

Linear-Time Non-Malleable Codes in the Bit-Wise Independent Tampering Model

Ronald Cramer¹, Ivan Damgård², Nico Döttling², Irene Giacomelli², and Chaoping Xing³

¹ CWI Amsterdam & Mathematical Institute, Leiden University, The Netherlands.

² Department of Computer Science, Aarhus University, Denmark.

³ Division of Mathematical Sciences, Nanyang Technological University, Singapore.

April 20, 2016

Abstract. Non-malleable codes were introduced by Dziembowski et al. (ICS 2010) as coding schemes that protect a message against tampering attacks. Roughly speaking, a code is non-malleable if decoding an adversarially tampered encoding of a message \mathbf{m} produces the original message \mathbf{m} or a value \mathbf{m}' (eventually \perp) completely unrelated with \mathbf{m} . Moreover, the probability of which one of these two events happens is also independent of \mathbf{m} . It is known that non-malleability is possible only for restricted classes of tampering functions. Since their introduction, a long line of works has established feasibility results of non-malleable codes against different families of tampering functions. However, for many interesting families the challenge of finding “good” non-malleable codes remains open. In particular, we would like to have *explicit constructions* of non-malleable codes with *high-rate* and efficient encoding/decoding algorithms (*i.e.* low computational complexity). In this work we present two explicit constructions: the first one is a natural generalization of the seminal work of Dziembowski et al. and gives rise to the first constant-rate non-malleable code with *linear-time* complexity (in a model including bit-wise independent tampering). The second construction is inspired by the recent works about non-malleable codes of Agrawal et al. (TCC 2015) and of Cheraghchi and Guruswami (TCC 2014) and improves the previous result in the bit-wise tampering model: it builds the first non-malleable codes with *linear-time* complexity and *optimal-rate* (*i.e.* rate $1 - o(1)$).

1 Introduction

Non-malleable codes [DPW10] are a relaxation of error-correcting and error-detecting codes that have useful applications in cryptography. For example, they can be used to protect keys that are stored in non-robust devices against tampering attacks. Recently, they also found application to computational cryptography (*e.g.* construction of non-malleable commitments [AGM⁺15b], domain extension for public-key encryption schemes [CMTV15,CDTV16]). Roughly speaking, a

coding scheme (Enc, Dec) is non-malleable with respect to the tampering function f if decoding $f(\text{Enc}(\mathbf{m}))$ produces the original message \mathbf{m} or a value \mathbf{m}' (eventually \perp) completely unrelated with \mathbf{m} . Moreover, the probability of which one of these two events happens is also independent of \mathbf{m} . As an illustration of the notion, consider a key that is stored in a device. The adversary is able to tamper with the key and gets to see the effect of using the device with the tampered key inside. If the key was coded with a non-malleable code and is decoded before use, this attack becomes useless, as the key actually used after tampering is either unchanged or is unrelated to the original key.

Since a tampering function can always try to decode, modify the message, and encode again, it is clear that non-malleable codes are impossible without restrictions on the tampering function. We therefore restrict the adversary to using functions from a specific class \mathcal{F} . In this case, we say that we have a non-malleable code with respect to the family \mathcal{F} . For example, if the encoding is made by n symbols from a finite field \mathbb{F} , then we can restrict the tampering function to be a function with n independent components (f_1, \dots, f_n) (symbol-wise independent tampering, or bit-wise independent tampering if $\mathbb{F} = \{0, 1\}$). Other important features of the coding scheme are the rate and the computational complexity⁴. Since 2010, a line of works has established increasingly stronger results concerning the feasibility of non-malleable codes against different families of tampering functions. However, for many interesting families the challenge of finding “good” non-malleable codes remains open. In particular, we would like to have *explicit constructions* of non-malleable codes with *high rate* and efficient encoding/decoding algorithm (*i.e.* low computational complexity).

Many of the known constructions of non-malleable codes, such as [DPW10], [CG14b, AGM⁺15a, AGM⁺15b] use *linear secret-sharing schemes* (LSSS) as one of the main building blocks. Roughly speaking, a secret-sharing scheme is a randomised algorithm that encodes a message \mathbf{m} as a longer vector \mathbf{s} such that \mathbf{m} can be computed from large enough sets of entries in \mathbf{s} , while smaller set give no information about \mathbf{m} . LSSS with extra properties (uniformity and distance) are used already by Dziembowski et al. in [DPW10] where they introduce and motivate the formal notion of non-malleable codes and also construct the first family of non-malleable codes in the bit-wise independent tampering model. The computational complexity of the code is quadratic in the size of the input length. Secondly, via the probabilistic method they show that for any family \mathcal{F} of tampering functions such that $|\mathcal{F}| \leq 2^{2^\alpha n}$ for some constant $\alpha < 1$ (n is the length of the encoding) there exist constant-rate non-malleable codes with respect to \mathcal{F} . In this case, the description of the code is of exponential size, thus the encoding and decoding algorithms are inefficient. More recently, Cheraghchi and Guruswami [CG14a] prove that for this kind of families the optimal rate is $1 - \alpha$; they construct non-malleable codes approaching this rate. Again, the con-

⁴ The rate of the coding scheme (Enc, Dec) is the quotient of the length of the message \mathbf{m} over the length of its encoding $\text{Enc}(\mathbf{m})$. The computational complexity of the scheme is maximum of the computational complexities of the two algorithm Enc and Dec in function of the length of \mathbf{m} .

struction is non-explicit and gives rise to inefficient codes. For families of single exponential size, *i.e.* $|\mathcal{F}| \leq 2^{p(n)}$ for some polynomial p , efficient (*i.e.* polynomial time) non malleable codes were constructed in [FMVW14]. This construction is also randomized, *i.e.* the construction succeeds with overwhelming probability in providing non-malleable codes achieving optimal rate $1 - o(1)$.

On the other hand, in [CG14b] an explicit (deterministic) construction of non-malleable codes with rate arbitrarily close to 1 in the bit-wise independent tampering model is given. The construction is based on the concatenation of a linear error-correcting secret-sharing scheme of rate close to 1 and a constant-size non-malleable code. This construction is instantiated using Reed-Solomon codes and has thus computational complexity at least $O(n \log(n))$ (super-linear).

Bit-wise independent tampering functions act on each bit of the encoding independently. In the more general C -split state model the encoding is partitioned into C blocks (C is a constant) and each block can be tampered arbitrarily but independently of the others blocks (*e.g.* [CKM11]). For $C = 10$, an efficient and explicit construction of constant rate non-malleable codes was given in [CZ14]. Several results can be found in the recent literature when $C = 2$ (2-split-state model) [LL12,DKO13,FMNV14,CG14b]. The first explicit construction of non-malleable codes in this model [ADL14] had rate 0. Very recently Aggarwal et al. [ADKO15] constructed the first efficient and explicit non-malleable codes in the 2-split state model achieving constant rate.

In [AGM⁺15a,AGM⁺15b], Agrawal et al. construct explicit and non-malleable codes which are simultaneously resilient against bit-wise independent tampering and permutations. They get optimal rate, but super-linear time.

In a recent work [JW15], Jafargholi and Wichs introduce *tamper-detection codes* and use them together with leakage-resilient codes [DDV10] to construct non-malleable codes that achieve optimal rate when $|\mathcal{F}| \leq 2^{2\alpha n}$ and efficient encoding and decoding when $|\mathcal{F}| \leq 2^{p(n)}$. Tamper-detection codes for the simple family of additive tampering functions are called *algebraic manipulation detection codes* (AMD) and were already introduced by Cramer et al. in 2008 [CDF⁺08].

In conclusion, the natural question to ask is if we can achieve the optimal properties of linear-time complexity and rate approaching 1 simultaneously. This is not known, even for the restricted case of bit-wise tampering, and even we only ask for linear-time complexity⁵.

Our Contribution. In this paper, we study the above question and achieve positive results.

In the first part of our work, we push forward the idea of using linear secret sharing, and show that when the family of tampering functions has a clear structure (as in the symbol-wise independent tampering model), then simple constructions based on LSSS can achieve good results: we get constant-rate non-

⁵ Determining which cryptographic primitives can be instantiated in linear-time is an interesting and challenging program started by Ishai et al. in [IKOS08].

malleable codes with optimal computational complexity $O(k)$, where k is the length of the input message. To obtain this, we also use the recent results about linear-time encodable error-correcting codes and linear-time computable universal hash functions [IKOS08,DI14].

Building on the first result, we then achieve both linear-time complexity and optimal rate, that is rate $1 - o(1)$, for non-malleable codes in the bit-wise independent tampering model. It is instructive to observe that optimal-rate non-malleable codes with superlinear time complexity were constructed in [CG14b,AGM⁺15a], and that these codes are based on secret sharing schemes with (relatively) large privacy and reconstruction thresholds. The problem we face is that there are no constructions of linear secret sharing schemes with linear-time complexity for the required parameter range⁶. We therefore propose a novel construction which is based on slightly weaker primitives which can be instantiated for the rate $1 - o(1)$ and linear-time complexity regime.

Overview of our Constructions. As mentioned, we present two deterministic constructions for linear-time non-malleable codes: Construction 1 can be seen as a generalization of the original construction of [DPW10] and gives rise to the first linear-time non-malleable codes with constant rate in the symbol-wise independent tampering model. More generally, we prove that given a family of tamper-detection codes with any computational complexity and rate, it is possible to *explicitly* construct a family of non-malleable codes with constant rate and linear-time complexity (Theorem 2). The other ingredients of this first construction are constant-rate AMD codes and constant-rate LSSS with good privacy (but where one needs almost all shares to reconstruct). We present linear-time instantiations of both these primitives using the results of [DI14]. Construction 1 encodes a message \mathbf{m} with three sequential steps: first \mathbf{m} is encoded with an AMD code, then the result is shared by a LSSS with privacy and finally each share is encoded by a tamper-detection code. In particular, in Construction 1 if

$$\begin{array}{ccccccc}
 \mathbb{F}^k & \xrightarrow{\text{AMD}} & \mathbb{F}^{\Theta(k)} & \xrightarrow{\text{LSSS}} & (\mathbb{F}^\ell)^m & \xrightarrow{\text{TD}^m} & (\mathbb{F}^{\ell'})^m \\
 \mathbf{m} & \longmapsto & \mathbf{s} & \longmapsto & \mathbf{s} & & \\
 & & & & \parallel & & \\
 & & & & (\mathbf{s}_1, \dots, \mathbf{s}_m) & \longmapsto & (\mathbf{c}_1, \dots, \mathbf{c}_m)
 \end{array}$$

Fig. 1. The encoding algorithm of Construction 1

the tamper-detection code is secure against the family of tampering function \mathcal{F}

⁶ A recent Monte-Carlo construction by Cramer et al. [CDD⁺15] can be instantiated for a parameter range where the rate of the secret sharing scheme is bounded away from 1 by a constant, but not for rate approaching 1.

with constant error, then the resulting code is non-malleable respect to the family \mathcal{F}^+ of functions (f_1, \dots, f_m) where each f_i is a function from \mathcal{F} , a constant function or the identity and it has error negligible in the length of the input. Hence, depending on how one instantiates the components of the construction, one can handle more general tampering models than bit-wise (*e.g.* symbol-wise tampering)⁷. A key point for the efficiency is that the shares produced by the LSSS used are of constant size (*i.e.* $m = \Theta(k)$ and ℓ constant). This implies that applying the tamper-detection code to all the shares results only in a constant overhead for the computational complexity.

With Construction 2, we achieve linear-time non-malleable codes with optimal rate approaching 1, still with an explicit (deterministic) construction (Theorem 4). The most efficient constructions of optimal rate non-malleable codes are from [CG14b, AGM⁺15a]. Both these constructions require a secret sharing scheme with good privacy and non-trivial reconstruction threshold. Together with the rate close to 1 constraint, these are challenging features to achieve in linear-time. In our construction, we also use a secret-sharing scheme with rate close to 1, but we do not require any reconstruction property for this scheme. Instead, we combine the sharing scheme with other two tailored primitives, each implementable in linear-time, and a short constant-rate non-malleable code. The modular design of our construction makes the security proof much simpler and more intuitive than previous constructions: each primitive takes care of a specific property needed to prove non-malleability. The encoding is done in the following way: first the input message is shared with a sharing scheme that has rate $1 - o(1)$ and t -uniformity (that is, if \mathbf{s} is the share vector of \mathbf{m} , then each set of t components of \mathbf{s} are distributed uniformly on \mathbb{F}^t). Then we use the two tailored primitives: first, a keyed almost universal function is used to compute the first hash of \mathbf{s} , $h_{\mathbf{k}}(\mathbf{s})$. Second, we compute short deterministic hash $\text{Com}(\mathbf{s})$, using a new primitive that we call a *compressor*. This compressed value $\text{Com}(\mathbf{s})$ comes with the guaranty of having high entropy. The two hash values and the key for the almost universal hash function can be thought of as an “authentication tag” of \mathbf{m} . The final encoding is given by the share vector \mathbf{s} and a non-malleable encoding of this tag, this encoding does not have to be high-rate nor linear-time.

$$\begin{array}{ccccccc}
 \mathbb{F}^k & \xrightarrow{\text{sharing}} & \mathbb{F}^{k+o(k)} & \xrightarrow{\text{hashing}} & \mathbb{F}^{k+o(k)} \times \mathbb{F}^{o(k)} \times \mathbb{F}^{o(k)} & \xrightarrow{\text{short NM}} & \mathbb{F}^{k+o(k)} \times \mathbb{F}^{o(k)} \\
 \mathbf{m} & \longmapsto & \mathbf{s} & \longmapsto & (\mathbf{s}, h_{\mathbf{k}}(\mathbf{s}), \text{Com}(\mathbf{s})) & & \\
 & & & & \parallel & & \\
 & & & & (\mathbf{s}, \mathbf{h}, \mathbf{c}) & \longmapsto & (\mathbf{s}, \text{NM}(\mathbf{k}, \mathbf{h}, \mathbf{c}))
 \end{array}$$

Fig. 2. The encoding algorithm of Construction 2

⁷ Notice that the concrete instantiation we give in Corollary 3 leads to bit-wise tampering.

Structure of the paper: In Section 2, we fix the notation and give the basic definitions we need further on in the paper. In Section 3 first we give linear-time construction for AMD codes and LSSS with privacy, then we present Construction 1 in general and finally, we instantiate it for the binary case (bit-wise independent tampering model). Section 4 is also divided in two parts: in the first one we define and instantiate the primitives that are necessary for Construction 2; the latter is described in the second part of the section together with the bit-wise independent tampering model result.

2 Preliminaries

2.1 Notation

For an integer n , we write $[n] = \{1, 2, \dots, n\}$ and, given $A \subseteq [n]$, $|A|$ denotes the cardinality of A , while A^c indicates the complement set of A , i.e. $A^c = [n] \setminus A$. With the notation (z_1, \dots, z_n) we indicate an element of the n -times cartesian product of \mathbb{F}^ℓ , where \mathbb{F} is a finite field of cardinality q and ℓ is a positive integer. Given $\mathbf{z} = (z_1, \dots, z_n) \in (\mathbb{F}^\ell)^n$ and a subset $A \subseteq [n]$, we will use \mathbf{z}_A to denote the vector $(z_i)_{i \in A} \in (\mathbb{F}^\ell)^{|A|}$. Given two vectors $\mathbf{z} = (z_1, \dots, z_n), \mathbf{v} = (v_1, \dots, v_n) \in (\mathbb{F}^\ell)^n$, the *generalized Hamming Distance* between \mathbf{z} and \mathbf{v} is defined by $d_{\text{Ham}}^\ell(\mathbf{z}, \mathbf{v}) = |\{i \in [n] \mid z_i \neq v_i\}|$.

If Alg is an algorithm (randomized or not) that takes as input a value from \mathbb{F}^n , then the computational complexity of Alg is the number of field elementary operations that Alg executes to compute the output.

We indicate with id the identity function. We say that a function ε is *negligible* in n ($\varepsilon(n) = \text{negl}(n)$) if for every polynomial p there exists a constant c such that $\varepsilon(n) < \frac{1}{p(n)}$ when $n > c$.

2.2 Probability

For a random variable X , the notation $v \leftarrow X$ denotes that v is sampled randomly according to X . For a set S , $v \leftarrow S$ denotes that v is sampled uniformly at random from S . Given two random variables X and Y with finite range S , the *statistical distance* between X and Y is defined as $\text{SD}(X, Y) = \frac{1}{2} \sum_{i \in S} |\Pr[X = i] - \Pr[Y = i]|$.

Let $X = (X_1, \dots, X_n)$ be a random variable with range S^n and t be a positive integer less or equal to n . We say that X is *t-wise independent* if for any $A = \{i_1, \dots, i_t\}$ subset of $[n]$ of cardinality t and for any vector $\mathbf{b} = (b_1, \dots, b_t) \in S^t$, it holds that $\Pr[X_A = \mathbf{b}] = \prod_{j=1}^t \Pr[X_{i_j} = b_j]$. We say that X is *t-wise uniform* on S^n if for any $A \subseteq [n]$ of cardinality t , X_A has the uniform distribution on S^t . If $t = n$ we simply say that X is an uniform random variable on S^n .

The *min-entropy* of X is defined by $H_\infty(X) = -\log_2(\max_{\mathbf{b}} \Pr[X = \mathbf{b}])$. Note that if $X = (X_1, \dots, X_n)$ and X_A is $|A|$ -wise independent then $H_\infty(X) \geq \sum_{i \in A} H_\infty(X_i)$ and $H_\infty(\sum_{i \in A} X_i) \geq \max_{i \in A} H_\infty(X_i)$.

2.3 Tamper-Detection and Non-Malleability

Let \mathbb{F} be a finite field and n, ℓ, k be positive integers. An ℓ -folded n -code over \mathbb{F} is a non-empty subset of $(\mathbb{F}^\ell)^n$; we will refer to n as the length of the code. Given a set $A \subseteq [n]$, with the notation \mathcal{C}_A we indicate the set $\{\mathbf{c}_A \mid \mathbf{c} \in \mathcal{C}\}$. If $\psi : \mathcal{C} \rightarrow \mathbb{F}^k$ is a regular function, the pair (\mathcal{C}, ψ) is called ℓ -folded (n, k) -coding scheme. The rate of a scheme is the ratio $k/\ell n$. If $\mathbb{F} = \{0, 1\}$, the scheme is called binary. If \mathcal{C} is a vector space over \mathbb{F} , then the code is called *linear*. The dimension of a linear code is its dimension as vector space over \mathbb{F} . Moreover, if the map ψ is an \mathbb{F} -linear map, also the scheme (\mathcal{C}, ψ) is called linear.

Remark 1. Given an ℓ -folded (n, k) -coding scheme (\mathcal{C}, ψ) any randomized algorithm $\text{Enc} : \mathbb{F}^k \rightarrow \mathcal{C}$ that on input $\mathbf{m} \in \mathbb{F}^k$ outputs $\mathbf{c} \in \psi^{-1}(\{\mathbf{m}\})$ selected uniformly at random is called *encoding algorithm*. On the other side, the *decoding algorithm* is the name used for the deterministic algorithm $\text{Dec} : (\mathbb{F}^\ell)^n \rightarrow \mathbb{F}^k \cup \{\perp\}$ that maps \mathbf{c} to $\mathbf{m} = \psi(\mathbf{c}) \in \mathbb{F}^k$ if $\mathbf{c} \in \mathcal{C}$ and to \perp otherwise. For convenience⁸, in the following we will always identify a coding scheme (\mathcal{C}, ψ) with the pair (Enc, Dec) .

While keeping \mathbb{F} fixed, we will assume throughout that $n = n(k)$. The computational complexity (as a function of k) of a coding scheme is the maximum taken over the computational complexities of Enc and Dec , respectively. We say that a coding scheme is *linear-time* if both Enc and Dec have complexity $O(k)$.

Let (Enc, Dec) be an ℓ -folded (n, k) -coding scheme over \mathbb{F} . Given an encoding $\mathbf{c} \leftarrow \text{Enc}(\mathbf{m})$ for the message $\mathbf{m} \in \mathbb{F}^k$, tampering with \mathbf{c} can be represented by considering a function $f : (\mathbb{F}^\ell)^n \rightarrow (\mathbb{F}^\ell)^n$ that modifies the encoding \mathbf{c} in $\tilde{\mathbf{c}} = f(\mathbf{c})$. The output of $\text{Dec}(\tilde{\mathbf{c}})$ now depends on the original message \mathbf{m} and also on the tampering function f . To represent this, we consider the following random variable $\text{Real}_f^{\mathbf{m}}$.

$$\text{Real}_f^{\mathbf{m}} := \begin{cases} \text{sample } \mathbf{c} \leftarrow \text{Enc}(\mathbf{m}); \\ \text{compute } \tilde{\mathbf{c}} = f(\mathbf{c}); \\ \text{output } \tilde{\mathbf{m}} = \text{Dec}(\tilde{\mathbf{c}}); \end{cases}$$

A simple but strong property that we can ask for is that the coding scheme is able to detect with overwhelming probability the tampering caused by all the functions f from a specific family \mathcal{F} .

Definition 1 (TD Code, [JW15]). *Given a family \mathcal{F} of functions over $(\mathbb{F}^\ell)^n$, an (n, k) -tamper detection code with respect to \mathcal{F} and with error ϵ is an (n, k) -coding scheme such that $\Pr[\text{Real}_f^{\mathbf{m}} \neq \perp] \leq \epsilon$ for any $\mathbf{m} \in \mathbb{F}^k$ and any $f \in \mathcal{F}$.*

⁸ The two definitions are equivalent. Given the pair (Enc, Dec) such that for any \mathbf{m} it holds $\Pr[\text{Dec}(\text{Enc}(\mathbf{m})) = \mathbf{m}] = 1$, define \mathcal{C} as the image of Enc in $(\mathbb{F}^\ell)^n$ and ψ as the map Dec restricted to \mathcal{C} .

For example, any error-correcting code from coding theory with minimal distance d is a TD code with respect to the family \mathcal{F}_{dist} of functions that modify less than d components in the input vector (i.e. $d_{\text{Ham}}^\ell(f(\mathbf{x}), \mathbf{x}) < d$).

The name *algebraic manipulation detection (AMD) code*, introduced by [CDF⁺08], is used for TD codes with respect to the family \mathcal{F}_{amd} of additive tampering functions. That is, functions of the form $f_\Delta(\mathbf{x}) = \mathbf{x} + \Delta$ where the vector Δ is a non-zero constant vector independent of \mathbf{x} .

Unfortunately, tampering detection can not be achieved for many natural families. For example, consider the family \mathcal{F}_{const} of all constant functions $f_{\mathbf{c}}(\mathbf{x}) = \mathbf{c}$ for $\mathbf{c} \in (\mathbb{F}^\ell)^n$; if \mathbf{c} is a valid encoding, then $\Pr[\text{Real}_{f_{\mathbf{c}}}^{\mathbf{m}} \neq \perp] = 1$ for all $\mathbf{m} \in \mathbb{F}^k$. In order to be able to consider larger families of tampering functions, the definition of tampering detection needs to be relaxed. Instead of asking that the tampering is detected, we can ask that the result of the tampering action is independent of the original message. This property, called non-malleability is weaker than tampering-detection, nevertheless it offers enough protection against tampering attacks: an adversary that actively modifies encoded data can not control the practical effect of his action on the encoded message.

Definition 2 (NM Code, [DPW10]). An (n, k) -coding scheme (Enc, Dec) is said to be non-malleable with respect to a family \mathcal{F} with error ϵ if the following holds for any $f \in \mathcal{F}$. There exists a random variable D_f on $\mathbb{F}^k \cup \{\perp, \text{same}\}$ such that, given

$$\text{Ideal}_f^{\mathbf{m}} := \begin{cases} \text{sample } \mathbf{m}^* \leftarrow D_f; \\ \text{if } \mathbf{m}^* = \text{same} & \text{then } \mathbf{m}' = \mathbf{m}; \\ & \text{otherwise } \mathbf{m}' = \mathbf{m}^*; \\ \text{output } \mathbf{m}'; \end{cases}$$

then $\text{SD}(\text{Real}_f^{\mathbf{m}}, \text{Ideal}_f^{\mathbf{m}}) \leq \epsilon$ for any $\mathbf{m} \in \mathbb{F}^k$.

In the rest of the paper we will mainly consider the family of *symbol-wise independent tampering* functions. That is, if the encoding has the form $\mathbf{c} = (\mathbf{c}_1, \dots, \mathbf{c}_n) \in (\mathbb{F}^\ell)^n$, then each component \mathbf{c}_i can be modified arbitrarily but independently of the values of the others components. We will use the following notation:

$$\mathcal{F}_{\ell, n}^q = \{f = (f_1, \dots, f_n) \mid f_i : \mathbb{F}^\ell \rightarrow \mathbb{F}^\ell\}$$

and $f(\mathbf{c}) = (f_1(\mathbf{c}_1), \dots, f_n(\mathbf{c}_n))$. Note that if $q = 2$ and $\ell = 1$, $\mathcal{F}_{1, n}^2$ is the family considered in the *bit-wise independent tampering* model.

2.4 Secret-Sharing

Suppose that (Enc, Dec) is an ℓ -folded (n, k) -coding scheme over \mathbb{F} . Let t, r be positive integers.

Definition 3 (t -privacy). (Enc, Dec) has t -privacy if the following holds for each set $A \subset [n]$ of \mathbb{F}^ℓ -coordinates with $|A| = t$. For each $\mathbf{m}, \mathbf{m}' \in \mathbb{F}^k$, the

distributions of $(\text{Enc}(\mathbf{m}))_A$ and $(\text{Enc}(\mathbf{m}'))_A$ on $(\mathbb{F}^\ell)^t$ are identical. The scheme has t -uniformity if these distributions are the uniform ones on $(\mathbb{F}^\ell)^t$.

Definition 4 (r -reconstruction). (Enc, Dec) has r -reconstruction if the following holds for each set $A \subset [n]$ of \mathbb{F}^ℓ -coordinates with $|A| = r$. If $\mathbf{c}, \mathbf{c}' \in \mathcal{C}$ satisfy $\mathbf{c}_A = \mathbf{c}'_A$, then $\text{Dec}(\mathbf{c}) = \text{Dec}(\mathbf{c}')$.

Note that any scheme has n -reconstruction. Moreover, if the coding scheme has r -reconstruction and t -privacy, then $t < r$.

Remark 2. Given an ℓ -folded linear (n, k) -coding scheme, it is easy to prove that t -privacy and t -uniformity are equivalent to the following conditions, respectively.

- (t -privacy) for each set $A \subseteq [n]$ of \mathbb{F}^ℓ -coordinates with $|A| = t$, the map that maps \mathbf{c} in \mathcal{C} to the pair $(\text{Dec}(\mathbf{c}), \mathbf{c}_A)$ is surjective;
- (t -uniformity) the same condition as before holds and moreover $\mathcal{C}_A = (\mathbb{F}^\ell)^t$.

Definition 5 (Secret-Sharing Scheme (SSS)). An ℓ -folded (n, t, r, k) -secret-sharing scheme over \mathbb{F} (with uniformity) is an ℓ -folded (n, k) -coding scheme over \mathbb{F} with t -privacy (t -uniformity) and r -reconstruction.

If the coding scheme is linear then we call it linear secret-sharing scheme (LSSS).

Notice that in the existing literature, the algorithms Enc and Dec of a SSS are often indicated with the notation Sh (*sharing algorithm*) and Rec (*reconstruction algorithm*), respectively. Later on in the paper we will use this notation.

In this work, we will use secret-sharing schemes with different parameters and properties as building blocks for constructing efficient NM codes. In particular, for Construction 1 we are interested in the following aspect: what happens if the reconstruction algorithm of a t -private LSSS is applied to a share vector where at most t components have been tampered arbitrarily. The answer is stated in the next lemma.

Lemma 1. Let (Sh, Rec) a t -private ℓ -folded (n, k) -LSSS. Consider a set $I \subseteq [n]$ of \mathbb{F}^ℓ -coordinates with $|I| \geq n - t$ and a function $f = (f_1, \dots, f_n) \in \mathcal{F}_{\ell, n}^q$ such that $f_i = \text{id}$ for all $i \in I$. Then there exists a random variable Δ_f on $(\mathbb{F}^\ell)^n \cup \{\perp\}$ such that for any $\mathbf{m} \in \mathbb{F}^k$, $\text{Rec}(f(\text{Sh}(\mathbf{m})))$ has the same distribution of $\mathbf{m} + \Delta_f$ (with the convention that $\mathbf{m} + \perp = \perp$).

Proof. Clearly, $f(\text{Sh}(\mathbf{m}))$ has the same distribution of $\text{Sh}(\mathbf{m}) + [f(\text{Sh}(\mathbf{m})) - \text{Sh}(\mathbf{m})]$. Since $f_i = \text{id}$ for all $i \in I$ and $|I| \geq n - t$, the t -privacy implies that $f(\text{Sh}(\mathbf{m})) - \text{Sh}(\mathbf{m})$ has the same distribution of $f(\text{Sh}(\mathbf{0})) - \text{Sh}(\mathbf{0})$. Define Δ_f as the random variable that outputs $\text{Rec}(f(\text{Sh}(\mathbf{0})) - \text{Sh}(\mathbf{0}))$, then the lemma follows from the linearity property of the map Rec . \square

3 Constant-Rate and Linear-Time NM Codes

In this section, we describe our first main result: Construction 1 (Figure 4) combines an AMD code, a LSSS and a TD code with constant error in order to construct a constant-rate NM code (with negligible error) whose computational complexity is controlled by the complexity of the two first schemes used (the AMD code and the LSSS).

3.1 Building Blocks for Construction 1

Before describing Construction 1, we build *linear-time* constant-rate AMD codes and LSSSs.

AMD Codes

We recall that a coding scheme (Enc, Dec) (with alphabet \mathbb{F}) is an (n, k) -AMD code with error ϵ if for any $\mathbf{m} \in \mathbb{F}^k$ and any non-zero $\Delta \in \mathbb{F}^n$, it holds that $\Pr[\text{Dec}(\text{Enc}(\mathbf{m}) + \Delta) \neq \perp] \leq \epsilon$. This special family of TD codes are of particular interest because, despite their simple definition, they can be used as basic tools of generic constructions for coding scheme that achieve security against tampering family larger than \mathcal{F}_{amd} (see for example [DPW10] and our Construction 1). Clearly, the parameters (*i.e.* the rate) and the efficiency of the final schemes depend on the ones of the AMD codes used. In particular, in order to prove our result about constant-rate and linear-time NM codes (Theorem 2), we need to build constant-rate and linear-time AMD codes. Our construction, presented in the following Corollary 1, is based on the family of linear uniform functions from [DI14].

Lemma 2 (Linear Uniform Family, Theorem 4 in [DI14]). *For any positive integers c and large enough k there exist a positive constant b ($b \geq c$) and a family of functions $\{g_{\mathbf{k}} : \mathbb{F}^k \rightarrow \mathbb{F}^{ck}\}_{\mathbf{k}}$ with $\mathbf{k} \in \mathbb{F}^{bk}$, such that the following holds:*

1. $g_{\mathbf{k}}$ has computational complexity $O(k)$;
2. $g_{\mathbf{k}}$ is \mathbb{F} -linear and $g_{\mathbf{k}_1 + \mathbf{k}_2} = g_{\mathbf{k}_1} + g_{\mathbf{k}_2}$;
3. for any $\mathbf{y} \in \mathbb{F}^{ck}$ and $\mathbf{x} \in \mathbb{F}^k$ with $\mathbf{x} \neq \mathbf{0}$, if \mathbf{k} is chosen uniformly at random from \mathbb{F}^{bk} then $\Pr[g_{\mathbf{k}}(\mathbf{x}) = \mathbf{y}] = \frac{1}{q^{ck}}$.

Corollary 1 (Linear-Time and Constant-Rate AMD code). *For any large enough integer k , there exists a linear-time (k', k) -AMD code with error q^{-k} and $k' = \Theta(k)$.*

Proof. Given k , let \mathcal{G} be the family from Lemma 2 with $c = 1$. For the sake of simplicity we consider separately the cases $b = 1$ and $b > 1$.

First, assume that $b = 1$ and define

- $\text{Enc}_{\text{amd}}(\mathbf{m}) = (\mathbf{m}, \mathbf{k}, \mathbf{r}, g_{\mathbf{k}}(\mathbf{m}), g_{\mathbf{k}}(\mathbf{r}), g_{\mathbf{k}}(\mathbf{k}))$, where $\mathbf{k}, \mathbf{r} \in \mathbb{F}^k$ are chosen uniformly at random.

$$- \text{Dec}_{\text{amd}}(\mathbf{v}_1, \mathbf{v}_2, \mathbf{v}_3, \mathbf{v}_4, \mathbf{v}_5, \mathbf{v}_6) = \begin{cases} \mathbf{v}_1 & \text{if } g_{\mathbf{v}_2}(\mathbf{v}_1) = \mathbf{v}_4, g_{\mathbf{v}_2}(\mathbf{v}_3) = \mathbf{v}_5, g_{\mathbf{v}_3}(\mathbf{v}_2) = \mathbf{v}_3 \\ \perp & \text{otherwise} \end{cases}$$

We will show that $(\text{Enc}_{\text{amd}}, \text{Dec}_{\text{amd}})$ is a $(6k, k)$ -AMD code with error $\frac{1}{q^k}$. That is, given a non-zero error vector $\Delta = (\Delta_1, \Delta_2, \Delta_3, \Delta_4, \Delta_5, \Delta_6) \in \mathbb{F}^{6k}$, we will prove that $\Pr[\text{Dec}_{\text{amd}}(\text{Enc}_{\text{amd}}(\mathbf{m}) + \Delta) \neq \perp] \leq \frac{1}{q^k}$. For this purpose notice that

$$\text{Dec}_{\text{amd}}(\text{Enc}_{\text{amd}}(\mathbf{m}) + \Delta) = \text{Dec}(\mathbf{m} + \Delta_1, \mathbf{k} + \Delta_2, \mathbf{r} + \Delta_3, g_{\mathbf{k}}(\mathbf{m}) + \Delta_4, g_{\mathbf{k}}(\mathbf{r}) + \Delta_5, g_{\mathbf{r}}(\mathbf{k}) + \Delta_6)$$

and that $\Pr[\text{Dec}_{\text{amd}}(\text{Enc}_{\text{amd}}(\mathbf{m}) + \Delta) \neq \perp]$ is equal to the probability that the following equations are all satisfied:

$$\begin{cases} g_{\mathbf{k} + \Delta_2}(\mathbf{m} + \Delta_1) = g_{\mathbf{k}}(\mathbf{m}) + \Delta_4 \\ g_{\mathbf{k} + \Delta_2}(\mathbf{r} + \Delta_3) = g_{\mathbf{k}}(\mathbf{r}) + \Delta_5 \\ g_{\mathbf{r} + \Delta_3}(\mathbf{k} + \Delta_2) = g_{\mathbf{r}}(\mathbf{k}) + \Delta_6 \end{cases}$$

The above system is equivalent to

$$\begin{cases} g_{\mathbf{k}}(\Delta_1) = \Delta_4 - g_{\Delta_2}(\mathbf{m} + \Delta_1) \\ g_{\mathbf{k}}(\Delta_3) = \Delta_5 - g_{\Delta_2}(\mathbf{r} + \Delta_3) \\ g_{\mathbf{r}}(\Delta_2) = \Delta_6 - g_{\Delta_3}(\mathbf{k} + \Delta_2) \end{cases} \quad (1)$$

If at least one among the vectors Δ_1 , Δ_2 and Δ_3 is different from zero (w.l.o.g. assume that $\Delta_1 \neq 0$) then

$$\Pr[\text{Dec}_{\text{amd}}(\text{Enc}_{\text{amd}}(\mathbf{m}) + \Delta) \neq \perp] \leq \Pr[g_{\mathbf{k}}(\Delta_1) = \Delta_4 - g_{\Delta_2}(\mathbf{m} + \Delta_1)] = 1/q^k$$

where that the last inequality holds as \mathcal{G} is a linear uniform family (property 3 in Lemma 2). On the other hand, if $\Delta_1 = \Delta_2 = \Delta_3 = 0$, then system (1) is satisfied if and only if also $\Delta_4 = \Delta_5 = \Delta_6 = 0$. But this situation is not possible since $\Delta \neq \mathbf{0}$. Thus, the proof in this first case is concluded.

If $b > 1$, the previous construction is still possible, but with worse rate. To see this, split both the vectors \mathbf{r} and \mathbf{k} in b pieces of length k and, in the encoding algorithm Enc_{amd} , substitute the vector $g_{\mathbf{k}}(\mathbf{r})$ with the vectors $g_{\mathbf{k}}(\mathbf{r}_1), \dots, g_{\mathbf{k}}(\mathbf{r}_b)$ and the vector $g_{\mathbf{r}}(\mathbf{k})$ with the vectors $g_{\mathbf{r}}(\mathbf{k}_1), \dots, g_{\mathbf{r}}(\mathbf{k}_b)$, respectively. It is easy to verify that in this case we obtain an (k', k) -AMD code with $k' = (3+2b)k$ and error $\frac{1}{q^k}$. By inspection, the computational complexity of the scheme is $O(k)$. \square

Linear Codes and LSSSs

Let \mathcal{D} be an ℓ -folded linear m -code of dimension k over the finite field \mathbb{F} . The minimum distance of \mathcal{D} is defined as $d = \min\{d_{\text{Ham}}^\ell(\mathbf{c}, \mathbf{c}') \mid \mathbf{c}, \mathbf{c}' \in \mathcal{D}, \mathbf{c} \neq \mathbf{c}'\}$. If \mathbf{G} is a $k \times m$ matrix over \mathbb{F}^ℓ , we say that \mathbf{G} is a generator matrix for the code \mathcal{D} if $\mathcal{D} = \{\mathbf{m} \cdot \mathbf{G} \mid \mathbf{m} \in \mathbb{F}^k\}$. We say the \mathcal{D} is a *linear-time encodable code* if the map $\mathbf{m} \rightarrow \mathbf{m} \cdot \mathbf{G}$ can be computed by an algorithm that has computational

complexity $O(k)$.

For Construction 1, we are interested in linear-time (m, t, m, k) -LSSS with large privacy (*i.e.* $t > m/2$) and constant-rate. Recently [CDD⁺15], the first linear-time constant-rate LSSS was shown, using a construction based on a combination of suitable linear codes and universal hash functions. More concretely, while being linear over a fixed finite field and supporting an unbounded number of players (or shares) m , there are constants $\epsilon_T, \epsilon_t, \epsilon_r$ with $0 < \epsilon_s, \epsilon_t, \epsilon_r < 1$ and an integer ℓ (the share size) such that the length k of the secret satisfies $k \geq \epsilon_s \ell m$, the privacy parameter t satisfies $t \geq \epsilon_t m$ and the reconstruction parameter r satisfies $r \leq \epsilon_r m$. Moreover, both the sharing and the reconstruction algorithm have complexity linear in m . Although here we also need constant-rate linear-time sharing scheme, we do not use the result from for Construction 1 and instead we construct our constant-rate linear-time sharing scheme for two reasons. First, the construction in [CDD⁺15] is a Monte-Carlo construction, while in this work we are interested only in explicit (deterministic) constructions. Second, later on (Section 4) we will require constant-rate sharing scheme with t -uniformity (instead of only t -privacy). Our schemes from Corollary 2 have this extra property that is not satisfied by the schemes presented in [CDD⁺15].

Now we present how to construct the required LSSS using linear codes. In particular, we are going to use the explicit family of linear-time encodable codes with good distance from [DI14]:

Lemma 3 (Linear-Time Codes, Theorem 2 in [DI14]). *For any real number $\delta \in (0, 1)$ and large enough integer k , there exist a real number $\rho \in (0, 1)$, a positive integer ℓ and a code over \mathbb{F} such that the following hold. The code is ℓ -folded; if m is the length of the code and d is its minimum distance, then $\frac{k}{\ell} < m \leq \frac{k}{\ell\rho}$ and $d \geq \delta m$. Furthermore, the code is linear-time encodable.*

The following Lemma generalizes and rephrases Theorem 2 in [CGH⁺85] asserting that LSSS with t -uniformity can be obtained from linear codes with distance $t + 1$.

Lemma 4. *Let \mathbf{G} be the generator matrix of an ℓ -folded linear code of length m , dimension k and minimum distance d . Assume that $\mathbf{G} = (\mathbf{I}_k, \mathbf{M})$ where \mathbf{I}_k is the $k \times k$ identity matrix (systematic form of the code). Then the scheme define in Figure 3 is an ℓ -folded $(m, d - 1, m, k)$ -LSSS with uniformity.*

If the code is linear-time encodable, then the LSSS obtained has linear-time complexity.

Proof. According to Remark 2, showing that the map $\psi_A : \mathbf{c} \rightarrow (\mathbf{c} \cdot \mathbf{G}^\top, \mathbf{c}_A)$ is surjective over $\mathbb{F}^k \times (\mathbb{F}^\ell)^{d-1}$ for any $A \subseteq [m]$ of size $d - 1$ is enough to prove that $(\text{Sh}_1, \text{Rec}_1)$ (see Figure 3) has $d - 1$ uniformity. Clearly \mathbf{G} (and then \mathbf{G}^\top) has rank k (over \mathbb{F}) and the map $\mathbf{c} \rightarrow \mathbf{c} \cdot \mathbf{G}^\top$ is surjective. Moreover since \mathbf{G} generates a code of distance d , we can remove any $d - 1$ rows of \mathbf{G} (*i.e.* $d - 1$ columns from \mathbf{G}^\top) and the punctured matrix still has rank k (as any two distinct codewords differ in at least d coordinates). This means that for any \mathbf{m} we can solve in \mathbf{x}

<p>Input: $\mathbf{m} \in \mathbb{F}^k$</p> <p>$\text{Sh}_1(\mathbf{m})$:</p> <p style="padding-left: 20px;">$\mathbf{x}' \leftarrow (\mathbb{F}^\ell)^{m-k}$</p> <p style="padding-left: 20px;">Compute $\mathbf{x}'' = \mathbf{m} - \mathbf{x}' \cdot \mathbf{M}^\top$</p> <p style="padding-left: 20px;">Output $\mathbf{x} = (\mathbf{x}'', \mathbf{x}')$</p>	<p>Input: $\mathbf{c} \in (\mathbb{F}^\ell)^m$</p> <p>$\text{Rec}_1(\mathbf{c})$:</p> <p style="padding-left: 20px;">Compute $\mathbf{m} = \mathbf{c} \cdot \mathbf{G}^\top$</p> <p style="padding-left: 20px;">Output \mathbf{m}</p>
--	---

Fig. 3. Linear-time and constant-rate LSSS

the linear system $\mathbf{x} \cdot \mathbf{G}^\top = \mathbf{m}$ even when $d - 1$ components of \mathbf{x} are fixed. This trivially implies that also the map ψ_A is surjective and concludes the proof of the uniformity property.

Finally, it follows directly from Tellegen's principle (See Appendix A) that if the underlying code is linear-time encodable, then both the algorithms Sh_1 and Rec_1 are linear-time. \square

Instantiating Lemma 4 with ad-hoc linear-time encodable codes provides us with LSSS with linear-time reconstruction algorithms.

Corollary 2 (Linear-Time and Constant-Rate LSSS). *For any real number $\delta \in (0, 1)$ and large enough k there exist a positive integer ℓ and a (m, k) -coding scheme over \mathbb{F} such that the scheme is an ℓ -folded linear-time LSSS δm uniformity. Moreover, $m = \Theta(k)$.*

Proof. Given δ and k , let \mathbf{M} be the generator matrix of the code of Lemma 3, then the matrix $\mathbf{G} = (\mathbf{I}_k, \mathbf{M})$ defines a ℓ -folded linear code of dimension k , length $m + k$ and distance at least $\delta m + 1$. The Corollary follows from Lemma 4. \square

3.2 Construction 1

Finally, we are ready to give the details of Construction 1 and its security proof. All the schemes in the following are defined over the finite field \mathbb{F} and are 1-folded if it is not explicitly stated otherwise. Consider the following building blocks:

- Let $(\text{Enc}_{\text{amd}}, \text{Dec}_{\text{amd}})$ be a (k', k) -AMD code with error ϵ ;
- Let $(\text{Sh}_1, \text{Rec}_1)$ be an ℓ -folded (m, t, m, k') -LSSS with privacy;
- Finally let $(\text{Enc}_{\text{td}}, \text{Dec}_{\text{td}})$ be an (ℓ', ℓ) -TD codes with respect to the family \mathcal{F} and with error α .

The new coding scheme $(\text{ENC}_1, \text{DEC}_1)$ is defined in Figure 4.

We indicate with \mathcal{F}^+ the set of tampering functions $f : (\mathbb{F}^{\ell'})^m \rightarrow (\mathbb{F}^{\ell'})^m$ in $\mathcal{F}_{\ell', m}^q$ such each f_i is a function from $\mathcal{F} \cup \mathcal{F}_{\text{const}} \cup \{\text{id}\}$. That is, each block \mathbf{c}_i of the encoding is modified by the adversary using a function $f_i : \mathbb{F}^{\ell'} \rightarrow \mathbb{F}^{\ell'}$, which can be any function from $\mathcal{F} \cup \mathcal{F}_{\text{const}} \cup \{\text{id}\}$ provided that it doesn't depend on the others blocks of the encoding.

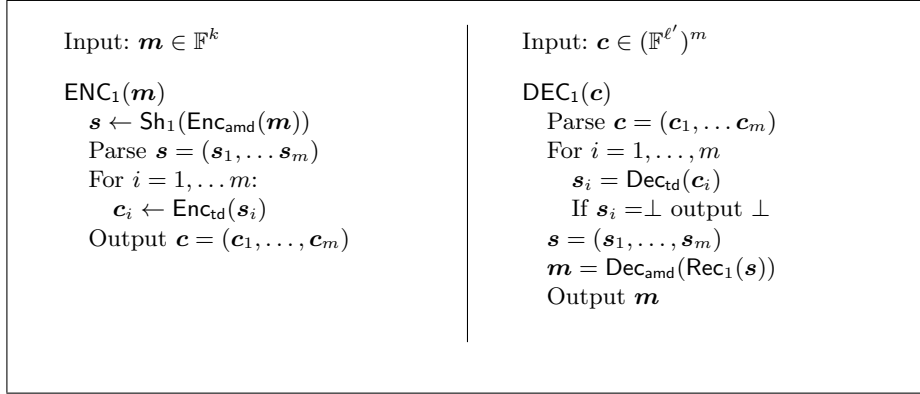


Fig. 4. Construction 1

Theorem 1. *If $t > \frac{m}{2}$, then the coding scheme $(\text{ENC}_1, \text{DEC}_1)$ defined in Figure 4 is an ℓ' -folded (m, k) -NM codes with respect to the family \mathcal{F}^+ with error less or equal to $\max\{\epsilon, \alpha^{2t-m}\}$.*

Moreover, if ρ is the rate of $(\text{Enc}_{\text{amd}}, \text{Dec}_{\text{amd}})$ and ρ' is the rate of $(\text{Sh}_1, \text{Rec}_1)$, then the rate of $(\text{ENC}_1, \text{DEC}_1)$ is $\rho\rho' \frac{\ell}{\ell'}$.

Proof. The correctness of the scheme $(\text{ENC}_1, \text{DEC}_1)$ (i.e. $\Pr[\text{DEC}_1(\text{ENC}_1(\mathbf{m})) = \mathbf{m}] = 1$ for any $\mathbf{m} \in \mathbb{F}^k$) and the statement about the rate are easy to verify and follow directly from the construction (Figure 4).

Fix $f = (f_1, f_2, \dots, f_m) \in \mathcal{F}^+$, to prove the non-malleability property, we have to define D_f as in Definition 2 and bound the error $\text{SD}(\text{Real}_f^{\mathbf{m}}, \text{Ideal}_f^{\mathbf{m}})$ for any $\mathbf{m} \in \mathbb{F}^k$. Let $\mathbf{c} = (\mathbf{c}_1, \dots, \mathbf{c}_m) = \text{ENC}_1(\mathbf{m})$ and $\mathbf{s} = (\mathbf{s}_1, \dots, \mathbf{s}_m) = \text{Sh}_1(\text{Enc}_{\text{amd}}(\mathbf{m}))$. Notice that a valid encoding in the new scheme is a vector $\mathbf{c} = (\mathbf{c}_1, \dots, \mathbf{c}_m)$ of m blocks each of which is an encoding done by the constant-size tamper-detection code $(\text{Enc}_{\text{td}}, \text{Dec}_{\text{td}})$. Each block is independently modified by the function $f_i : \mathbb{F}^{\ell'} \rightarrow \mathbb{F}^{\ell'}$ (with $f_i \in \mathcal{F} \cup \mathcal{F}_{\text{const}} \cup \{id\}$). If $f_i \neq id$, we say that the i -th block is tampered. Since $(\text{Enc}_{\text{td}}, \text{Dec}_{\text{td}})$ is an TD code, for any block such that $f_i \in \mathcal{F}$ we know that the outputs of $\text{Dec}_{\text{td}}(f_i(\mathbf{c}_i))$ is \perp with probability greater or equal to $1 - \alpha$. Using this and the t -privacy property, in the following we will show that we can have enough information on the output of $\text{DEC}_1(f(\text{ENC}_1(\mathbf{m})))$ only looking at how many blocks have been modified by function not in \mathcal{F} . More precisely, define the following sets: $I \subseteq [m]$ is the set of indices i such that f_i is the identity function, $C \subseteq [m]$ is the set of indices i such that f_i is a constant function on $\mathbb{F}^{\ell'}$ and $J = [m] \setminus (I \cup C) = (I \cup C)^c$. Consider now the following cases:

- 1) Suppose that many blocks are tampered using constant functions (i.e. $|C| \geq m - t$), then t -privacy determines the distribution of the blocks not touched by a constant function, while the other blocks are fixed to a constant by the function f_i . Hence, if we define D_f as

- sample $\mathbf{r} = (\mathbf{r}_1, \dots, \mathbf{r}_m)$ accordingly to the distribution of $\text{Sh}_1(\mathbf{0})$ and output the result of $\text{DEC}_1(f(\mathbf{r}))$

then because of the t -privacy we have that $\text{SD}(\text{Real}_f^m, \text{Ideal}_f^m) = 0$.

2) Otherwise we can assume that few blocks are tampered by constant functions (*i.e.* $|J| + |I| > t$) and we consider two sub-cases.

2.a) Suppose that few blocks are tampered (*i.e.* $|I| \geq m - t$) and look at what happens during the execution of DEC_1 on input $f(\mathbf{c})$. If there exists $i \in I^c$ such that $\text{Dec}_{\text{td}}(f_i(\mathbf{c}_i)) = \perp$, then the the entire decoding outputs \perp . Otherwise, we have exactly the situation described by Lemma 1: in the decoding phase the algorithm Rec_1 is applied to a share vector $\tilde{\mathbf{s}}$ where at most t shares have been modified respect to the original share vector \mathbf{s} . It follows that $\text{Rec}_1(\tilde{\mathbf{s}}) = \text{Enc}_{\text{amd}}(\mathbf{m}) + \Delta_f$ where the distribution of the vector Δ_f is independent of \mathbf{m} . In particular, if $\Delta_f = \mathbf{0}$, then the decoding $\text{DEC}_1(f(\mathbf{c}))$ outputs the original message \mathbf{m} , else it outputs \perp with probability grater than or equal to $1 - \epsilon$ because of the application of the AMD decoding algorithm Dec_{amd} to $\text{Enc}_{\text{amd}}(\mathbf{m}) + \Delta_f$.

Thus, in this case we define D_f by the following steps:

- sample $\mathbf{r} = (\mathbf{r}_1, \dots, \mathbf{r}_m)$ accordingly to the distribution of $\text{Sh}_1(\mathbf{0})$. If there exists $i \in I^c$ such that $\text{Dec}_{\text{td}}(f_i(\text{Enc}_{\text{td}}(\mathbf{r}_i))) = \perp$, then output \perp . Otherwise continue with the next step;
- compute the vector $\mathbf{e} = (e_1, \dots, e_m)$ such that $e_i = \text{Dec}_{\text{td}}(f_i(\text{Enc}_{\text{td}}(\mathbf{r}_i))) - \text{Enc}_{\text{td}}(\mathbf{r}_i)$. If $\text{Rec}_1(\mathbf{e}) = \mathbf{0}$, D_f outputs *same*; otherwise it outputs \perp .

Because of the t -privacy, the probability that there exists $i \in I^c$ such that $\text{Dec}_{\text{td}}(f_i(\mathbf{c}_i)) = \perp$ is equal to the probability that there exists $i \in I^c$ such that $\text{Dec}_{\text{td}}(f_i(\text{Enc}_T(\mathbf{r}_i))) = \perp$. Moreover, the vector $\text{Rec}_1(\mathbf{e})$ has the same distribution as Δ_f . Finally, we have that

$$\text{SD}(\text{Real}_f^m, \text{Ideal}_f^m) = \Pr[\text{Dec}_{\text{amd}}(\text{Enc}_{\text{amd}}(\mathbf{m}) + \Delta_f) \neq \perp] \leq \epsilon$$

2.b) Else we can use the assumption on t and m and say that more than $2t - m$ blocks are modified by functions in \mathcal{F} . That is, $|J| > t - m + t = 2t - m > 0$. Independently for all these blocks, the tamper-detection code outputs a message different from \perp with probability less than or equal to α . Thus, $\text{DEC}_1(f(\mathbf{c})) = \perp$ with probability less than or equal to α^{2t-m} . Therefore, in this last case we define D_f to output \perp and we have that

$$\begin{aligned} \text{SD}(\text{Real}_f^m, \text{Ideal}_f^m) &= \Pr[\text{Real}_f^m \neq \perp] \\ &\leq \Pr[\text{Dec}_{\text{td}}(f_i(\mathbf{c}_i)) \neq \perp \quad \forall i \in J] \\ &\leq \alpha^{2t-m} \end{aligned}$$

□

We are now ready to state the first of the results about linear-time NM codes that we present in this paper:

Theorem 2 (Linear-Time and Constant-Rate NM codes). *If for infinitely many integer b , there exists an (b', b) -TD code with respect of a family \mathcal{F} and with constant error α , then there exist a positive integer ℓ' such that the following holds. For any large enough integer k there exists an ℓ' -folded (m, k) -NM code $(\text{ENC}_1, \text{DEC}_1)$ with respect of the family \mathcal{F}^+ and $m = \Theta(k)$. Furthermore, the NM code has error negligible in k and linear-time computational complexity.*

Proof. Given k , instantiate Construction 1 (Figure 4) with the (k, k') -AMD code given by Corollary 1 and with the ℓ -folded $(m, \delta m, m, k')$ -LSSS given by Corollary 2 (with $\delta > 1/2$). Notice that $k' = \Theta(k)$ and $m = \Theta(k') = \Theta(k)$ and ℓ is constant respect to k . Finally, use the first TD code from the family stated in the thesis such that the input length is at least ℓ to complete the instantiation. Let ℓ' be the output length of the TD code used. Notice that also ℓ' is constant respect to k . It follows from Theorem 1 that the obtained scheme is non-malleable with respect to \mathcal{F}^+ and has constant rate. Moreover, the error $\epsilon + \alpha^{2\delta m - m} = q^{-\Theta(k)} + \alpha^{(2\delta - 1)\Theta(k)}$ is negligible in k . Finally, since ℓ, ℓ' are constant and the AMD code and the LSSS are both linear-time, the computational complexity of the algorithms $\text{ENC}_1, \text{DEC}_1$ is $O(k)$. \square

In [CG14b] an infinite family of TD code with respect the family \mathcal{F} of bit-wise tampering functions that are neither the identity nor constant functions is given. Each code in the family has an error less or equal to $2/3$.

Lemma 5 (Lemma 3.5 in [CG14b]). *For any $\alpha > 0$ and any large enough ℓ (i.e. $\ell \geq \ell(\alpha) = O(\log^2(1/\alpha)/\alpha)$), there exists an (ℓ, ℓ') -TD code respect to the family $\mathcal{F} = \mathcal{F}_{1,n}^2 \setminus (\mathcal{F}_{const} \cup \{id\})$ with error $2/3$.*

The previous lemma together with the results we show in the previous section (Corollaries 2 and 1) can be used to instantiate our Construction 1 and to obtain the following result in bit-wise independent tampering model.

Corollary 3 (Binary Case for Construction 1). *For any large enough integer k , there exists a linear-time binary (N, k) -NM code with respect of the family $\mathcal{F}_{1,N}^2$ and with error negligible in k . Furthermore $N = \Theta(k)$.*

4 Optimal-Rate and Linear-Time NM Codes

In this section, we will construct a linear-time non-malleable code with rate approaching 1 (Construction 2).

4.1 Building Blocks for Construction 2

Before showing our second main result (Construction 2), we present the required building blocks.

Independence Generators and Secret-Sharing Schemes

In order to achieve linear-time and optimal-rate NM codes, we will employ linear-time (n, t, n, k) -secret-sharing schemes again, however we will need stronger assumptions regarding the rate and the privacy property of the used scheme. Namely, besides linear-time complexity, we require that the rate is not merely constant but that it approaches 1, *i.e.*, length of a full share-vector divided by the length of the secret tends to 1 when the n tends to infinity. By general bounds on secret sharing, this implies that the privacy parameter t is sub-linear in the number of players n and that reconstruction is essentially by the full player set only. But that is still fine for our purposes here (as long as privacy is nonconstant). Moreover, we note that we do not require linearity of the scheme either. Besides, we require that any t shares are uniformly and independently distributed over the share-space (t -uniformity). Below we show how to construct the schemes required here by combining results on t -wise independence generators and constant-rate secret sharing.

A t -wise independence generator is a deterministic algorithm that expands a short random seed in a longer t -uniform vector. More precisely:

Definition 6 (t -wise Independence Generator, [Gol99]). *Let k, k' and t be positive integers. A function $\text{Gen} : \mathbb{F}^{k'} \rightarrow \mathbb{F}^k$ is a t -wise independence generator if the following holds. For each uniform random variable X over $\mathbb{F}^{k'}$ (called the seed), $\text{Gen}(X)$ is t -wise uniform over \mathbb{F}^k .*

We will now provide a t -wise independence generator with seed-length and independence sub-linear in the output length. The construction combines results of Christiani and Pagh [CP14] and Siegel [Sie04]. Note that the parameter regime we are interested in here differs from that in [CP14].

Definition 7 (Unique Neighbour Expander Graph). *Let $\Gamma = (L, R, E)$ be a finite undirected bipartite graph, with left-vertex set L , right-vertex set R and edge set E . Let n, m, d, e be positive integers. Then Γ is an (n, m, d, e) -unique neighbour expander if the following holds. First, $|L| = n$ and $|R| = m$ and each vertex $v \in L$ has degree d . Second, for each set $S \subseteq L$ of size at most e , there exists a vertex $v \in R$ that has a unique neighbour in S .*

Siegel [Sie04] showed how such an expander can be used to extend the output length of an independence generator at a constant factor loss in the independence. Note that Precisely:

Lemma 6 (Lemma 2.6, Corollary 2.11 in [Sie04]). *Let $\Gamma = (L, R, E)$ be a (n, m, d, e) -unique neighbour expander. Then there exists a \mathbb{F} -linear function $\text{Expand}_\Gamma : \mathbb{F}^m \rightarrow \mathbb{F}^n$ such that the following holds: if X is a (de) -wise uniform random variable over \mathbb{F}^m , then $\text{Expand}_\Gamma(X)$ is an e -wise uniform random variable on \mathbb{F}^n . Moreover, Expand_Γ has computational complexity $O(n)$.*

Proof. (Sketch) Given Γ as in the lemma, the function $\text{Expand}_\Gamma : \mathbb{F}^m \rightarrow \mathbb{F}^n$, $(\mathbf{x}_1, \dots, \mathbf{x}_m) \mapsto (\mathbf{y}_1, \dots, \mathbf{y}_n)$ is defined by

$$\mathbf{y}_i = \sum_{j \in \Gamma(i)} \mathbf{x}_j$$

where $\Gamma(i) \subseteq R$ indicates the neighbours of $i \in L$.

Assume that $A = \{a_1, \dots, a_e\}$ is a subset of $[n]$ of size e . Since Γ be a (n, m, d, e) -unique neighbour expander, there is $v_1 \in R$ that has a unique neighbour in A , wlog we can assume that this neighbour is a_1 . Now we consider $A \setminus \{a_1\}$ and applying the definition again we obtain $v_2 \in R$ that has a unique neighbour a_2 in $A \setminus \{a_1\}$. Continuing in this way we can prove that for any $i \in [e]$ there is $v_i \in R$ such that $v_i \in \Gamma(a_i)$ and $v_i \notin \Gamma(a_j)$ for all $j \geq i + 1$.

Now let $X = (X_1, \dots, X_m)$ be a (de) -wise uniform random variable over \mathbb{F}^m and $Y = (Y_1, \dots, Y_n) = \text{Expand}_\Gamma(X)$. Consider Y_A , because of the previous observation, we have that Y_{a_i} is independent of $(Y_{a_{i+1}}, \dots, Y_{a_e})$ for any $i = 1, \dots, e - 1$. Indeed, $Y_{a_i} = X_{v_i} + Z_i$ for some uniform random variable Z_i and $(Y_{a_{i+1}}, \dots, Y_{a_e})$ is trivially independent of X_{v_i} because $v_i \notin \Gamma(a_j)$ for all $j \geq i + 1$. It follows by induction that Y_{a_1}, \dots, Y_{a_e} are independent. Therefore, Y_A has the uniform distribution on \mathbb{F}^e . \square

In order to be able to use this lemma iteratively, as we will do shortly, an explicit finite family $\{\Gamma_i\}_i$ is required such that $n_i = m_{i+1}$ and $e_i = de_{i+1}$ for all indices i . Christiani and Pagh [CP14] observed that a construction of Capalbo et al. (Theorem 7.1 in [CRVW02]) in fact has this property.

Lemma 7 (Lemma 3 in [CP14]). *For each positive integer c , there are positive integers d and m' and a real number α such that the following holds. For any integer $m \geq m'$ there exists a (cm, m, d, e) -unique neighbour expander Γ with $e \geq \alpha m/d$. The construction of Γ is explicit, i.e. Γ can be constructed in time $\text{poly}(m)$.*

These results give immediate rise to the t -wise independence generator, which will be used later on in the construction of the secret-sharing scheme we require. Note that the generator we construct here is \mathbb{F} -linear.

Lemma 8. *For each real number $\epsilon \in (0, 1)$, there exists a real number $\delta \in (0, \epsilon)$ such that the following holds. For any sufficiently large integer k there exists an explicit t -wise independence generator $\text{Gen} : \mathbb{F}^{k'} \rightarrow \mathbb{F}^k$, where $t = \Omega(k^{1-\epsilon})$ and $k' = \Theta(k^{1-\delta})$. Moreover, Gen is a \mathbb{F} -linear function and has computational complexity $O(k)$.*

Proof. For concreteness, set $c = 2$ in Lemma 7 and let d, m' and α be as given in that lemma. Let ℓ and t be positive integers with $\ell \geq 1$ and $t \geq \max\{1/\alpha, m'\}$ and define $m_i := 2^i d^\ell t$ for any integer $i \geq 0$; notice that $m_i = 2m_{i+1}$. Since $m_i \geq t \geq m'$ for any $i \geq 0$, Lemma 7 implies that there exists an explicit family $\{\Gamma_i\}_{i \geq 0}$ where each Γ_i is a $(2m_i, m_i, d, e_i)$ -unique neighbour expander with

$e_i \geq \alpha m_i/d = \alpha 2^i d^{\ell-1} t$. For any $0 \leq i \leq \ell-1$ define $e'_i := d^{\ell-1-i} \lfloor \alpha t \rfloor$ and notice that $e'_i \geq 1$ and $e'_i = d e'_{i+1}$; moreover, since $e'_i \leq \alpha d^{\ell-1-i} t \leq \alpha 2^i d^{\ell-1} t \leq e_i$, the graph Γ_i is also a $(2m_i, m_i, d, e'_i)$ -unique neighbour expander. Therefore the family $\{\Gamma_i\}_{i=0, \dots, \ell-1}$ has the required parameters.

Now start with a $d e'_0$ -wise uniform random variable on $\mathbb{F}^{m_0} = \mathbb{F}^{d^{\ell} t}$ (e.g. taking the uniform random variable on \mathbb{F}^{m_0}) and apply the map Expand_{Γ_i} from Lemma 6 for all $0 \leq i \leq \ell-1$. In this way we obtain a $\lfloor \alpha t \rfloor$ -wise independence generator $\text{Gen} : \mathbb{F}^{d^{\ell} t} \rightarrow \mathbb{F}^{(2d)^{\ell} t}$ with computational complexity $\sum_{i=0}^{\ell-1} O(2m_i) = O((2d)^{\ell} t)$.

Finally, given ϵ and k as in the statement of the lemma, we choose $t = \lceil k^{1-\epsilon} \rceil$ and $\ell = \lceil \epsilon \log_{2d} k \rceil$. Notice that for k large enough $t \geq \max\{1/\alpha, m'\}$, $\ell \geq 1$ and moreover, the output length of the generator satisfies $(2d)^{\ell} t \geq (2d)^{\epsilon \log_{2d} k} k^{1-\epsilon} = k$ and $(2d)^{\ell} < (2d)(2d)^{\epsilon \log_{2d} k} (k^{1-\epsilon} + 1) = 2d(k + k^{\epsilon})$. Therefore, after truncating if necessary the original output of Gen we obtain a t -wise independence generator of output of length k and computational complexity $O(k)$. Write $z = \log_{2d} d$. The seed length k' satisfies $k' = d^{\ell} t = (2d)^{z \ell} t < (2d)^{z(1+\epsilon \log_{2d} k)} (1 + k^{1-\epsilon}) = (2d)^z k^{z \epsilon} (1 + k^{1-\epsilon})$, which is of the order $k^{1-(1-z)\epsilon}$. Moreover, $k' = d^{\ell} t = (2d)^{z \ell} t \geq (2d)^{z \epsilon \log_{2d} k} k^{1-\epsilon} = k^{z \epsilon} k^{1-\epsilon}$. Thus, choosing $\delta = (1-z)\epsilon$ concludes the proof. \square

We show now how to use the t -wise independence generator of Lemma 8 to build a linear-time secret-sharing scheme with t' -uniformity, $t' = \Theta(t)$ and rate $1 - o(1)$. The high-level idea is simple, to share a secret $\mathbf{m} \in \mathbb{F}^k$ we do the following. First, we mask \mathbf{m} using $\text{Gen}(\mathbf{s})$ where \mathbf{s} is a uniformly random seed for Gen . Then, we share the seed \mathbf{s} with a constant-rate sharing scheme. The final share vector is defined by the concatenation of $\mathbf{m} + \text{Gen}(\mathbf{s})$ and the share vector of \mathbf{s} .

Lemma 9 (Linear-Time and Optimal-Rate LSSS). *For any real number $\epsilon \in (0, 1)$ and any large enough k , there exists a linear-time (n, t, n, k) -LSSS with uniformity such that $t = \Omega(k^{1-\epsilon})$ and $n = k + o(k)$.*

Proof. Given $\epsilon \in (0, 1)$ and k large enough, by Lemma 8 there exists a t -wise independence generator $\text{Gen} : \mathbb{F}^{k'} \rightarrow \mathbb{F}^k$ with $t = \Omega(k^{1-\epsilon})$ and $k' = \Theta(k^{1-\delta})$ ($\delta \leq \epsilon$). Let $(\text{Sh}_1, \text{Rec}_1)$ be the (m, t', m, k') -LSSS from Corollary 2. Notice⁹ that $m = \Theta(k')$ and that the scheme is t' -uniform with $t' = \Theta(k')$. Consider the scheme in Figure 5 and define $s = \min\{t, t'\}$. It is easy to verify that $(\text{Sh}_2, \text{Rec}_2)$ is a linear-time (n, s, n, k) -LSSS with uniformity. Moreover, $s = \Omega(k^{1-\epsilon})$ and $n = k + m = n + O(k^{1-\delta})$. \square

Compressors

We introduce a novel primitive, a *compressor*. Suppose we are given a vector whose coordinates are t -wise independent random variables. A compressor is a

⁹ The family of LSSSs from Corollary 2 is ℓ folded, where ℓ is a constant respect to k' . Thus, the scheme $(\text{Sh}_1, \text{Rec}_1)$ can be “unfolded” and still it remains a constant-rate scheme.

Sh₂(m): Sample $\mathbf{s} \leftarrow \mathbb{F}^{k'}$ Output $\mathbf{c} = (\mathbf{m} + \text{Gen}(\mathbf{s}), \text{Sh}_1(\mathbf{s}))$	Rec₂(c): Parse $\mathbf{c} = (\mathbf{c}_1, \mathbf{c}_2)$ Compute $\mathbf{s} = \text{Rec}_1(\mathbf{c}_2)$ If $\mathbf{s} = \perp$, then output \perp Otherwise output $\mathbf{c}_1 - \text{Gen}(\mathbf{s})$
--	--

Fig. 5. Linear-time and optimal-rate LSSS

deterministic function that, when applying it to the given vector, results in a shorter vector with nontrivial entropy, assuming that the original vector contains at least t coordinates with nontrivial entropy¹⁰.

Definition 8 (Compressor). *Let t, n, n' be positive integers and r a positive real number. A function $\text{Com} : \mathbb{F}^n \rightarrow \mathbb{F}^{n'}$ is a (t, r) -compressor if the following holds. Suppose that $X = (X_1, \dots, X_n)$ is a t -wise independent random variable on \mathbb{F}^n such that there is a set $A \subseteq [n]$ of cardinality t and a real number $c > 0$ for which $H_\infty(X_i) \geq c$ for all $i \in A$. Then $H_\infty(\text{Com}(X)) \geq rc$.*

This primitive is used in the security proof of Construction 2 to handle the case of a component-wise tampering function that has many non-constant components. More precisely, we will use the following fact:

Lemma 10. *Let $f = (f_1, \dots, f_n) \in \mathcal{F}_{1,n}^q$ be a function such that least t of the functions $f_i : \mathbb{F} \rightarrow \mathbb{F}$ are non-constant. If $\text{Com} : \mathbb{F}^n \rightarrow \mathbb{F}^{n'}$ is a (t, r) -compressor and X is a t -wise uniform random variable on \mathbb{F}^n , then for any vector $\mathbf{b} \in \mathbb{F}^{n'}$, $\Pr[\text{Com}(f(X)) = \mathbf{b}] \leq \left(\frac{q-1}{q}\right)^t$.*

Proof. By the conditions on f , there is a set $A \subseteq [n]$ of cardinality t such that, for each $i \in A$ it holds that $H_\infty(f_i(X_i)) \geq \log_2(q/(q-1))$. Since X is t -wise independent, it follows by definition of compressor that $H_\infty(\text{Com}(f(X))) \geq r \log_2(q/(q-1))$. \square

The next lemma shows a simple construction of a compressor suitable for our purposes later on.

Lemma 11. *For any real number $\epsilon \in (0, 1)$ and for any large enough positive integer n there exists an (r^2, r) -compressor $\text{Com} : \mathbb{F}^n \rightarrow \mathbb{F}^{n'}$ with $r^2 = \Omega(n^{1-\epsilon})$ and $n' = o(n)$. Moreover Com has computational complexity $O(n)$.*

Proof. Given ϵ , for any $n \geq 1$ define $r = \lceil n^{(1-\epsilon)/2} \rceil$ and $n' = \lfloor n/r \rfloor$. Notice that $n'r \leq n$ and, if n large enough, $r^2 \leq n$. Consider the function $\text{Com} : \mathbb{F}^n \rightarrow \mathbb{F}^{n'}$,

¹⁰ Since we require compressors to be deterministic, generic methods for privacy amplification do not apply here.

$(\mathbf{x}_1, \dots, \mathbf{x}_n) \mapsto (\mathbf{y}_1, \dots, \mathbf{y}_{n'})$ defined by

$$\mathbf{y}_i = \sum_{j=1}^r \mathbf{x}_{(i-1)r+j} \quad \text{for } i = 1, \dots, n'$$

Thus, a vector in the domain is viewed as comprising n' consecutive blocks of r coordinates and, for $i = 1, \dots, n'$, the sum taken over the coordinates in the i -th block gives the i -th coordinate in the image of the vector under Com . We now verify that Com is a (r^2, r) -compressor. Suppose $X = (X_1, \dots, X_n)$ be a r^2 -wise independent random variable on \mathbb{F}^n and suppose $A \subset [n]$ with $|A| = r^2$ satisfies $H_\infty(X_i) \geq c > 0$ for each $i \in A$. Define $(Y_1, \dots, Y_{n'}) = \text{Com}(X)$. By the pigeonhole principle, there exists a $B \subseteq [n']$ with $|B| = r$ such that each Y_i with $i \in B$ is sum of at least one X_i with $i \in A$. This, together with r^2 -independence of X , implies that the corresponding random variable $Y_B = (Y_i)_{i \in B}$ has the properties that $H_\infty(Y_i) \geq c$ for each $i \in B$ and that the Y_i 's are independent. In conclusion, $H_\infty(\text{Com}(X)) \geq H_\infty(Y_B) \geq rc$. By inspection, the computational complexity of Com is $O(n)$. \square

Almost Universal Families

Our Construction 2 that we present later on in Section 4.2 depends in particular on universal hash functions. For our purposes, we require that these functions are linear-time computable and have vanishingly small key- and output-lengths. Hence, the linear uniform family of [DI14] (see Lemma 2) does not apply directly due to its linear key-length. Note that, besides linear-time, the uniform output property of this particular family enables arbitrary output-length. Below we show an easy adaptation of the family from [DI14] suitable for our purposes. It is a μ -almost universal family. But since μ is very small, it is good enough for our purposes.

Definition 9 (Almost Universal Family). *Let $\mu \in (0, 1)$ be a real number and let n, m be positive integers. Suppose \mathcal{H} is a family of functions $h_{\mathbf{k}} : \mathbb{F}^n \rightarrow \mathbb{F}^m$, one for each $\mathbf{k} \in \mathbb{F}^a$. Then \mathcal{H} is μ -almost universal if the following holds. For any pair of distinct $\mathbf{x}, \mathbf{x}' \in \mathbb{F}^n$, we have $|\{\mathbf{k} \in \mathbb{F}^a \mid h_{\mathbf{k}}(\mathbf{x}) = h_{\mathbf{k}}(\mathbf{x}')\}| \leq \mu q^a$.*

Lemma 12. *For any real number $\beta \in (0, 1)$ and any positive integer n , there exists a μ -universal family $\mathcal{H} = \{h_{\mathbf{k}} : \mathbb{F}^n \rightarrow \mathbb{F}^m\}_{\mathbf{k} \in \mathbb{F}^a}$ with $a = o(n)$, $m = o(n)$ and $\mu = \Theta(q^{-n^{(1-\beta)}})$. Moreover, each function $h_{\mathbf{k}}$ has computational complexity $O(n)$.*

Proof. Given $\beta \in (0, 1)$ and $n \geq 1$, define $k = \lfloor n^{1-\beta/2} \rfloor$ and $k' = \lfloor n^{1-\beta} \rfloor$. It is immediate to verify that in Lemma 2 the range dimension ck of the linear uniform family \mathcal{G} may be replaced by $k' \leq k$ and the result still holds. Therefore, we can assume that there exist a positive integer b and μ -almost universal family $\mathcal{G} = \{g_{\mathbf{k}} : \mathbb{F}^k \rightarrow \mathbb{F}^{k'}\}_{\mathbf{k} \in \mathbb{F}^{bk}}$ with $\mu = 1/q^{k'}$. Moreover, $g_{\mathbf{k}}$ has computational complexity $O(k)$. Now define $n' = \lceil n/k \rceil$ and $h_{\mathbf{k}} : \mathbb{F}^n \rightarrow \mathbb{F}^{k'n'}$ as follows:

$$h_{\mathbf{k}}(\mathbf{x}_1, \dots, \mathbf{x}_n) = (g_{\mathbf{k}}(\mathbf{y}_1), \dots, g_{\mathbf{k}}(\mathbf{y}_{n'}))$$

where¹¹ $\mathbf{y}_i = (\mathbf{x}_{(i-1)k+1}, \mathbf{x}_{(i-1)k+2}, \dots, \mathbf{x}_{ik})$ for any $i = 1, 2, \dots, n'$. Define $m = k'n'$ and $a = bk$. Then $0 < m < n^{1-\beta}(n/k + 1)$, which has order $n^{1-\beta/2}$, and $0 < a \leq bn^{1-\beta/2}$. The computational complexity of $h_{\mathbf{k}}$ is $n'O(k) = O(n)$. Finally, for any distinct $\mathbf{x}, \mathbf{x}' \in \mathbb{F}^n$ there is $i \in [n']$ such that $\mathbf{y}_i \neq \mathbf{y}'_i$. Then,

$$|\{\mathbf{k} \in \mathbb{F}^{bk} \mid h_{\mathbf{k}}(\mathbf{x}) = h_{\mathbf{k}}(\mathbf{x}')\}| \leq |\{\mathbf{k} \in \mathbb{F}^{bk} \mid g_{\mathbf{k}}(\mathbf{y}_i) = g_{\mathbf{k}}(\mathbf{y}'_i)\}| \leq \mu q^{bk}$$

□

4.2 Construction 2

Finally, we are ready to give the details of Construction 2 and its security proof. Consider the following ingredients (all the scheme are over the finite field \mathbb{F}):

- Let $(\text{Sh}_2, \text{Rec}_2)$ an (n, t, n, k) -SSS with uniformity;
- Let $\text{Com} : \mathbb{F}^n \rightarrow \mathbb{F}^{n'}$ be a (t, r) -compressor;
- Let $\mathcal{H} = \{h_{\mathbf{k}} : \mathbb{F}^n \rightarrow \mathbb{F}^m\}$ be a μ -almost universal family with key-space \mathbb{F}^a ;
- Let (Enc, Dec) be a (b', b) -NM code with respect to a family \mathcal{F} with error ϵ . We require that $b = a + m + n'$.

Let $N = n + b'$, the new (N, k) -coding scheme $(\text{ENC}_2, \text{DEC}_2)$ is defined in Figure 6.

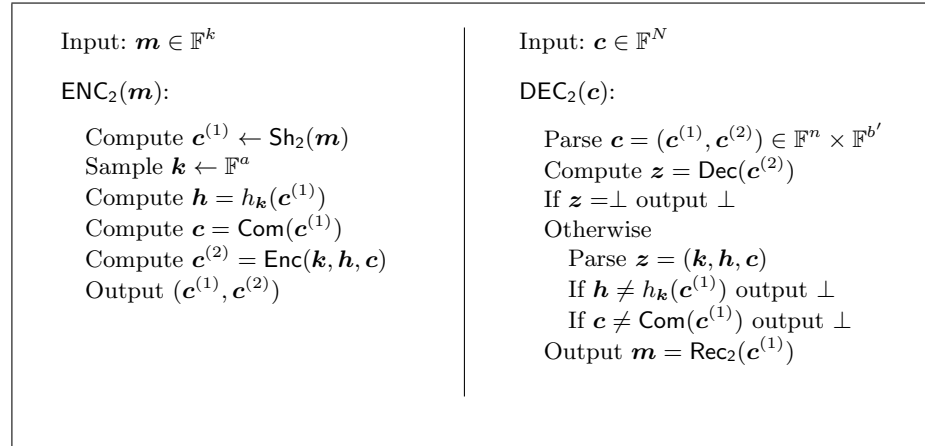


Fig. 6. Construction 2

¹¹ Notice that $n'k \geq n$. If $n'k > n$, the vector $\mathbf{y}_{n'}$ is obtained from the last components of \mathbf{x} padded with zeros.

Theorem 3. *The coding scheme $(\text{ENC}_2, \text{DEC}_2)$ is an (N, k) -non-malleable code with respect to the family $\mathcal{F}_{1,n}^q \times \mathcal{F}$ with error less or equal to*

$$\max \left\{ \left(\frac{q-1}{q} \right)^t + \mu, \left(\frac{q-1}{q} \right)^r \right\} + \epsilon$$

Proof. It is trivial to verify that the scheme $(\text{DEC}_2, \text{ENC}_2)$ is correct. To prove non-malleability, for each tampering function F we have to show a simulator which only depends on F and whose output is very close to $\text{DEC}_2(F(\text{ENC}_2(\mathbf{m})))$ for any given $\mathbf{m} \in \mathbb{F}^k$. More precisely, according to Definition 2 for any $F = (f, g) \in \mathcal{F}_{1,n}^q \times \mathcal{F}$, we have to define a random variable D_F and bound the error $\epsilon' = \text{SD}(\text{Real}_F^{\mathbf{m}}, \text{Ideal}_F^{\mathbf{m}})$ for any $\mathbf{m} \in \mathbb{F}^k$.

Given $\mathbf{m} \in \mathbb{F}^k$ and $F = (f, g) \in \mathcal{F}_{1,n}^q \times \mathcal{F}$, write $f = (f_1, \dots, f_n)$ and $\text{ENC}_2(\mathbf{m}) = (\mathbf{c}^{(1)}, \mathbf{c}^{(2)})$. Notice that the left part of the encoding, $\mathbf{c}^{(1)}$, is tampered by the function $f \in \mathcal{F}_{1,n}^q$, while the right part, $\mathbf{c}^{(2)} = \text{Enc}(\mathbf{k}, \mathbf{h}, \mathbf{c})$, by the function g from \mathcal{F} . Since (Enc, Dec) is a NM-code, there exists the random variable D_g such that $\text{SD}(\text{Real}_g^{\mathbf{z}}, \text{Ideal}_g^{\mathbf{z}}) \leq \epsilon$ for all $\mathbf{z} \in \mathbb{F}^b$. That is, we can simulate the output of decoding the right part, $\text{Real}_g^{\mathbf{z}} = \text{Dec}(g(\mathbf{c}^{(2)}))$, given only the function g . Thus, the first step in order to define D_F is the following: sample \mathbf{z}^* randomly according to D_g . The results of the sampling can be classified in three cases: \perp , *same* or some vector $(\mathbf{k}^*, \mathbf{h}^*, \mathbf{c}^*)$. The value of \mathbf{z}^* determines the output \mathbf{z}' of $\text{Ideal}_g^{\mathbf{z}}$. Then, we proceed in the definition of D_F in a different way for each one of the three aforementioned cases. Moreover in each case we will bound the conditional error

$$\epsilon_{\mathbf{z}'} = \sum_{\mathbf{m}' \in \{\perp\} \cup \mathbb{F}^k} | \Pr[\text{Real}_F^{\mathbf{m}} = \mathbf{m}' | R_{\mathbf{z}'}] - \Pr[\text{Ideal}_F^{\mathbf{m}} = \mathbf{m}' | I_{\mathbf{z}'}] |$$

where $R_{\mathbf{z}'}$ and $I_{\mathbf{z}'}$ are short-cuts for the events $\text{Real}_g^{\mathbf{z}'} = \mathbf{z}'$ and $\text{Ideal}_g^{\mathbf{z}'} = \mathbf{z}'$, respectively.

- 1) Assume that $\mathbf{z}^* = \perp$, then $\mathbf{z}' = \perp$. In this case we define D_F^2 to output \perp and we easily get the following error bound

$$\begin{aligned} \epsilon_{\perp} &= \sum_{\mathbf{m}' \in \{\perp\} \cup \mathbb{F}^k} | \Pr[\text{Real}_F^{\mathbf{m}} = \mathbf{m}' | R_{\perp}] - \Pr[\text{Ideal}_F^{\mathbf{m}} = \mathbf{m}' | I_{\perp}] | \\ &= | \Pr[\text{Real}_F^{\mathbf{m}} = \perp | R_{\perp}] - \Pr[\text{Ideal}_F^{\mathbf{m}} = \perp | I_{\perp}] | = 1 - 1 = 0 \end{aligned}$$

- 2) If $\mathbf{z}^* = \text{same}$, then $\mathbf{z}' = \mathbf{z} = (\mathbf{k}, \mathbf{h}, \mathbf{c})$. Define $I \subseteq [n]$ the set of indices i such that f_i is the identity function on \mathbb{F} . Consider the following two situations.
 - First, assume that many f_i are the identity function (*i.e.* $|I| \geq n - t$). Then the difference $f(\mathbf{c}^{(1)}) - \mathbf{c}^{(1)}$ has distribution independent of \mathbf{m} because $\mathbf{c}^{(1)}$ is t -wise uniform on \mathbb{F}^n . In particular, both the event $f(\mathbf{c}^{(1)}) = \mathbf{c}^{(1)}$ and its complement occur with probability independent of the message \mathbf{m} . If $f(\mathbf{c}^{(1)}) = \mathbf{c}^{(1)}$, then the decoding obviously outputs

the original message \mathbf{m} . Otherwise, we have $f(\mathbf{c}^{(1)}) \neq \mathbf{c}^{(1)}$ and the check $\mathbf{h} = h_{\mathbf{k}}(f(\mathbf{c}^{(1)}))$ fails with probability at least $1 - \mu$. If the check fails, the decoding outputs \perp . Thus, we define D_F in the following way:

- sample $\mathbf{r}_i \leftarrow \mathbb{F}$ for all $i \in I^c$; if $f_i(\mathbf{r}_i) = \mathbf{r}_i$ for all $i \in I^c$ then outputs *same*, otherwise output \perp .

Because the t -uniformity property the event $f_i(\mathbf{r}_i) = \mathbf{r}_i$ for all $i \in I^c$ has the same probability as the event $f(\mathbf{c}^{(1)}) = \mathbf{c}^{(1)}$ and therefore, as a consequence of the check involving the hash function, we can bound the error in the following way.

$$\begin{aligned} \epsilon_{\mathbf{z}} &= \sum_{\mathbf{m}' \in \{\perp\} \cup \mathbb{F}^k} |\Pr[\text{Real}_F^{\mathbf{m}} = \mathbf{m}' \mid R_{\mathbf{z}}] - \Pr[\text{Ideal}_F^{\mathbf{m}} = \mathbf{m}' \mid I_{\mathbf{z}}]| \\ &\leq 2 \cdot \Pr[\text{Real}_F^{\mathbf{m}} \neq \perp \mid R_{\mathbf{z}} \text{ and } f(\mathbf{c}^{(1)}) \neq \mathbf{c}^{(1)}] \\ &\leq 2 \cdot \Pr[h_{\mathbf{k}}(f(\mathbf{c}^{(1)})) = h_{\mathbf{k}}(\mathbf{c}^{(1)}) \mid f(\mathbf{c}^{(1)}) \neq \mathbf{c}^{(1)}] \leq 2\mu \end{aligned}$$

- In the second case, assume that many f_i are not the identity function (*i.e.* $|I| < n - t$). Then, there exists a set $A \subseteq I^c$ of size t , and it follows from the uniformity property that the events $f_i(\mathbf{c}_i^{(1)}) \neq \mathbf{c}_i^{(1)}$ with $i \in A$ are independent and each of them occurs with probability at least $1/q$. Therefore, very likely $f(\mathbf{c}^{(1)}) \neq \mathbf{c}^{(1)}$ and DEC_2 outputs \perp because of the check done using the hash function $h_{\mathbf{k}}$. For this reason, in this case we define D_F to output \perp and we can bound the error as follows.

$$\begin{aligned} \epsilon_{\mathbf{z}} &= \sum_{\mathbf{m}' \in \{\perp\} \cup \mathbb{F}^k} |\Pr[\text{Real}_F^{\mathbf{m}} = \mathbf{m}' \mid R_{\mathbf{z}}] - \Pr[\text{Ideal}_F^{\mathbf{m}} = \mathbf{m}' \mid I_{\mathbf{z}}]| \\ &\leq 2 \cdot \Pr[\text{Real}_F^{\mathbf{m}} \neq \perp \mid R_{\mathbf{z}}] \leq 2 \cdot \Pr[h_{\mathbf{k}}(f(\mathbf{c}^{(1)})) = h_{\mathbf{k}}(\mathbf{c}^{(1)})] \\ &\leq 2 \cdot \Pr[f(\mathbf{c}^{(1)}) = \mathbf{c}^{(1)}] + 2\mu \\ &\leq 2 \left(\frac{q-1}{q} \right)^t + 2\mu \end{aligned}$$

- 3) If $\mathbf{z}^* = (\mathbf{k}^*, \mathbf{h}^*, \mathbf{c}^*)$, then we have that $\mathbf{z}' = \mathbf{z}^*$. Let $C \subseteq [n]$ be the set of all indices i such that f_i is a constant function on \mathbb{F} . Consider the following two situations.

- If many f_i are constant functions (*i.e.* $|C| \geq n - t$), then the t -uniformity implies that the distribution of vector $f(\mathbf{c}^{(1)})$ is independent of \mathbf{m} . In particular, this distribution is known given the tampering function f . Thus, we can define D_F in this way:

- sample $\mathbf{r} \leftarrow \mathbb{F}^n$ and output $\text{DEC}_2(f(\mathbf{r}), \mathbf{z}^*)$.

Because the t -uniformity property the distribution of $f(\mathbf{c}^{(1)})$ is the same of $f(\mathbf{r})$ and therefore

$$\epsilon_{\mathbf{z}^*} = \sum_{\mathbf{m}' \in \{\perp\} \cup \mathbb{F}^k} |\Pr[\text{Real}_F^{\mathbf{m}} = \mathbf{m}' \mid R_{\mathbf{z}^*}] - \Pr[\text{Ideal}_F^{\mathbf{m}} = \mathbf{m}' \mid I_{\mathbf{z}^*}]| = 0$$

- Otherwise more than t components f_i are not constant functions (*i.e.* $|C| < n - t$) and it follows from Lemma 10 that $\text{Com}(f(\mathbf{c}^{(1)}))$ is a random variable with min-entropy at least $r \log_2(q/(q-1))$. Moreover, the distribution of $\text{Com}(f(\mathbf{c}^{(1)}))$ is independent of \mathbf{c}^* . Therefore in this case the probability that the check done using the compressor is satisfied can be bounded by

$$\Pr[\text{Com}(f(\mathbf{c}^{(1)})) = \mathbf{c}^*] \leq \left(\frac{q-1}{q}\right)^r$$

Remember that if the check is not satisfied then, DEC_2 outputs abort. Thus, we can define D_F to output always \perp and we get an error bounded by:

$$\begin{aligned} \epsilon_{\mathbf{z}^*} &= \sum_{\mathbf{m}' \in \{\perp\} \cup \mathbb{F}^k} |\Pr[\text{Real}_F^{\mathbf{m}} = \mathbf{m}' \mid R_{\mathbf{z}^*}] - \Pr[\text{Ideal}_F^{\mathbf{m}} = \mathbf{m}' \mid I_{\mathbf{z}^*}]| \\ &\leq 2 \cdot \Pr[\text{Real}_F^{\mathbf{m}} \neq \perp \mid R_{\mathbf{z}^*}] \leq 2 \cdot \Pr[\text{Com}(f(\mathbf{c}^{(1)})) = \mathbf{c}^*] \\ &\leq 2 \left(\frac{q-1}{q}\right)^r \end{aligned}$$

To conclude the proof notice that the maximum of the bounds we have computed before gives us a bound for the full error ϵ' . Indeed, if $p_{\mathbf{z}'} = \Pr(R_{\mathbf{z}'}) = \Pr[\text{Real}_g^{\mathbf{z}} = \mathbf{z}']$ and $q_{\mathbf{z}'} = \Pr(I_{\mathbf{z}'}) = \Pr[\text{Ideal}_g^{\mathbf{z}} = \mathbf{z}']$ with $\mathbf{z}' \in \{\perp\} \cup \mathbb{F}^b$, then

$$\begin{aligned} \epsilon' &= \frac{1}{2} \sum_{\mathbf{m}' \in \{\perp\} \cup \mathbb{F}^k} |\Pr[\text{Real}_F^{\mathbf{m}} = \mathbf{m}'] - \Pr[\text{Ideal}_F^{\mathbf{m}} = \mathbf{m}']| \\ &= \frac{1}{2} \sum_{\mathbf{m}' \in \{\perp\} \cup \mathbb{F}^k} \left| \sum_{\mathbf{z}' \in \{\perp\} \cup \mathbb{F}^b} \Pr[\text{Real}_F^{\mathbf{m}} = \mathbf{m}' \mid R_{\mathbf{z}'}] \cdot p_{\mathbf{z}'} - \Pr[\text{Ideal}_F^{\mathbf{m}} = \mathbf{m}' \mid I_{\mathbf{z}'}] \cdot q_{\mathbf{z}'} \right| \\ &\leq \frac{1}{2} \sum_{\mathbf{m}'} \sum_{\mathbf{z}'} |\Pr[\text{Real}_F^{\mathbf{m}} = \mathbf{m}' \mid R_{\mathbf{z}'}] - \Pr[\text{Ideal}_F^{\mathbf{m}} = \mathbf{m}' \mid I_{\mathbf{z}'}]| \cdot p_{\mathbf{z}'} + \Pr[\text{Ideal}_F^{\mathbf{m}} = \mathbf{m}' \mid I_{\mathbf{z}'}] \cdot |q_{\mathbf{z}'} - p_{\mathbf{z}'}| \\ &\leq \frac{1}{2} \left(\sum_{\mathbf{z}'} \sum_{\mathbf{m}'} |\Pr[\text{Real}_F^{\mathbf{m}} = \mathbf{m}' \mid R_{\mathbf{z}'}] - \Pr[\text{Ideal}_F^{\mathbf{m}} = \mathbf{m}' \mid I_{\mathbf{z}'}]| \cdot p_{\mathbf{z}'} \right) + \epsilon \\ &\leq \epsilon + \frac{1}{2} \sum_{\mathbf{z}'} \epsilon_{\mathbf{z}'} \cdot p_{\mathbf{z}'} \leq \epsilon + \frac{M}{2} \end{aligned}$$

where $M = \max_{\mathbf{z}'} \{\epsilon'_{\mathbf{z}'}\}$. This concludes the proof. \square

We are now ready to state the main result about linear-time NM codes that we present in this paper:

Theorem 4 (Linear-Time and Optimal-Rate NM codes). *Suppose that there exists real number $\alpha \in (0, 2)$ such that for any positive integer b there exists a (b', b) -NM-code (Enc, Dec) with respect of a family \mathcal{F} , with error $\epsilon(b) = \text{negl}(b)$ (the error is a negligible function of the message length) and $b' = O(b^\alpha)$,*

then the following holds. For any positive integer k large enough, there exists an (N, k) -NM code $(\text{ENC}_2, \text{DEC}_2)$ with respect of the family $\mathcal{F}_{1,n}^q \times \mathcal{F}$ and with error negligible in k . Furthermore $N = k + o(k)$ and, if the computational complexity of (Enc, Dec) is sub-quadratic in b , then $(\text{ENC}_2, \text{DEC}_2)$ is linear-time.

Proof. Instantiate Construction 2 (Figure 6) with the t -uniform sharing scheme from Lemma 9, the compressor from Lemma 11 (with $\epsilon \leq \frac{2}{\alpha} - 1$) and the universal family from Lemma 12 (with $\beta \geq 2 - \frac{2}{\alpha}$). It is easy to check that $b = (a+m)+n' = O(n^{1-\beta/2}) + O(n^{(1+\epsilon)/2})$ and $n = k + o(k)$. Thus, $b' = O(n^{1-\beta/2})^\alpha + O(n^{(1+\epsilon)/2})^\alpha = O(n)$. It follows from Theorem 3 that $(\text{ENC}_2, \text{DEC}_2)$ is (N, k) -NM with respect of the family $\mathcal{F}_{1,n}^q \times \mathcal{F}$ and that $N = n + b' = k + o(k)$. Moreover, since $t, r^2 = \Omega(k^{1-\epsilon})$ and b tends to infinity as k tends to infinity, the error written in Theorem 3 is negligible in k . Finally, since all the building blocks mentioned before are linear-time, then the computational complexity of the new scheme is $O(k) + O(b^\alpha) = O(k)$. \square

From the latter Theorem 4 together with the result shown in Section 3 (Corollary 3), it follows that:

Corollary 4 (Binary Case for Construction 2). *For any large enough k , there exists linear-time binary (N, k) -NM code with respect of the family $\mathcal{F}_{1,N}^2$ and with error negligible in k . Furthermore, $N = k + o(k)$.*

References

- [ADKO15] Divesh Aggarwal, Yevgeniy Dodis, Tomasz Kazana, and Maciej Obremski. Non-malleable reductions and applications. In *Proceedings of the Forty-Seventh Annual ACM on Symposium on Theory of Computing, STOC 2015, Portland, OR, USA, June 14-17, 2015*, pages 459–468, 2015.
- [ADL14] Divesh Aggarwal, Yevgeniy Dodis, and Shachar Lovett. Non-malleable codes from additive combinatorics. In *Proceedings of the 46th Annual ACM Symposium on Theory of Computing, STOC '14*, pages 774–783, New York, NY, USA, 2014. ACM.
- [AGM⁺15a] Shashank Agrawal, Divya Gupta, Hemanta K Maji, Omkant Pandey, and Manoj Prabhakaran. A rate-optimizing compiler for non-malleable codes against bit-wise tampering and permutations. In *Theory of Cryptography*, pages 375–397. Springer, 2015.
- [AGM⁺15b] Shashank Agrawal, Divya Gupta, Hemanta K. Maji, Omkant Pandey, and Manoj Prabhakaran. Explicit non-malleable codes against bit-wise tampering and permutations. In Rosario Gennaro and Matthew Robshaw, editors, *Advances in Cryptology – CRYPTO 2015*, volume 9215 of *Lecture Notes in Computer Science*, pages 538–557. Springer Berlin Heidelberg, 2015.
- [CDD⁺15] Ronald Cramer, Ivan Bjerre Damgård, Nico Döttling, Serge Fehr, and Gabriele Spini. Linear secret sharing schemes from error correcting codes and universal hash functions. In Elisabeth Oswald and Marc Fischlin, editors, *Advances in Cryptology - EUROCRYPT 2015*, volume 9057 of *Lecture Notes in Computer Science*, pages 313–336. Springer Berlin Heidelberg, 2015.

- [CDF⁺08] Ronald Cramer, Yevgeniy Dodis, Serge Fehr, Carles Padró, and Daniel Wichs. Detection of algebraic manipulation with applications to robust secret sharing and fuzzy extractors. In *Advances in Cryptology–EUROCRYPT 2008*, pages 471–488. Springer, 2008.
- [CDTV16] Sandro Coretti, Yevgeniy Dodis, Björn Tackmann, and Daniele Venturi. Non-malleable encryption: Simpler, shorter, stronger. In Eyal Kushilevitz and Tal Malkin, editors, *Theory of Cryptography*, volume 9562 of *Lecture Notes in Computer Science*, pages 306–335. Springer Berlin Heidelberg, 2016.
- [CG14a] Mahdi Cheraghchi and Venkatesan Guruswami. Capacity of non-malleable codes. In *Proceedings of the 5th Conference on Innovations in Theoretical Computer Science*, ITCS '14, pages 155–168, New York, NY, USA, 2014. ACM.
- [CG14b] Mahdi Cheraghchi and Venkatesan Guruswami. Non-malleable coding against bit-wise and split-state tampering. In *Theory of Cryptography*, pages 440–464. Springer, 2014.
- [CGH⁺85] Benny Chor, Oded Goldreich, Johan Hasted, Joel Freidmann, Steven Rudich, and Roman Smolensky. The bit extraction problem or t-resilient functions. In *Foundations of Computer Science, 1985., 26th Annual Symposium on*, pages 396–407. IEEE, 1985.
- [CKM11] SeungGeol Choi, Aggelos Kiayias, and Tal Malkin. Bitr: Built-in tamper resilience. In DongHoon Lee and Xiaoyun Wang, editors, *Advances in Cryptology – ASIACRYPT 2011*, volume 7073 of *Lecture Notes in Computer Science*, pages 740–758. Springer Berlin Heidelberg, 2011.
- [CMTV15] Sandro Coretti, Ueli Maurer, Björn Tackmann, and Daniele Venturi. From single-bit to multi-bit public-key encryption via non-malleable codes. In Yevgeniy Dodis and Jesper Buus Nielsen, editors, *Theory of Cryptography*, volume 9014 of *Lecture Notes in Computer Science*, pages 532–560. Springer Berlin Heidelberg, 2015.
- [CP14] T. Christiani and R. Pagh. Generating k-independent variables in constant time. In *Foundations of Computer Science (FOCS), 2014 IEEE 55th Annual Symposium on*, pages 196–205, 2014.
- [CRVW02] Michael R. Capalbo, Omer Reingold, Salil P. Vadhan, and Avi Wigderson. Randomness conductors and constant-degree lossless expanders. In *Proceedings on 34th Annual ACM Symposium on Theory of Computing, May 19-21, 2002, Montréal, Québec, Canada*, pages 659–668, 2002.
- [CZ14] E. Chattopadhyay and D. Zuckerman. Non-malleable codes against constant split-state tampering. In *Foundations of Computer Science (FOCS), 2014 IEEE 55th Annual Symposium on*, pages 306–315, 2014.
- [DDV10] Francesco Davì, Stefan Dziembowski, and Daniele Venturi. Leakage-resilient storage. In JuanA. Garay and Roberto De Prisco, editors, *Security and Cryptography for Networks*, volume 6280 of *Lecture Notes in Computer Science*, pages 121–137. Springer Berlin Heidelberg, 2010.
- [DI14] Erez Druk and Yuval Ishai. Linear-time encodable codes meeting the gilbert-varshamov bound and their cryptographic applications. In *Proceedings of the 5th conference on Innovations in theoretical computer science*, pages 169–182. ACM, 2014.
- [DKO13] Stefan Dziembowski, Tomasz Kazana, and Maciej Obremski. Non-malleable codes from two-source extractors. In Ran Canetti and JuanA. Garay, editors, *Advances in Cryptology – CRYPTO 2013*, volume 8043 of

- Lecture Notes in Computer Science*, pages 239–257. Springer Berlin Heidelberg, 2013.
- [DPW10] Stefan Dziembowski, Krzysztof Pietrzak, and Daniel Wichs. Non-malleable codes. In *ICS*, pages 434–452, 2010.
- [FMNV14] Sebastian Faust, Pratyay Mukherjee, Jesper Buus Nielsen, and Daniele Venturi. Continuous non-malleable codes. In Yehuda Lindell, editor, *Theory of Cryptography*, volume 8349 of *Lecture Notes in Computer Science*, pages 465–488. Springer Berlin Heidelberg, 2014.
- [FMVW14] Sebastian Faust, Pratyay Mukherjee, Daniele Venturi, and Daniel Wichs. Efficient non-malleable codes and key-derivation for poly-size tampering circuits. In Phong Q. Nguyen and Elisabeth Oswald, editors, *Advances in Cryptology – EUROCRYPT 2014*, volume 8441 of *Lecture Notes in Computer Science*, pages 111–128. Springer Berlin Heidelberg, 2014.
- [Gol99] Oded Goldreich. *Modern cryptography, probabilistic proofs and pseudo-randomness*. Algorithms and combinatorics. Springer, Berlin, New York, 1999.
- [IKOS08] Yuval Ishai, Eyal Kushilevitz, Rafail Ostrovsky, and Amit Sahai. Cryptography with constant computational overhead. In Cynthia Dwork, editor, *STOC*, pages 433–442. ACM, 2008.
- [JW15] Zahra Jafargholi and Daniel Wichs. Tamper detection and continuous non-malleable codes. In Yevgeniy Dodis and Jesper Buus Nielsen, editors, *Theory of Cryptography*, volume 9014 of *Lecture Notes in Computer Science*, pages 451–480. Springer Berlin Heidelberg, 2015.
- [LL12] Feng-Hao Liu and Anna Lysyanskaya. Tamper and leakage resilience in the split-state model. In Reihaneh Safavi-Naini and Ran Canetti, editors, *Advances in Cryptology – CRYPTO 2012*, volume 7417 of *Lecture Notes in Computer Science*, pages 517–532. Springer Berlin Heidelberg, 2012.
- [Sie04] Alan Siegel. On universal classes of extremely random constant-time hash functions. *SIAM Journal on Computing*, 33(3):505–543, 2004.
- [Tel52] B. D. H. Tellegen. A general network theorem, with applications. *Philips Research Reports*, 7:259–269, 1952.

A Appendix

A.1 Tellegen’s Principle

We will briefly discuss a technique known as Tellegen’s principle. Assume that we are given a linear algorithm T computing the function $f(\mathbf{x}) = \mathbf{A} \cdot \mathbf{x}$, where \mathbf{A} is a $m \times n$ matrix over some ring R and \mathbf{x} is a vector from R^n . Then we can transform T into an algorithm T' computing the function $f'(\mathbf{y}) = \mathbf{A}^\top \cdot \mathbf{y}$, where $\mathbf{y} \in R^m$ and \mathbf{A}^\top is the transpose of the matrix \mathbf{A} , which has the same computational complexity as T . We will discuss this transformation for arithmetic circuits. We can decompose a circuit into a sequence of elementary instructions ϕ_i , where each ϕ_i is a linear transformation on all the wires. We can thus write the matrix \mathbf{A} as

$$\mathbf{A} = \phi_n \cdot \phi_{n-1} \cdots \phi_2 \cdot \phi_1.$$

Transposing \mathbf{A} immediately yields

$$\mathbf{A}^\top = \phi_1^\top \cdot \phi_2^\top \cdots \phi_{n-1}^\top \cdot \phi_n^\top.$$

Thus, we only have to consider the effect of transposition to the elementary instructions ϕ_i .

- Instruction ϕ_i multiplies a wire \mathbf{x} with a constant $\alpha \in R$ and writes the output in the same register. In this case $\phi_i^\top = \phi_i$, as the transformation matrix ϕ_i is diagonal and thus symmetric.
- Instruction ϕ_i adds wire \mathbf{y} to wire \mathbf{x} . In this case ϕ_i^\top adds wire \mathbf{x} to wire \mathbf{y} .

These two instructions are sufficient to implement any linear transformation. For instance, to clear an (auxiliary) register, simply multiply it by 0. We summarize this in the following Lemma.

Lemma 13 (Tellegen’s Principle [Tel52]). *Let $\mathsf{T}(\mathbf{x})$ be a linear arithmetic circuit or linear RAM algorithm computing the function $\mathbf{A} \cdot \mathbf{x}$. Then there exists a linear arithmetic circuit $\mathsf{T}'(\mathbf{y})$ that computes the function $\mathbf{A}^\top \cdot \mathbf{y}$ and has the same computational complexity as T .*