon$ for $\epsilon \in (0,1]$, we have $\Pr[E_i] \leq (1-\epsilon)^m \leq 1/e^{m\epsilon}$ and $\Pr[E] = \Pr_{\Sigma}\left[\bigcup_{i=1}^{2|sk|} E_i\right] \leq \frac{2|sk|}{e^{m\epsilon}}$, which is negligible when $(1-\alpha)m = \omega(\log(k))$, and the proof of the claim is complete. ■

Claim 3.4. *Assuming* $(\mathsf{KGen}, \mathsf{E}, \mathsf{D})$ *is* 1-IND-CPA *secure, for any $f \in \mathcal{F}^{\alpha}$ and any message s, $\mathsf{Exp}_1^{\Sigma,f,s} \approx \mathsf{Exp}_2^{\Sigma,f,s}$, where the probability runs over the randomness used by* Init, Enc.

Proof. In $\mathsf{Exp}_2^{\Sigma,f,s}$ we unfold the encoding procedure, however instead of calling SS_m, we make a call to $\bar{\mathsf{SS}}_m^f$. As we have already stated above, this modification does not induce any difference between the output of $\mathsf{Exp}_2^{\Sigma,f,s}$ and $\mathsf{Exp}_1^{\Sigma,f,s}$, with overwhelming probability, as z^* is indistinguishable from z in the eyes of f. Another difference between the two experiments is in the external "Else" branch: $\mathsf{Exp}_1^{\Sigma,f,s}$ makes a call on the decoder while $\mathsf{Exp}_2^{\Sigma,f,s}$, before calling $\mathsf{D}_{sk}(\tilde{e})$, checks if the tampering function has modified the shares in a way such that the reconstruction procedure will give $\tilde{sk} \neq sk$ or $\tilde{sk}' \neq sk'$. This check is done by the statement "If $\exists i : \bigoplus_{j\in(I\cap Z_i)} c[j] \neq \bigoplus_{j\in(I\cap Z_i)} \tilde{c}[j]$", without touching sk or sk^3 (cf. Claim 3.3).[13] We define the events E, E' as follows

$$E : \mathsf{Dec}(\tilde{c}) \neq \perp, E' : \exists i : \bigoplus\nolimits_{j\in(I\cap Z_i)} c[j] \neq \bigoplus\nolimits_{j\in(I\cap Z_i)} \tilde{c}[j].$$

Clearly, conditioned on $\neg E'$ the two experiments are identical, since we have $\tilde{sk} = sk$ and $\tilde{sk}' = sk'$, and the decoding process will output $\mathsf{D}_{sk}(\tilde{e})$ in both experiments. Thus, the statistical distance is bounded by $\Pr[E']$. Now, conditioned on $E' \wedge \neg E$, both experiments output $\perp$. Thus, we need to bound

[13] Recall that our operations are over $\mathbf{GF}(2^{\mathrm{poly}(k)})$.

$\Pr[E \wedge E']$. Assuming $\Pr[E \wedge E'] > p$, for $p = 1/\text{poly}(k)$, we define an attacker $\mathcal{A}$ that simulates $\mathsf{Exp}_2^{\Sigma,f,s}$, and uses f, s to break the 1-IND-CPA security of $(\mathsf{KGen}, \mathsf{E}, \mathsf{D})$ in the presence of z^*, with probability at least $1/2 + p''/2$, for $p'' = 1/\text{poly}(k)$.

First we prove that any 1-IND-CPA secure encryption scheme, remains secure even if the attacker receives $z^* \leftarrow \bar{\mathsf{SS}}_m^f(\Sigma, sk)$, as z^* consists of $m-1$ shares of each bit of sk and sk^3, i.e., for the entropy of sk we have $\mathbf{H}(sk|z^*) = \mathbf{H}(sk)$. Towards contradiction, assume there exists $\mathcal{A}$ that breaks the 1-IND-CPA security of $(\mathsf{KGen}, \mathsf{E}, \mathsf{D})$ in the presence of z^*, i.e., there exist s, s_0, s_1 such that $\mathcal{A}$ distinguishes between $(z^*, \mathsf{E}_{sk}(s), \mathsf{E}_{sk}(s_0))$ and $(z^*, \mathsf{E}_{sk}(s), \mathsf{E}_{sk}(s_1))$, with non-negligible probability p. We define an attacker $\mathcal{A}'$ that breaks the 1-IND-CPA security of $(\mathsf{KGen}, \mathsf{E}, \mathsf{D})$ as follows: $\mathcal{A}'$, given $(\mathsf{E}_{sk}(s), \mathsf{E}_{sk}(s_b))$, for some $b \in \{0,1\}$, samples $\hat{sk} \leftarrow \mathsf{KGen}(1^k)$, $\hat{z}^* \leftarrow \bar{\mathsf{SS}}_m^f(\Sigma, \hat{sk})$ and outputs $b' \leftarrow \mathcal{A}(z^*, \mathsf{E}_{sk}(s), \mathsf{E}_{sk}(s_b))$. Since $(z^*, \mathsf{E}_{sk}(s), \mathsf{E}_{sk}(s_b)) \approx (\hat{z}^*, \mathsf{E}_{sk}(s), \mathsf{E}_{sk}(s_b))$ the advantage of $\mathcal{A}'$ in breaking the 1-IND-CPA security of the scheme is the advantage of $\mathcal{A}$ in breaking the 1-IND-CPA security of the scheme in the presence of z^*, which by assumption is non-negligible, and this completes the current proof. We note that the proof idea presented in the current paragraph also applies for proving that other properties that will be used in the rest of the proof, such as semantic security and authenticity, of the encryption scheme, are retained in the presence of z^*.

Now we prove our claim. Assuming $\Pr[E \wedge E'] > p$, for $p = 1/\text{poly}(k)$, we define an attacker $\mathcal{A}$ that breaks the 1-IND-CPA security of $(\mathsf{KGen}, \mathsf{E}, \mathsf{D})$ in the presence of z^*, with non-negligible probability. $\mathcal{A}$ receives the encryption of s, which corresponds to the oracle query right before receiving the challenge ciphertext, the challenge ciphertext $e \leftarrow \mathsf{E}_{sk}(s_b)$, for uniform $b \in \{0,1\}$ and uniform messages s_0, s_1, as well as z^*. $\mathcal{A}$ is defined below.

$\mathcal{A}\left(z^* \leftarrow \bar{\mathsf{SS}}_m^f(\Sigma, sk), e' \leftarrow \mathsf{E}_{sk}(s), e \leftarrow \mathsf{E}_{sk}(s_b)\right)$:

1. **(Define the shares that will be accessed by f)**: For $i \in [2|sk|]$, define $w_i := (z_i^*)_{|_{[m]\setminus\{l_i\}}}$ and for $i \in [m-1]$ define $C_i = \|_{j=1}^{|sk|} w_j[i]$, $D_i = \|_{j=|sk|+1}^{2|sk|} w_j[i]$.
2. **(Apply f)** Set $c \leftarrow P_\Sigma(z^*||e)$, compute $\tilde{c}[I] \leftarrow f_\Sigma(c_{|_I})$ and let $\tilde{C}_i$, $\tilde{D}_i$, $i \in [m]$, be the tampered shares resulting after the application of f to $c_{|_I}$.
3. **(Guessing the secret key)** Let $U = \sum_{i=1}^{m-1} C_i$, $V = \sum_{i=1}^{m-1} D_i$, i.e., U, V denote the sum of the shares that are being accessed by the attacker (maybe partially), and $\tilde{U} = \sum_{i=1}^{m-1} \tilde{C}_i$, $\tilde{V} = \sum_{i=1}^{m-1} \tilde{D}_i$, are the corresponding tampered values after applying f on U, V. Define

$$p(X) := (U - \tilde{U})X^2 + (U^2 - \tilde{U}^2)X + (U^3 - \tilde{U}^3 - V + \tilde{V}),$$

and compute the set of roots of $p(X)$, denoted as $\mathcal{X}$, which are at most two. Then set

$$\hat{\mathcal{SK}} := \{x + U | x \in \mathcal{X}\}. \tag{1}$$

4. **(Output)** Execute the following steps,
   (a) For $\hat{sk} \in \hat{\mathcal{SK}}$, compute $s' \leftarrow \mathsf{D}_{\hat{sk}}(e')$, and if $s' = s$, compute $s'' \leftarrow \mathsf{D}_{\hat{sk}}(e)$. Output b' such that $s_{b'} = s''$.
   (b) Otherwise, output $b' \leftarrow \{0,1\}$.

In the first step $\mathcal{A}$ removes the dummy symbol "$*$" and computes the shares that will be partially accessed by f, denoted as C_i for sk and as D_i for sk^3. In the second step, it defines the tampered shares, $\tilde{C}_i$, $\tilde{D}_i$. Conditioned on E', it is not hard to see that $\mathcal{A}$ simulates perfectly $\mathsf{Exp}_2^{\Sigma,f,s}$. In particular, it simulates perfectly the input to f as it receives $e \leftarrow \mathsf{E}_{sk}(s)$ and all but $2|sk|$ of the actual bit-shares of sk, sk^3. Part of those shares will be accessed by f. Since for all i, $|I \cap Z_i| < m$, the attacker is not accessing any single bit of sk, sk^3. Let C_m, D_m, be the shares (not provided by the encryption oracle) that completely define sk and sk^3, respectively. By the definition of the encoding scheme and the fact that sk, $sk^3 \in \mathbf{GF}(2^{\mathrm{poly}(k)})$, we have $\sum_{i=1}^{m} C_i = sk$, $\sum_{i=1}^{m} D_i = sk^3$, and

$$(U + C_m)^3 = V + D_m. \tag{2}$$

In order for the decoder to output a non-bottom value, the shares created by the attacker must decode to $\tilde{sk}$, $\tilde{sk}'$, such that $\tilde{sk}^3 = \tilde{sk}'$, or in other words, if

$$\left(\tilde{U} + C_m\right)^3 = \tilde{V} + D_m. \tag{3}$$

From 2 and 3 we receive

$$(U - \tilde{U})C_m^2 + (U^2 - \tilde{U}^2)C_m + (U^3 - \tilde{U}^3) = V - \tilde{V}. \tag{4}$$

Clearly, $\Pr[E \wedge E' \wedge (U = \tilde{U})] = 0$. Thus, assuming $\Pr[E \wedge E'] > p$, for $p > 1/\mathrm{poly}(k)$, we receive

$$\begin{aligned} p < \Pr\left[E \wedge E' \wedge (U \neq \tilde{U})\right] &\leq \Pr\left[\mathsf{Dec}(\tilde{c}) \neq \perp \wedge E' \wedge U \neq \tilde{U}\right] \\ &\leq \Pr\left[\tilde{sk}^3 = \tilde{sk}' \wedge E' \wedge (U \neq \tilde{U})\right] \\ &\overset{(4,1)}{=} \Pr\left[C_m \in \mathcal{X}\right] \overset{(1)}{\leq} \Pr\left[sk \in \hat{\mathcal{SK}}\right], \end{aligned} \tag{5}$$

and $\mathcal{A}$ manages to recover C_m, and thus sk, with non-negligible probability $p' \geq p$. Let W be the event of breaking 1-IND-CPA security. Then,

$$\begin{aligned} \Pr[W] &= \Pr[W | sk \in \hat{\mathcal{SK}}] \cdot \Pr[sk \in \hat{\mathcal{SK}}] \\ &+ \Pr[W | sk \notin \hat{\mathcal{SK}}] \cdot \Pr[sk \notin \hat{\mathcal{SK}}] \\ &\overset{(5)}{=} p' + \frac{1}{2}(1 - p') = \frac{1}{2} + \frac{p'}{2}, \end{aligned} \tag{6}$$

and the attacker breaks the IND-CPA security of $(\mathsf{KGen}, \mathsf{E}, \mathsf{D})$. Thus, we have $\Pr[E \wedge E'] \leq \mathsf{negl}(k)$, and both experiments output $\perp$ with overwhelming probability. ■

Claim 3.5. *Assuming the authenticity property of* $(\mathsf{KGen}, \mathsf{E}, \mathsf{D})$*, for any* $f \in \mathcal{F}^{\alpha}$ *and any message* s, $\mathsf{Exp}_2^{\Sigma,f,s} \approx \mathsf{Exp}_3^{\Sigma,f,s}$*, where the probability runs over the randomness used by* Init, KGen *and* E.

Proof. Before proving the claim, recall that the authenticity property of the encryption scheme is preserved under the presence of z^* (cf. Claim 3.4). Let E be the event $\tilde{sk} = sk \wedge \tilde{sk}' = sk^3$ and E' be the event $\tilde{e} \neq e$. Conditioned on $\neg E$, the two experiments are identical, as they both output $\perp$. Also, conditioned on $E \wedge \neg E'$, both experiments output same^*. Thus, the statistical distance between the two experiments is bounded by $\Pr[E \wedge E']$. Let B be the event $\mathsf{D}_{sk}(\tilde{e}) \neq \perp$. Conditioned on $E \wedge E' \wedge \neg B$ both experiments output $\perp$. Thus, we need to bound $\Pr[E \wedge E' \wedge B]$.

Assuming there exist s, f, for which $\Pr[E \wedge E' \wedge B] > p$, where $p = 1/\mathrm{poly}(k)$, we define an attacker $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ that simulates $\mathsf{Exp}_3^{\Sigma,f,s}$ and breaks the authenticity property of the encryption scheme in the presence of z^*, with non-negligible probability. $\mathcal{A}$ is defined as follows: sample $(s, st) \leftarrow \mathcal{A}_1(1^k)$, and then, on input (z^*, e, st), where $e \leftarrow \mathsf{E}_{sk}(s)$, $\mathcal{A}_2$, samples $\Sigma \leftarrow \mathsf{Init}(1^k)$, sets $\tilde{c} \leftarrow 0^{\nu}$, $c \leftarrow P_{\Sigma}(z^*||e)$, computes $\tilde{c}[I] \leftarrow f(c_{|_I})$, $\tilde{c}[I^{\mathsf{c}}] \leftarrow c_{|_{I^{\mathsf{c}}}}$, $(\tilde{z}^*||\tilde{e}) \leftarrow P_{\Sigma}^{-1}(\tilde{c})$, and outputs $\tilde{e}$. Assuming $\Pr[E \wedge E' \wedge B] > p$, we have that $\mathsf{D}_{sk}(\tilde{e}) \neq \perp$ and $\tilde{e} \neq e$, with non-negligible probability and the authenticity property of $(\mathsf{KGen}, \mathsf{E}, \mathsf{D})$ breaks. ∎

Claim 3.6. *Assuming* $(\mathsf{KGen}, \mathsf{E}, \mathsf{D})$ *is semantically secure, for any* $f \in \mathcal{F}^{\alpha}$ *and any message* s, $\mathsf{Exp}_3^{\Sigma,f,s} \approx \mathsf{Exp}_3^{\Sigma,f,0}$*, where the probability runs over the randomness used by* Init, KGen, E. *"*$\approx$*" may refer to statistical or computational indistinguishability, and* 0 *is the zero-message.*

Proof. Recall that $(\mathsf{KGen}, \mathsf{E}, \mathsf{D})$ is semantically secure even in the presence of $z^* \leftarrow \bar{\mathsf{SS}}_m^f(\Sigma, sk)$ (cf. Claim 3.4), and towards contradiction, assume there exist $f \in \mathcal{F}^{\alpha}$, message s, and PPT distinguisher D such that

$$\left| \Pr\left[\mathsf{D}\left(\Sigma, \mathsf{Exp}_3^{\Sigma,f,s}\right) = 1\right] - \Pr\left[\mathsf{D}\left(\Sigma, \mathsf{Exp}_3^{\Sigma,f,0}\right)\right] = 1 \right| > p,$$

for $p = 1/\mathrm{poly}(k)$. We are going to define an attacker $\mathcal{A}$ that breaks the semantic security of $(\mathsf{KGen}, \mathsf{E}, \mathsf{D})$ in the presence of z^*, using $s_0 := s$, $s_1 := 0$. $\mathcal{A}$, given z^*, e, executes $\mathsf{Program}$.

$\mathsf{Program}(z^*, e)$:
$c \leftarrow P_{\Sigma}(z^*||e), \tilde{c} \leftarrow 0^{\nu}, \tilde{c}[I] \leftarrow f(c_{|_I})$
If $\exists i : |(I \cap Z_i)| = m$: $\tilde{s} \leftarrow \perp$
Else:
  If $\exists i : \bigoplus_{j \in (I \cap Z_i)} c[j] \neq \bigoplus_{j \in (I \cap Z_i)} \tilde{c}[j]$: $\tilde{s} \leftarrow \perp$
  Else: $\tilde{s} \leftarrow \perp$and If $\tilde{e} = e$: $\tilde{s} \leftarrow \mathsf{same}^*$
Output $\tilde{s}$.

It is not hard to see that $\mathcal{A}$ simulates $\mathsf{Exp}_3^{\Sigma,f,s_b}$, thus the advantage of $\mathcal{A}$ against the semantic security of $(\mathsf{KGen}, \mathsf{E}, \mathsf{D})$ is the same with the advantage of D in distinguishing between $\mathsf{Exp}_3^{\Sigma,f,s_0}$, $\mathsf{Exp}_3^{\Sigma,f,s_1}$, which by assumption is non-negligible. We have reached a contradiction and the proof of the claim is complete. ∎

From the above claims we have that for any $f \in \mathcal{F}^{\alpha}$ and any s, $\mathsf{Exp}_0^{\Sigma,f,s} \approx \mathsf{Exp}_3^{\Sigma,f,0}$, thus for any $f \in \mathcal{F}^{\alpha}$ and any s_0, s_1, $\mathsf{Exp}_0^{\Sigma,f,s_0} \approx \mathsf{Exp}_0^{\Sigma,f,s_1}$. Also, by the indistinguishability between $\mathsf{Exp}_0^{\Sigma,f,s}$ and $\mathsf{Exp}_3^{\Sigma,f,0}$, the second property of Lemma 2.6 has been proven as the output of $\mathsf{Exp}_3^{\Sigma,f,0}$ is in $\{s, \perp\}$, with overwhelming probability, and non-malleability with manipulation detection of our code follows by Lemma 2.6, since $\mathsf{Exp}_0^{\Sigma,f,s}$ is identical to Tamper_s^f of Lemma 2.6. ■

4 Removing the CRS

In this section we increase the alphabet size to $O(\log(k))$ and we provide a computationally secure, rate 1 encoding scheme in the standard model, tolerating modification of $(1 - o(1))\nu$ blocks, where ν is the total number of blocks in the codeword. Our construction is defined below and the intuition behind it has already been presented in the Introduction (cf. Sect. 1, Fig. 2). In the following, the projection operation will be also used with respect to bigger alphabets, enabling the projection of blocks.

Construction 4.1. *Let k, $m \in \mathbb{N}$, let $(\mathsf{KGen}, \mathsf{E}, \mathsf{D})$ be a symmetric encryption scheme and $(\mathsf{SS}_m, \mathsf{Rec}_m)$ be an m-out-of-m secret sharing scheme. We define an encoding scheme $(\mathsf{Enc}^*, \mathsf{Dec}^*)$, as follows:*

- $\mathsf{Enc}^*(1^k, \cdot)$*: for input message s, sample $sk \leftarrow \mathsf{KGen}\left(1^k\right)$, $e \leftarrow \mathsf{E}_{sk}(s)$.*
  - **(Secret share)** *Sample $z \leftarrow \mathsf{SS}_m(sk||sk^3)$, where $z = \|_{i=1}^{2|sk|} z_i$, $z \in \{0,1\}^{2m|sk|}$, and for $i \in [|sk|]$, z_i (resp. $z_{|sk|+i}$) is an m-out-of-m secret sharing of $sk[i]$ (resp. $sk^3[i]$).*
  - **(Construct blocks & permute)** *Set $l \leftarrow 2m|sk|$, $\mathsf{bs} \leftarrow \log l + 2$, $d \leftarrow |e|/\mathsf{bs}$, $\mathsf{bn} \leftarrow l + d$, sample $\rho := (\rho_1, \ldots, \rho_l) \xleftarrow{\text{rs}} \{0,1\}^{\log(\mathsf{bn})}$ and compute $C \leftarrow \Pi_\rho(z||e)$ as follows:*
    1. *Set $t \leftarrow 1$, $C_i \leftarrow 0^{\mathsf{bs}}$, $i \in [\mathsf{bn}]$.*
    2. **(Sensitive blocks)** *For $i \in [l]$, set $C_{r_i} \leftarrow (1||i||z[i])$.*
    3. **(Ciphertext blocks)** *For $i \in [\mathsf{bn}]$, if $i \neq r_j$, $j \in [l]$, $C_i \leftarrow (0||e[t : t + (\mathsf{bs} - 1)])$, $t \leftarrow t + (\mathsf{bs} - 1)$.*[14]

  Output $C := (C_1||\ldots||C_{\mathsf{bn}})$.
- $\mathsf{Dec}^*(1^k, \cdot)$*: on input C, parse it as $(C_1||\ldots||C_{\mathsf{bn}})$, set $t \leftarrow 1$, $l \leftarrow 2m|sk|$, $z \leftarrow 0^l$, $e \leftarrow 0$, $\mathcal{L} = \emptyset$ and compute $(z||e) \leftarrow \Pi^{-1}(C)$ as follows:*
  - *For $i \in [\mathsf{bn}]$,*
    - **(Sensitive block)** *If $C_i[1] = 1$, set $j \leftarrow C_i[2 : \mathsf{bs} - 1]$, $z[j] \leftarrow C_i[\mathsf{bs}]$, $\mathcal{L} \leftarrow \mathcal{L} \cup \{j\}$.*
    - **(Ciphertext block)** *Otherwise, set $e[t : t + \mathsf{bs} - 1] = C_i[2 : \mathsf{bs}]$, $t \leftarrow t + \mathsf{bs} - 1$.*
  - *If $|\mathcal{L}| \neq l$, output $\perp$, otherwise output $(z||e)$.*

[14] Here we assume that $\mathsf{bs} - 1$, divides the length of the ciphertext e. We can always achieve this property by padding the message s with zeros, if necessary.

If $\Pi^{-1}(C) = \perp$*, output* $\perp$*, otherwise, compute* $(sk||sk') \leftarrow \mathsf{Rec}_m(z)$*, and if* $sk^3 = sk'$*, output* $\mathsf{D}_{sk}(e)$*, otherwise output* $\perp$*.*

The set of indices of the blocks in which z_i *is stored will be denoted by* Z_i*.*

We prove security for the above construction by a reduction to the security of Construction 3.1. We note that that our reduction is non-black box with respect to the coding scheme in which security is reduced to; a generic reduction, i.e., non-malleable reduction [2], from the standard model to the CRS model is an interesting open problem and thus out of the scope of this work.

In the following, we consider $\Gamma = \{0,1\}^{O(\log(k))}$. The straightforward way to prove that $(\mathsf{Enc}^*, \mathsf{Dec}^*)$ is secure against $\mathcal{F}^{\alpha}_{\Gamma}$ by a reduction to the security of the bit-wise code of Sect. 3, would be as follows: for any $\alpha \in \{0,1\}$, $f \in \mathcal{F}^{\alpha}_{\Gamma}$ and any message s, we have to define α', $g \in \mathcal{F}^{\alpha'}$, such that the output of the tampered execution with respect to $(\mathsf{Enc}^*, \mathsf{Dec}^*)$, f, s, is indistinguishable from the tampered execution with respect to $(\mathsf{Init}, \mathsf{Enc}, \mathsf{Dec})$, g, s, and g is an admissible function for $(\mathsf{Init}, \mathsf{Enc}, \mathsf{Dec})$. However, this approach might be tricky as it requires the establishment of a relation between α and α' such that the sensitive blocks that f will receive access to, will be simulated using the sensitive bits accessed by g. Our approach is cleaner: for the needs of the current proof we leverage the power of Construction 3.1, by allowing the attacker to choose adaptively the codeword locations, as long as it does not request to read all shares of the secret key. Then, for every block that is accessed by the block-wise attacker f, the bit-wise attacker g requests access to the locations of the bit-wise code that enable him to fully simulate the input to g. We formally present our ideas in the following sections. In Sect. 4.1 we introduce the function class $\mathcal{F}_{\mathsf{ad}}$ that considers adaptive adversaries with respect to the CRS and we prove security of Construction 3.1 in Corollary 4.3 against a subclass of $\mathcal{F}_{\mathsf{ad}}$, and then, we reduce the security of the block-wise code $(\mathsf{Enc}^*, \mathsf{Dec}^*)$ against $\mathcal{F}^{\alpha}_{\Gamma}$ to the security of Construction 3.1 against $\mathcal{F}_{\mathsf{ad}}$ (cf. Sect. 4.2).

4.1 Security Against Adaptive Adversaries

In the current section we prove that Construction 3.1 is secure against the class of functions that request access to the codeword adaptively, i.e., depending on the CRS, as long as they access a bounded number of sensitive bits. Below, we formally define the function class $\mathcal{F}_{\mathsf{ad}}$, in which the tampering function picks up the codeword locations depending on the CRS, and we consider $\Gamma = \{0,1\}$.

Definition 4.2 (The function class $\mathcal{F}^{\nu}_{\mathsf{ad}}$)**.** *Let* $(\mathsf{Init}, \mathsf{Enc}, \mathsf{Dec})$ *be an* (κ, ν)*-coding scheme and let* $\mathbb{\Sigma}$ *be the range of* $\mathsf{Init}(1^k)$*. For any* $g = (g_1, g_2) \in \mathcal{F}^{\nu}_{\mathsf{ad}}$*, we have* $g_1 : \mathbb{\Sigma} \rightarrow \mathcal{P}([\nu])$*,* $g_2^{\Sigma} : \{0,1\}^{|\mathsf{range}(g_1)|} \rightarrow \{0,1\}^{|\mathsf{range}(g_1)|} \cup \{\perp\}$*, and for any* $c \in \{0,1\}^{\nu}$*,* $g^{\Sigma}(c) = g_2\left(c_{|_{g_1(\Sigma)}}\right)$*. For brevity, the function class will be denoted as* $\mathcal{F}_{\mathsf{ad}}$*.*

Construction 3.1 remains secure against functions that receive full access to the ciphertext, as well as they request to read all but one shares for each bit of

sk and sk^3. The result is formally presented in the following corollary and its proof, which is along the lines of the proof of Theorem 3.2, is given in the full version of the paper.

Corollary 4.3. *Let k, $m \in \mathbb{N}$. Assuming $(\mathsf{SS}_m, \mathsf{Rec}_m)$ is an m-out-of-m secret sharing scheme and $(\mathsf{KGen}, \mathsf{E}, \mathsf{D})$ is* 1-IND-CPA *secure authenticated encryption scheme, the code of Construction 4.1 is a* MD-NMC *against any $g = (g_1, g_2) \in \mathcal{F}_{\mathsf{ad}}$, assuming that for all $i \in [2|sk|]$, $(Z_i \cap g_1(\Sigma)) < m$, where $sk \leftarrow \mathsf{KGen}(1^k)$ and $\Sigma \leftarrow \mathsf{Init}(1^k)$.*

4.2 MD-NM Security of the Block-Wise Code

In the current section we prove security of Construction 4.1 against $\mathcal{F}^{\alpha}_{\Gamma}$, for $\Gamma = \{0,1\}^{O(\log(k))}$.

Theorem 4.4. *Let k, $m \in \mathbb{N}$, $\Gamma = \{0,1\}^{O(\log(k))}$ and $\alpha \in [0,1)$. Assuming $(\mathsf{SS}_m, \mathsf{Rec}_m)$ is an m-out-of-m secret sharing scheme and $(\mathsf{KGen}, \mathsf{E}, \mathsf{D})$ is a* 1-IND-CPA *secure authenticated encryption scheme, the code of Construction 4.1 is a* MD-NMC *against $\mathcal{F}^{\alpha}_{\Gamma}$, for any α, m, such that $(1-\alpha)m = \omega(\log(k))$.*

$g_1(\Sigma = (r_1, \ldots, r_l))$:

- **(Simulate block shuffling)**:
  Sample $\rho := (\rho_1, \ldots, \rho_l) \xleftarrow{\text{rs}} \{0,1\}^{\log(\mathsf{bn})}$
- **(Construct I)**: Set $I = \emptyset$,
  - **(Add ciphertext locations to I)**:
    For $j \in [|e| + l]$, if $j \notin \{r_i | i \in [l]\}$, $I \leftarrow (I \cup j)$.
  - **(Add sensitive bit locations to I according to I_{b})**:
    For $j \in [\mathsf{bn}]$, if $j \in I_{\mathsf{b}}$ and $\exists i \in [l]$ such that $j = \rho_i$, $I \leftarrow (I \cup r_i)$.
- **Output**: Output I.

Fig. 4. The function g_1 that appears in the hybrid experiments of Fig. 7.

$g_2^{\Sigma}(c_{|I})$:
$t \leftarrow 1$, $C^*_i \leftarrow 0^{\mathsf{bs}}$, $i \in [\mathsf{bn}]$.

- **(Reconstruct I)**: Compute $I \leftarrow g_1(\Sigma)$.
- **(Simulate ciphertext blocks)**:
  For $i \in [\mathsf{bn}]$, if $i \neq \rho_j$, $j \in [l]$, $C^*_i \leftarrow (0||e[t : t + (\mathsf{bs} - 1)])$, $t \leftarrow t + (\mathsf{bs} - 1)$.
- **(Simulate sensitive blocks)**:
  - For $i \in [|I|]$, if $\exists j \in [l]$, such that $\mathsf{Ind}(c_{|I}[i]) = r_j$, set $C^*_{\rho_j} \leftarrow \left(1||j||c_{|I}[i]\right)$.
  - Set $C^* := (C^*_1 || \ldots || C^*_{\mathsf{bn}})$ and $\tilde{C}^* := C^*$.
- **(Apply f)**: compute $\tilde{C}^*[I_{\mathsf{b}}] \leftarrow f(C^*_{|I_{\mathsf{b}}})$.
- **(Output)**: Output $\tilde{C}^*_{|I_{\mathsf{b}}}$.

Fig. 5. The function g_2 that appears in the hybrid experiments of Fig. 7.

Proof. Following Lemma 2.6, we prove that for any $f \in \mathcal{F}_{\Gamma}^{\alpha}$, and any pair of messages s_0, s_1, $\mathsf{Tamper}_{s_0}^{f} \approx \mathsf{Tamper}_{s_1}^{f}$, and for any s, $\Pr\left[\mathsf{Tamper}_{s}^{f} \notin \{\perp, s\}\right] \leq \mathsf{negl}(k)$, where Tamper denotes the experiment defined in Lemma 2.6 with respect to the encoding scheme of Construction 4.1, $(\mathsf{Enc}^*, \mathsf{Dec}^*)$. Our proof is given by a series of hybrids depicted in Fig. 7. We reduce the security $(\mathsf{Enc}^*, \mathsf{Dec}^*)$, to the security of Construction 3.1, $(\mathsf{Init}, \mathsf{Enc}, \mathsf{Dec})$, against $\mathcal{F}_{\mathsf{ad}}$ (cf. Corollary 4.3). The idea is to move from the tampered execution with respect to $(\mathsf{Enc}^*, \mathsf{Dec}^*)$, f, to a tampered execution with respect to $(\mathsf{Init}, \mathsf{Enc}, \mathsf{Dec})$, g, such that the two executions are indistinguishable and $(\mathsf{Init}, \mathsf{Enc}, \mathsf{Dec})$ is secure against g.

Let I_{b} be the set of indices of the blocks that f chooses to tamper with, where $|I_{\mathsf{b}}| \leq \alpha\nu$, and let $l \leftarrow 2m|sk|$, $\mathsf{bs} \leftarrow \log l + 2$, $\mathsf{bn} \leftarrow l + |e|/\mathsf{bs}$. Below we describe the hybrids of Fig. 7.

- $\mathsf{Exp}_0^{f,s}$: The current experiment is the experiment Tamper_{s}^{f}, of Lemma 2.6, with respect to $(\mathsf{Enc}^*, \mathsf{Dec}^*)$, f, s.
- $\mathsf{Exp}_1^{(g_1,g_2),s}$: The main difference between $\mathsf{Exp}_0^{f,s}$ and $\mathsf{Exp}_1^{(g_1,g_2),s}$, is that in the latter one, we introduce the tampering function (g_1, g_2), that operates over codewords of $(\mathsf{Init}, \mathsf{Enc}, \mathsf{Dec})$ and we modify the encoding steps so that the experiment creates codewords of the bit-wise code $(\mathsf{Init}, \mathsf{Enc}, \mathsf{Dec})$. (g_1, g_2) simulates partially the block-wise codeword C, while given partial access to the bit-wise codeword $c \leftarrow \mathsf{Enc}(s)$. As we prove in the full version, it simulates perfectly the tampering effect of f against $C \leftarrow \mathsf{Enc}^*(s)$.
  g_1 operates as follows (cf. Fig. 4): it simulates perfectly the randomness for the permutation of the block-wise code, denoted as ρ, and constructs a set of indices I, such that g_2 will receive access to, and tamper with, $c_{|_I}$. The set I is constructed with respect to the set of blocks I_{b}, that f chooses to read, as well as Σ, that reveals the original bit positions, i.e., the ones before permuting $(z||e)$. g_2 receives $c_{|_I}$, reconstructs I, simulates partially the blocks of the block-wise codeword, C, and applies f on the simulated codeword. The code of g_2 is given in Fig. 5. In the full version we show that g_2, given $c_{|_I}$, simulates perfectly $C_{|_{I_{\mathsf{b}}}}$, which implies that $g_2^{\Sigma}(c_{|_I}) = f(C_{|_{I_{\mathsf{b}}}})$, and the two executions are identical.
- $\mathsf{Exp}_2^{(g_1,g_3),s}$: In the current experiment, we substitute the function g_2 with g_3, and Dec^* with Dec, respectively. By inspecting the code of g_2 and g_3 (cf. Figs. 5 and 6, respectively), we observe that latter function executes the code of the former, plus the "Check labels and simulate $\tilde{c}[I]$" step. Thus the two experiments are identical up to the point of computing $f(C_{|_{I_{\mathsf{b}}}})$. The main idea here is that we want the current execution to be with respect to $(\mathsf{Init}, \mathsf{Enc}, \mathsf{Dec})$ against (g_1, g_3). Thus, we substitute Dec^* with Dec, and we expand the function g_2 with some extra instructions/checks that are missing from Dec. We name the resulting function as g_3 and we prove that the two executions are identical.
- Finally, we prove that for any f and any s, $\mathsf{Exp}_2^{(g_1,g_3),s} \approx \mathsf{Exp}_2^{(g_1,g_3),0}$ and $\Pr\left[\mathsf{Exp}_2^{(g_1,g_3),s} \notin \{\perp, s\}\right] \leq \mathsf{negl}(k)$. We do so by proving that (g_1, g_3) is

admissible for $(\mathsf{Init}, \mathsf{Enc}, \mathsf{Dec},)$, i.e., $(g_1, g_3) \in \mathcal{F}_{\mathsf{ad}}$, and g_3 will not request to access more that $m-1$ shares for each bit of sk, sk^3 (cf. Corollary 4.3). This implies security according to Lemma 2.6.

$g_3^{\Sigma}(c_{|_I})$:

$t \leftarrow 1$, $C_i^* \leftarrow 0^{\mathsf{bs}}$, $i \in [\mathsf{bn}]$.

- **(Reconstruct I)**: Compute $I \leftarrow g_1(\Sigma)$.
- **(Simulate ciphertext blocks)**: For $i \in [\mathsf{bn}]$, if $i \neq \rho_j$, $j \in [l]$, $C_i^* \leftarrow (0||e[t : t + (\mathsf{bs} - 1)])$, $t \leftarrow t + (\mathsf{bs} - 1)$.
- **(Simulate sensitive blocks)**:
  - For $i \in [|I|]$, if $\exists j \in [l]$, such that $\mathsf{Ind}(c_{|_I}[i]) = r_j$, set $C^*_{\rho_j} \leftarrow \left(1||j||c_{|_I}[i]\right)$.
  - Set $C^* := (C_1^*|| \ldots ||C^*_{\mathsf{bn}})$ and $\tilde{C}^* := C^*$.
- **(Apply f)**: compute $\tilde{C}^*[I_{\mathsf{b}}] \leftarrow f(C^*_{|_{I_{\mathsf{b}}}})$.
- **(Check labels and simulate $\tilde{c}[I]$)**: If $\Pi^{-1}(\tilde{C}^*) = \perp$, set $d \leftarrow 1$, otherwise set $(\tilde{z}^*||\tilde{e}) \leftarrow \Pi^{-1}(\tilde{C}^*)$, $\tilde{c}^* \leftarrow P_{\Sigma}(\tilde{z}^*||\tilde{e})$.
- **(Output)**: If $d = 1$ output $\perp$, otherwise output $\tilde{c}^*_{|_I}$.

Fig. 6. The function g_3 that appears in the hybrid experiments of Fig. 7.

$\mathsf{Exp}_0^{f,s}$:
$sk \leftarrow \mathsf{KGen}\left(1^k\right), e \leftarrow \mathsf{E}_{sk}(s)$
$z \leftarrow \mathsf{SS}_m(sk||sk^3)$

$\rho := (\rho_1, \ldots, \rho_l) \xleftarrow{\text{rs}} \{0,1\}^{\log(\mathsf{bn})}$
$C \leftarrow \Pi_{\rho}(z||e), \tilde{C} \leftarrow C$
$\tilde{C}[I_{\mathsf{b}}] \leftarrow f(C_{|_{I_{\mathsf{b}}}})$

$\tilde{s} \leftarrow \mathsf{Dec}^*(\tilde{C})$

Output same^* if $\tilde{s} = s$ and $\tilde{s}$ otherwise.

$\mathsf{Exp}_1^{(g_1,g_2),s}$:
$sk \leftarrow \mathsf{KGen}\left(1^k\right), e \leftarrow \mathsf{E}_{sk}(s)$
$z \leftarrow \mathsf{SS}_m(sk||sk^3)$

$\Sigma \leftarrow \mathsf{Init}^*(1^k), c \leftarrow P_{\Sigma}(z||e)$
$I \leftarrow g_1(\Sigma)$
$C \leftarrow \Pi_{\rho}(z||e), \tilde{C} \leftarrow C$
$\tilde{C}[I_{\mathsf{b}}] \leftarrow g_2^{\Sigma}(c_{|_I})$

$\tilde{s} \leftarrow \mathsf{Dec}^*(\tilde{C})$

Output same^* if $\tilde{s} = s$ and $\tilde{s}$ otherwise.

$\mathsf{Exp}_2^{(g_1,g_3),s}$:
$\Sigma \leftarrow \mathsf{Init}^*(1^k)$
$sk \leftarrow \mathsf{KGen}\left(1^k\right), e \leftarrow \mathsf{E}_{sk}(s)$
$z \leftarrow \mathsf{SS}_m(sk||sk^3)$

$c \leftarrow P_{\Sigma}(z||e), \tilde{c} \leftarrow c$
$I \leftarrow g_1(\Sigma)$
$\tilde{c}[I] \leftarrow g_3^{\Sigma}(c_{|_I})$

$\tilde{s} \leftarrow \mathsf{Dec}(\Sigma, \tilde{c})$

Output same^* if $\tilde{s} = s$ and $\tilde{s}$ otherwise.

Fig. 7. The hybrid experiments for the proof of Theorem 4.4.

The indistinguishability between the hybrids is given in the full version of the paper. ■

5 Continuous MD-NMC with Light Updates

In this section we enhance the block-wise scheme of Sect. 4 with an update mechanism, that uses only shuffling and refreshing operations. The resulting code is secure against continuous attacks, for a notion of security that is weaker than the original one [30], as we need to update our codeword. Below we define the update mechanism, which is denoted as Update^*.

Construction 5.1. *Let $k, m \in \mathbb{N}$, $(\mathsf{KGen}, \mathsf{E}, \mathsf{D})$, $(\mathsf{SS}_m, \mathsf{Rec}_m)$ be as in Construction 4.1. We define the update procedure, Update^*, for the encoding scheme of Construction 4.1, as follows:*

– $\mathsf{Update}^*(1^k, \cdot)$*: on input* C*, parse it as* $(C_1 || \ldots || C_{\mathsf{bn}})$*, set* $l \leftarrow 2m|sk|$*,* $\hat{\mathcal{L}} = \emptyset$*, and set* $\hat{C} := (\hat{C}_1 || \ldots || \hat{C}_{\mathsf{bn}})$ *to 0.*
  - **(Secret share** $0^{2|sk|}$**)***: Sample* $z \leftarrow \mathsf{SS}_m\left(0^{2|sk|}\right)$*, where* $z = ||_{i=1}^{2|sk|} z_i$*,* $z \in \{0,1\}^{2m|sk|}$*, and for* $i \in [2|sk|]$*,* z_i *is an* m*-out-of-*m *secret sharing of the* 0 *bit.*
  - **(Shuffle & Refresh)***: Sample* $\rho := (\rho_1, \ldots, \rho_l) \xleftarrow{\mathsf{rs}} \{0,1\}^{\log(\mathsf{bn})}$*. For* $i \in [\mathsf{bn}]$*,*
    * **(Sensitive block)** *If* $C_i[1] = 1$*,*
      · **(Shuffle)***: Set* $j \leftarrow C_i[2 : \mathsf{bs} - 1]$*,* $\hat{C}_{\rho_j} \leftarrow C_i$*.*
      · **(Refresh)***: Set* $\hat{C}_{\rho_j}[\mathsf{bs}] \leftarrow \hat{C}_{\rho_j}[\mathsf{bs}] \oplus z[j]$*.*
    * **(Ciphertext block)**
      If $C_i[1] = 0$*, set* $j \leftarrow \min_n \left\{ n \in [\mathsf{bn}] \middle| n \notin \hat{\mathcal{L}}, n \neq \rho_i, i \in [l] \right\}$*, and* $\hat{C}_j \leftarrow C_i$*,* $\hat{\mathcal{L}} \leftarrow \hat{\mathcal{L}} \cup \{j\}$*.*

Output $\hat{C}$*.*

The following definition of security is along the lines of the one given in [30], adapted to the notion of non-malleability with manipulation detection. Also, after each invocation the codewords are updated, where in our case the update mechanism is only using shuffling and refreshing operations. In addition, there is no need for self-destruct after detecting an invalid codeword [28].

Definition 5.2 (Continuously MD-NMC with light updates)**.** *Let* $\mathsf{CS} = (\mathsf{Enc}, \mathsf{Dec})$ *be an encoding scheme,* $\mathcal{F}$ *be a functions class and* $k, q \in \mathbb{N}$*. Then,* CS *is a* q*-continuously non-malleable (*q-CNM*) code, if for every, sufficiently large* $k \in \mathbb{N}$*, any pair of messages* s_0*,* $s_1 \in \{0,1\}^{\mathrm{poly}(k)}$*, and any* PPT *algorithm* $\mathcal{A}$*,* $\left\{\mathsf{Tamper}^{\mathcal{A}}_{s_0}(k)\right\}_{k\in\mathbb{N}} \approx \left\{\mathsf{Tamper}^{\mathcal{A}}_{s_1}(k)\right\}_{k\in\mathbb{N}}$*, where,*

$\mathsf{Tamper}^{\mathcal{A}}_s(k)$:
$C \leftarrow \mathsf{Enc}(s), \tilde{s} \leftarrow 0$
For $\tau \in [q]$:
  $f \leftarrow \mathcal{A}(\tilde{s}), \tilde{C} \leftarrow f(C), \tilde{s} \leftarrow \mathsf{Dec}(\tilde{C})$
  If $\tilde{s} = s$: $\tilde{s} \leftarrow \mathsf{same}^*$
  $C \leftarrow \mathsf{Update}^*(1^k, C)$
out $\leftarrow \mathcal{A}(\tilde{s})$
Return : *out*

and for each round the output of the decoder is not in $\{s, \perp\}$ *with negligible probability in* k*, over the randomness of* $\mathsf{Tamper}^{\mathcal{A}}_s$*.*

In the full version of the paper we prove the following statement.

Theorem 5.3. *Let* q*,* k*,* m*,* $\in \mathbb{N}$*,* $\Gamma = \{0,1\}^{O(\log(k))}$ *and* $\alpha \in [0,1)$*. Assuming* $(\mathsf{SS}_m, \mathsf{Rec}_m)$ *is an* m*-out-of-*m *secret sharing scheme and* $(\mathsf{KGen}, \mathsf{E}, \mathsf{D})$ *is a* 1-IND-CPA*, authenticated encryption scheme, the scheme of Construction 5.1 is a continuously* MD-NMC *with light updates, against* $\mathcal{F}^{\alpha}_{\Gamma}$*, for any* α*,* m*, such that* $(1-\alpha)m = \omega(\log(k))$*.*

In the above theorem, q can be polynomial (resp. exponential) in k, assuming the underlying encryption scheme is computationally (resp. unconditionally) secure.

References

1. Aggarwal, D., Agrawal, S., Gupta, D., Maji, H.K., Pandey, O., Prabhakaran, M.: Optimal computational split-state non-malleable codes. In: Kushilevitz, E., Malkin, T. (eds.) TCC 2016. LNCS, vol. 9563, pp. 393–417. Springer, Heidelberg (2016). https://doi.org/10.1007/978-3-662-49099-0_15
2. Aggarwal, D., Dodis, Y., Kazana, T., Obremski, M.: Non-malleable reductions and applications. In: STOC, pp. 459–468 (2015)
3. Aggarwal, D., Dodis, Y., Lovett, S.: Non-malleable codes from additive combinatorics. In: STOC, pp. 774–783 (2014)
4. Agrawal, S., Gupta, D., Maji, H.K., Pandey, O., Prabhakaran, M.: Explicit non-malleable codes against bit-wise tampering and permutations. In: Gennaro, R., Robshaw, M. (eds.) CRYPTO 2015. LNCS, vol. 9215, pp. 538–557. Springer, Heidelberg (2015). https://doi.org/10.1007/978-3-662-47989-6_26
5. Agrawal, S., Gupta, D., Maji, H.K., Pandey, O., Prabhakaran, M.: A rate-optimizing compiler for non-malleable codes against bit-wise tampering and permutations. In: Dodis, Y., Nielsen, J.B. (eds.) TCC 2015. LNCS, vol. 9014, pp. 375–397. Springer, Heidelberg (2015). https://doi.org/10.1007/978-3-662-46494-6_16
6. Ball, M., Dachman-Soled, D., Kulkarni, M., Malkin, T.: Non-malleable codes for bounded depth, bounded fan-in circuits. In: Fischlin, M., Coron, J.-S. (eds.) EUROCRYPT 2016. LNCS, vol. 9666, pp. 881–908. Springer, Heidelberg (2016). https://doi.org/10.1007/978-3-662-49896-5_31
7. Ball, M., Dachman-Soled, D., Kulkarni, M., Malkin, T.: Non-malleable codes from average-case hardness: AC^0, decision trees, and streaming space-bounded tampering. Cryptology ePrint Archive, Report 2017/1061 (2017)
8. Bao, F., Deng, R.H., Han, Y., Jeng, A., Narasimhalu, A.D., Ngair, T.: Breaking public key cryptosystems on tamper resistant devices in the presence of transient faults. In: Christianson, B., Crispo, B., Lomas, M., Roe, M. (eds.) Security Protocols 1997. LNCS, vol. 1361, pp. 115–124. Springer, Heidelberg (1998). https://doi.org/10.1007/BFb0028164
9. Bellare, M., Tessaro, S., Vardy, A.: Semantic security for the wiretap channel. In: Safavi-Naini, R., Canetti, R. (eds.) CRYPTO 2012. LNCS, vol. 7417, pp. 294–311. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-32009-5_18
10. Biham, E., Shamir, A.: Differential fault analysis of secret key cryptosystems. In: Kaliski, B.S. (ed.) CRYPTO 1997. LNCS, vol. 1294, pp. 513–525. Springer, Heidelberg (1997). https://doi.org/10.1007/BFb0052259
11. Boneh, D., DeMillo, R.A., Lipton, R.J.: On the importance of checking cryptographic protocols for faults. In: Fumy, W. (ed.) EUROCRYPT 1997. LNCS, vol. 1233, pp. 37–51. Springer, Heidelberg (1997). https://doi.org/10.1007/3-540-69053-0_4
12. Boneh, D., DeMillo, R.A., Lipton, R.J.: On the importance of eliminating errors in cryptographic computations. J. Cryptol. **14**(2), 101–119 (2001)
13. Boyko, V.: On the security properties of OAEP as an all-or-nothing transform. In: Wiener, M. (ed.) CRYPTO 1999. LNCS, vol. 1666, pp. 503–518. Springer, Heidelberg (1999). https://doi.org/10.1007/3-540-48405-1_32

14. Canetti, R., Dodis, Y., Halevi, S., Kushilevitz, E., Sahai, A.: Exposure-resilient functions and all-or-nothing transforms. In: Preneel, B. (ed.) EUROCRYPT 2000. LNCS, vol. 1807, pp. 453–469. Springer, Heidelberg (2000). https://doi.org/10.1007/3-540-45539-6_33
15. Chandran, N., Goyal, V., Mukherjee, P., Pandey, O., Upadhyay, J.: Block-wise non-malleable codes. IACR Cryptology ePrint Archive, p. 129 (2015)
16. Chandran, N., Kanukurthi, B., Raghuraman, S.: Information-theoretic local non-malleable codes and their applications. In: Kushilevitz, E., Malkin, T. (eds.) TCC 2016-A. LNCS, vol. 9563, pp. 367–392. Springer, Heidelberg (2016). https://doi.org/10.1007/978-3-662-49099-0_14
17. Chattopadhyay, E., Zuckerman, D.: Non-malleable codes against constant split-state tampering. In: FOCS, pp. 306–315 (2014)
18. Cheraghchi, M., Guruswami, V.: Capacity of non-malleable codes. In: ITCS 2014 (2014)
19. Choi, S.G., Kiayias, A., Malkin, T.: BiTR: built-in tamper resilience. In: Lee, D.H., Wang, X. (eds.) ASIACRYPT 2011. LNCS, vol. 7073, pp. 740–758. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-25385-0_40
20. Coretti, S., Maurer, U., Tackmann, B., Venturi, D.: From single-bit to multi-bit public-key encryption via non-malleable codes. In: Dodis, Y., Nielsen, J.B. (eds.) TCC 2015. LNCS, vol. 9014, pp. 532–560. Springer, Heidelberg (2015). https://doi.org/10.1007/978-3-662-46494-6_22
21. Cramer, R., Dodis, Y., Fehr, S., Padró, C., Wichs, D.: Detection of algebraic manipulation with applications to robust secret sharing and fuzzy extractors. In: Smart, N. (ed.) EUROCRYPT 2008. LNCS, vol. 4965, pp. 471–488. Springer, Heidelberg (2008). https://doi.org/10.1007/978-3-540-78967-3_27
22. Dachman-Soled, D., Kulkarni, M., Shahverdi, A.: Locally decodable and updatable non-malleable codes in the bounded retrieval model. Cryptology ePrint Archive, Report 2017/303 (2017). http://eprint.iacr.org/2017/303
23. Dachman-Soled, D., Kulkarni, M., Shahverdi, A.: Tight upper and lower bounds for leakage-resilient, locally decodable and updatable non-malleable codes. In: Fehr, S. (ed.) PKC 2017. LNCS, vol. 10174, pp. 310–332. Springer, Heidelberg (2017). https://doi.org/10.1007/978-3-662-54365-8_13
24. Dachman-Soled, D., Liu, F.-H., Shi, E., Zhou, H.-S.: Locally decodable and updatable non-malleable codes and their applications. In: Dodis, Y., Nielsen, J.B. (eds.) TCC 2015. LNCS, vol. 9014, pp. 427–450. Springer, Heidelberg (2015). https://doi.org/10.1007/978-3-662-46494-6_18
25. Döttling, N., Nielsen, J.B., Obremski, M.: Information theoretic continuously non-malleable codes in the constant split-state model. Cryptology ePrint Archive, Report 2017/357 (2017). http://eprint.iacr.org/2017/357
26. Dziembowski, S., Kazana, T., Obremski, M.: Non-malleable codes from two-source extractors. In: Canetti, R., Garay, J.A. (eds.) CRYPTO 2013. LNCS, vol. 8043, pp. 239–257. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-40084-1_14
27. Dziembowski, S., Pietrzak, K., Wichs, D.: Non-malleable codes. In: ICS (2010)
28. Faonio, A., Nielsen, J.B.: Non-malleable codes with split-state refresh. In: Fehr, S. (ed.) PKC 2017. LNCS, vol. 10174, pp. 279–309. Springer, Heidelberg (2017). https://doi.org/10.1007/978-3-662-54365-8_12
29. Faust, S., Hostáková, K., Mukherjee, P., Venturi, D.: Non-malleable codes for space-bounded tampering. In: Katz, J., Shacham, H. (eds.) CRYPTO 2017. LNCS, vol. 10402, pp. 95–126. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-63715-0_4

30. Faust, S., Mukherjee, P., Nielsen, J.B., Venturi, D.: Continuous non-malleable codes. In: Lindell, Y. (ed.) TCC 2014. LNCS, vol. 8349, pp. 465–488. Springer, Heidelberg (2014). https://doi.org/10.1007/978-3-642-54242-8_20
31. Faust, S., Mukherjee, P., Nielsen, J.B., Venturi, D.: A tamper and leakage resilient von neumann architecture. In: Katz, J. (ed.) PKC 2015. LNCS, vol. 9020, pp. 579–603. Springer, Heidelberg (2015). https://doi.org/10.1007/978-3-662-46447-2_26
32. Faust, S., Mukherjee, P., Venturi, D., Wichs, D.: Efficient non-malleable codes and key-derivation for poly-size tampering circuits. In: Nguyen, P.Q., Oswald, E. (eds.) EUROCRYPT 2014. LNCS, vol. 8441, pp. 111–128. Springer, Heidelberg (2014). https://doi.org/10.1007/978-3-642-55220-5_7
33. Genkin, D., Ishai, Y., Prabhakaran, M.M., Sahai, A., Tromer, E.: Circuits resilient to additive attacks with applications to secure computation. In: STOC 2014, pp. 495–504 (2014)
34. Jafargholi, Z., Wichs, D.: Tamper detection and continuous non-malleable codes. In: Dodis, Y., Nielsen, J.B. (eds.) TCC 2015. LNCS, vol. 9014, pp. 451–480. Springer, Heidelberg (2015). https://doi.org/10.1007/978-3-662-46494-6_19
35. Katz, J., Lindell, Y.: Introduction to Modern Cryptography (2007)
36. Kiayias, A., Liu, F.-H., Tselekounis, Y.: Practical non-malleable codes from l-more extractable hash functions. In: Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security, CCS 2016, pp. 1317–1328. ACM, New York (2016)
37. Liu, F.-H., Lysyanskaya, A.: Tamper and leakage resilience in the split-state model. In: Safavi-Naini, R., Canetti, R. (eds.) CRYPTO 2012. LNCS, vol. 7417, pp. 517–532. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-32009-5_30
38. Micali, S., Reyzin, L.: Physically observable cryptography. In: Naor, M. (ed.) TCC 2004. LNCS, vol. 2951, pp. 278–296. Springer, Heidelberg (2004). https://doi.org/10.1007/978-3-540-24638-1_16
39. Ozarow, L.H., Wyner, A.D.: Wire-tap channel II. AT&T Bell Lab. Tech. J. **63**(10), 2135–2157 (1984)
40. Resch, J.K., Plank, J.S.: AONT-RS: blending security and performance in dispersed storage systems. In: FAST 2011 (2011)
41. Rivest, R.L.: All-or-nothing encryption and the package transform. In: Biham, E. (ed.) FSE 1997. LNCS, vol. 1267, pp. 210–218. Springer, Heidelberg (1997). https://doi.org/10.1007/BFb0052348
42. Shaltiel, R., Silbak, J.: Explicit list-decodable codes with optimal rate for computationally bounded channels. In: APPROX/RANDOM 2016 (2016)
43. Stinson, D.R.: Something about all or nothing (transforms). Des. Codes Crypt. **22**(2), 133–138 (2001)
44. Tunstall, M., Mukhopadhyay, D., Ali, S.: Differential fault analysis of the advanced encryption standard using a single fault. In: Ardagna, C.A., Zhou, J. (eds.) WISTP 2011. LNCS, vol. 6633, pp. 224–233. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-21040-2_15
45. Wyner, A.D.: The wire-tap channel. Bell Syst. Tech. J. **54**(8), 1355–1387 (1975)