Electronic Colloquium on Computational Complexity, Report No. 69 (2014)

Explicit Non-Malleable Codes Resistant to Permutations

Shashank Agrawal[*] Divya Gupta[†] Hemanta K. Maji[‡] Omkant Pandey[§]
Manoj Prabhakaran[¶]

May 4, 2014

Abstract

The notion of non-malleable codes was introduced as a relaxation of standard error-correction and error-detection. Informally, a code is non-malleable if the message contained in a modified codeword is either the original message, or a completely unrelated value.

In the information theoretic setting, although existence of such codes for various rich classes of tampering functions is known, explicit constructions exist only for highly structured family of tampering functions. Prior explicit constructions of non-malleable codes rely on the "compartmentalized" structure of the tampering function, i.e. the codeword is partitioned into *a priori fixed* blocks and each block can *only* be tampered independently. The prominent examples of this model are the family of bit-wise independent tampering functions and the split-state model.

We consider an infinitely large natural class of non-compartmentalized tampering functions. In our model, the tampering function can permute the bits of the encoding and (optionally) perturb them. In the information theoretic setting, we provide an *explicit* and *efficient*, *rate-1* non-malleable code for *multi-bit messages.*

Lack of explicit constructions of non-malleable codes for non-compartmentalized tampering functions severely inhibits their utility in cryptographic protocols. As a motivation for our construction, we show an application of non-malleable codes to cryptographic protocols. In an idealized setting, we show how string commitments can be based on one-bit commitments, if non-malleable codes exist. Further, as an example of a non-trivial use of non-malleable codes in standard cryptographic protocols (not in an idealized model), we show that if explicit non-malleable codes are obtained for a slightly larger class of tampering functions than we currently handle, one can obtain a very simple non-malleable commitment scheme, under somewhat strong assumptions.

Keywords: Non-malleable Codes, Explicit Construction, Information Theoretic, Non-malleable Commitment.

[*] University of Illinois, Urbana-Champaign. sagrawl2@illinois.edu.

[†] University of California, Los Angeles. divyag@cs.ucla.edu.

[‡] University of California, Los Angeles. hemanta.maji@gmail.com.

[§] University of Illinois, Urbana-Champaign. omkant@gmail.com.

[¶] University of Illinois, Urbana-Champaign. mmp@illinois.edu.

ISSN 1433-8092

Contents

1 Introduction 1

1.1 Our Contribution . . . 3

1.2 Prior Work . . . 3

1.3 Technical Overview . . . 4

1.3.1 Basic Construction . . . 5

1.3.2 Main Construction . . . 7

2 Preliminaries 8

2.1 Classes of Tampering Functions . . . 9

2.2 Non-Malleable Codes Security Experiment . . . 9

2.3 Hash Functions . . . 10

3 Building Blocks 11

4 Basic Non-Malleable Encoding Scheme 15

4.1 Proof of Non-Malleability against $\mathcal{S}_N$ . . . 17

4.2 Proof of Theorem 1 . . . 20

5 Rate 1 Non-Malleable Encoding Scheme 20

5.1 Proof of Theorem 2 . . . 22

References 27

A Ensuring Independence 28

B Unpredictability 29

C Proof of Theorem 1 31

D Proof of Theorem 2 34

D.1 Proof of Case 1 . . . 34

D.2 Proof of Case 2 . . . 36

E Application to Non-malleable Commitments 37

F Mathematical Tools 41

G Algebraic Geometric Codes Parameters 44

1 Introduction

Non-Malleable Codes have emerged as an object of fundamental interest, at the intersection of coding theory and cryptography. Informally, a code is non-malleable if the message contained in a codeword that has been tampered with is either the original message, or a completely unrelated value. As a relatively new problem, several basic questions are still open. In particular, *explicit constructions* of non-malleable codes that can withstand expressive families of attack is one of the main challenges in this area. This is the subject of this paper.

But we begin our investigation into non-malleable codes with an enticing question, from an entirely different domain:

Can non-malleable string-commitments be "entirely based" on non-malleable bit-commitments?

To formalize this problem, we consider an idealized model of bit commitments: to commit a bit to Bob, Alice can create a small physical token which has the bit "locked" inside (and later, she can send him a "key" to open the token). This completely hides the bit from Bob until Alice reveals it to him; on the other hand, Alice cannot change the bit inside the token once she has sent it to Bob. Further, this is a non-malleable bit commitment scheme, in that if Bob plays a man-in-the-middle adversary, and wants to send a commitment to Carol, he can can only send the token from Alice as it is, or create a new token himself, independent of the bit committed to by Alice. In fact, further, let us assume that Bob cannot create a token on his own. Then, the only man-in-the-middle attacks possible are to forward Alice's token to Carol, or to send nothing to Carol, and Bob has to decide on an action independent of the bit Alice committed to.

Now, we ask whether, in this model, one can make non-malleable string commitments (relying on no computational assumptions). *This is a question about non-malleable codes in disguise!* Indeed, if we required the commitment protocol to involve just a single round of token transfer, then a commitment protocol is nothing but a non-malleable encoding of a string into bits. If all codewords have the same length, and if Bob receives only one commitment, then the class of attacks we need to protect against is that of *bit-level permutations.*

This application also brings out an important aspect of non-malleable codes: whether they are explicit or not. While there indeed is an efficient randomized construction of non-malleable codes that can resist permutations [FMVW13], it will not be suitable in this case, because neither the sender nor the receiver in a commitment scheme can be trusted to pick the code honestly (Bob could play either role), and non-malleable codes are not guaranteed to stay non-malleable if the description of the code itself can be tampered with.

In this work, we present the first explicit construction of a non-malleable code that can resist bit-level permutations. In fact, we consider a more general class of attacks in which an attacker can first permute the bits of the encoding and then perturb it by passing each bit through some channel of its choice. We allow the adversary the family of all channels (with probabilities which are constant, independent of the security parameter), *except for the two constant channels* (first of which set its output always to 0 and the other which sets it to 1). In return, we obtain a stronger non-malleable guarantee than usual: the attack can result only in a codeword with the same message, or an invalid codeword. (This corresponds to our stronger non-malleability requirement that Bob cannot create his own commitments other than by simply forwarding Alice's commitment.) Also, the probability of generating an invalid codeword is the same for all messages.

Our construction is in two steps: first we build a rate-0 non-malleable code,[1] and then use it in constructing a rate-1 non-malleable code.

Our rate-0 construction is fairly natural, though its analysis turns out to be more complicated. At a high-level the construction combines an "outer code" which has a sufficiently large distance and dual distance (for example, a "packed secret-sharing scheme" based on the Reed-Solomon code) with an "inner code" which is *transparent* to bit-level permutations – namely, a unary code. Each character of the outer code-word is encoded into a block of bits using the inner code. To obtain some level of resistance to mixing bits from different blocks, we keep the bits in the codeword randomly permuted, and also, we ensure a good density of invalid codewords in the code (achieved, specifically, by requiring that the the weight of each codeword is, say, even), so that blindly mixing together bits from different codewords has *some* probability of creating an invalid codeword. A careful combinatorial argument can be used to show that, despite dependencies among the blocks caused by a permutation attack, the probability of having all attacked blocks remaining valid decreases multiplicatively. If the outer code has a large distance, this ensures that the probability of creating a different valid codeword is negligible. However, we need to ensure not only that the attack has negligible chance of modifying one codeword into a different valid codeword, but also that the probability of creating an invalid codeword is (almost) independent of the actual message. Roughly, this is based on the large dual distance of the outer code.

The code above does not have a positive rate (i.e., the size of the codewords is super-linear in the size of the messages). Next, we show that this code can then be boot-strapped into one with rate 1. The idea is to encode the message using a rate-1 code of sufficiently large dual distance to obtain a codeword c, and then use the non-malleable code to encode a (weakly collision-resistant) hash function h and $h(c)$. (The proof of non-malleability of this construction uses specific properties of our rate-0 non-malleable code and does not apply if an arbitrary non-malleable code is used.) Note that, to obtain rate 1, it is important that the description of h itself is sub-linear in the size of c, precluding the use of a 2-universal hash function. Instead we use a Merkle-tree construction applied to a 2-universal hash function family over a domain of $\Theta(\kappa)$-bit long strings.

Finally, we return to the problem of non-malleable commitments. While the motivating problem was posed in an idealized setting, one could ask if the same approach – of encoding a string using a non-malleable code, and then committing each bit individually – would yield a non-malleable string commitment scheme in the standard model. Interestingly, we give a new construction and analysis to show that this is indeed possible, with a somewhat strong assumption (used to implement a simple non-malleable bit-commitment protocol), if explicit non-malleable codes that can tolerate permutations along with *all channels* (including the two constant channels) are constructed. We leave open the problem of constructing non-malleable codes against this class of attacks.

[1] Rate can be defined as $\inf_\kappa \lim_{L\to\infty} \frac{L}{N(L,\kappa)}$, where $N(L,\kappa)$ stands for the number of bits in a codeword in the code for L-bit long messages with security parameter κ (the advantage an adversary can have in the malleability experiment should be negligible in κ).

1.1 Our Contribution

We construct, in the information theoretic setting, explicit non-malleable rate 1 codes against a tampering function family which allows permutation of the code followed by some perturbation of each bit. Our message space is $\{0,1\}^L$ and our codewords lie in $\{0,1\}^N$. The set $\mathcal{S}_N$ represents the set of all permutations over $[N]$. A channel f over $\{0,1\}$ takes as input a bit $b \in \{0,1\}$ and outputs a distribution over $\{0,1\}$. If support of f is $\{0,1\}$,[2] then we say that it is a non-constant channel. For example, $f(b) = 1 \oplus b$ (toggle channel), $f(b) = \mathbf{U}_{\{0,1\}}$ (uniform bit generation channel) and $f(b) = b$ (message-forwarding channel) are special cases of non-constant channels.

Our class of tampering functions is specified by $(\pi, f_1, \ldots, f_N)$ such that $\pi \in \mathcal{S}_N$, and f_is are non-constant channels. Given an encoding $c_{[N]} \equiv c_1 \ldots c_N$, the tampering function produces the following (distribution) of strings: $f_1(c_{\pi^{-1}(1)}) \ldots f_N(c_{\pi^{-1}(N)})$. That is, our tampering functions allows permutation of the codeword (at bit level representation) followed by passing each bit of the outcome through a non-constant channel. This class is represented by $\mathcal{C}_{\{0,1\}} \circ \mathcal{S}_N$. Our main theorem states the following:

Informal Theorem 1. *There exists an explicit non-malleable rate 1 code with efficient encoding and decoding against an adversary who is allowed to permute the output codeword followed by application of non-constant channels.*

Note that the tampering family $\mathcal{C}_{\{0,1\}} \circ \mathcal{S}_N$ is an infinitely large family.[3]

1.2 Prior Work

Prior to the study of non-malleable codes, [CDF$^+$08] considered a related notion of detecting arithmetic manipulations.

Non-malleable Codes. The study of non-malleable codes was formalized and motivated in [DPW10]. They showed existence of a constant rate non-malleable code against the class of all bit-wise independent tampering functions. Existence of rate 1 non-malleable codes against various classes of tampering functions is known. For example, existence of such codes with rate $(1-\alpha)$ was shown against any tampering function family of size $2^{2^{\alpha n}}$; but this scheme has inefficient encoding and decoding [CG14a]. For tampering functions of size $2^{\mathsf{poly}(n)}$, rate 1 codes (with efficient encoding and decoding) exist with overwhelming probability [FMVW13].

On the other hand, explicit constructions of non-malleable codes with good rate have remained elusive, except for some well structured tampering function classes. Recently, an explicit rate 1 code for the class of bit-wise independent tampering function was proposed by [CG14b]. Note that a tampering function in this class tampers each bit independently. For the generalized model, where

[2] Support of f is defined to be the union of supports of $f(0)$ and $f(1)$ distributions.

[3] The class of "simple channels" $\mathcal{C}_{\mathsf{simple},\{0,1\}}$ is the set of channels which contain: 1) $f(b) = b$, 2) $f(b) = 1 \oplus b$, 3) $f(b) = 0$ and 4) $f(b) = 1$. Although each channel in the class $\mathcal{C}_{\{0,1\}}$ can be written as a convex linear combination of channels in $\mathcal{C}_{\mathsf{simple},\{0,1\}}$, it is *not* the case that any tampering function in the class $\mathcal{C}_{\{0,1\}} \circ \mathcal{S}_N$ can be written as a convex linear combination of tampering functions in the class $\mathcal{C}_{\mathsf{simple},\{0,1\}} \circ \mathcal{S}_N$. So, to construct non-malleable codes against the infinite family $\mathcal{C}_{\{0,1\}} \circ \mathcal{S}_N$, it is *not* sufficient to construct a non-malleable code against the finite family $\mathcal{C}_{\mathsf{simple},\{0,1\}} \circ \mathcal{S}_N$.

the codeword is partitioned into separate blocks and each block can be tampered arbitrarily but independently, an encoding scheme was proposed in [CKM11]. In the most general such setting, i.e. there are only two such independent partitions (known as the split-state model), an explicit encoding scheme for bits was proposed by [DKO13]. Recently, in a major break-through result, an explicit scheme (of rate 0) was proposed for arbitrary length messages by [ADL13].

Note that, known explicit construction of codes against particular tampering function classes exploit the "compartmentalized" nature of the family of tampering functions, i.e. the codeword can be *a priori* partitioned into pieces such that the tampering function is applied independently to each partition. For example, bit-wise independent tampering functions act on each bit independently; and in the split-state model, the tampering on each state is independent. The class of functions being studied in this paper is one of the most natural classes of tampering functions without the aforementioned "compartmentalization" property. Achieving non-malleability via explicit codes against non-compartmentalized class of functions turns out to be cryptographically interesting (see Appendix E).

Codes under computational assumptions. The idea of improving the rate of error-correcting codes by considering computationally limited channels stems from the work of Lipton [Lip94]. Restricting the channels to be computationally efficient allows one to use cryptographic assumptions, e.g., Micali et. al. [MPSW05] show how to combine digital signatures with list-decodable codes to go beyond the classical error correction bound for unique decoding. Further constructions in various settings were provided in [OPS07, HO08, GS10, CKO14]. In the setting of non-malleable codes as well, constructions based on computational assumptions have been explored, e.g., in [LL12, FMNV14].

Non-malleable commitments. There is an extensive amount of literature on non-malleable commitments starting from the work of Dolev, Dwork and Naor [DDN91] leading to recent constant-round constructions based on one-way functions [Goy11, LP11]. Our suggested application of non-malleable codes to non-malleable commitments is similar in spirit to the work of Meyers and Shelat [MS09] on the completeness of bit encryption.

1.3 Technical Overview

An obvious starting point to construct a code which is non-malleable against the class of all permutations is a unary encoding scheme, i.e. a message $s \in \{0,1\}^L$ is interpreted as a number $s \in \mathbb{Z}_{2^L}$ and the encoding is the characteristic vector of a size s subset of $[N]$. Note that in this case, we need $N = 2^L$ (the rate is exponentially close to 0); and hence it cannot be efficiently encoded or decoded.

The next logical step is to explore Reed-Solomon based share-packing techniques over a field $\mathbb{F}$ of characteristic 2 such that $|\mathbb{F}| = \Theta(L)$. Suppose there are n elements in this encoding of s. Each of these elements need to be protected against tampering. To achieve this goal, we can attempt to de-randomize the existential results of [FMVW13, CG14a] to construct non-malleable (inner) encodings of the elements. But note that the tampering family size continues to be exponential in n, because each (encoding of an) element in the codeword can possibly receive bits from any position of the original codeword. Therefore, de-randomization of these existential proofs cannot be efficiently performed. Note that efficiency improvements by choosing share-packing techniques with Algebraic Geometric (AG) codes over constant size fields also faces the identical bottleneck.

Our rate 1 encoding scheme is modularly constructed in two steps. First, an efficient non-malleable code is constructed which achieves rate $\Theta(1/L)$. This encoding scheme is an amalgamation of the two ideas mentioned above, i.e. share-packing techniques using algebraic geometric codes and unary encoding scheme as an inner encoding scheme. There are other subtleties which arise but can be resolved using a variant of the unary encoding scheme mentioned above. Finally, we construct a rate 1 encoding scheme using this code to bootstrap. This step employes a Reed-Solomon based (rate 1) share-packing of the message; followed by appending a *small* tag using the non-malleable code constructed previously.

Henceforth, these two encoding schemes shall be referred to as the *basic* and *main* constructions, respectively. We emphasize that our main construction works *only* for the particular basic construction we provide, i.e. we do *not* show a mechanism to convert any basic non-malleable scheme into a rate 1 non-malleable scheme. Our basic construction provides several additional combinatorial properties which make is amenable to rate amplification.

1.3.1 Basic Construction

For ease of presentation, we introduce our construction using Reed-Solomon codes instead of algebraic geometric codes in this section.

Suppose $\mathbb{F}$ is a sufficiently large field of characteristic 2 such that $\{f_{-\ell}, \dots, f_{-1}, f_1, \dots, f_n\} \subseteq \mathbb{F}$. To encode a message $s_{[\ell]} \equiv (s_1, \dots, s_\ell) \in \mathbb{F}^\ell$, pick a random polynomial $p(\cdot)$ of degree $< k$ such that $p(f_{-i}) = s_i$, for all $i \in [\ell]$. The encoding is defined to be $c_{[n]} \equiv (\ \langle 1, p(f_1)\rangle, \dots, \langle n, p(f_n)\rangle\ ) \in ([n] \times \mathbb{F})^n$. This serves as our outer encoding scheme and each element is interpreted in $\mathbb{Z}_{n|\mathbb{F}|}$.

Let $m = 6n\,|\mathbb{F}|$. The inner encoding of a secret $x \in \mathbb{Z}_{n|\mathbb{F}|}$ is the characteristic vector of a random subset of $[m]$ of size $(m/3) + 2x$.

Our final encoding scheme is a concatenation of the outer and the inner encoding schemes. In particular, we have $L = \ell \lg |\mathbb{F}|$, $N = nm = 6n^2\,|\mathbb{F}|$, $k = n/3$ and $\ell = k/2$. Rate of this code is $L/N = \tilde{\Theta}(1/L^2)$.[4] Final encoding of s is represented by: $c_{[N]} \in \{0,1\}^N$.

Suppose $\pi \in \mathcal{S}_N$ be a tampering function. The codeword obtained by applying this tampering function to the codeword $c_{[N]}$ is $\pi(c_{[N]}) \equiv c_{\pi^{-1}[N]} := c_{\pi^{-1}(1)} \dots c_{\pi^{-1}(N)}$.

Intuition of Non-malleability Proof. We consider two special tampering functions to illustrate our proof outline.

Example 1. Consider the tampering function π which identically maps the first $(n-2)m$ bits, i.e. it "preserves" the encodings of the first $(n-2)$ blocks. And it permutes some bits across the encodings of the last two blocks.

[4] If algebraic geometric codes are used, then $L = c\ell$ and $N = 6n^2 2^c$, where $|\mathbb{F}| = 2^c$ and c is any constant $\geqslant 6$. In this case, rate is: $\Theta(1/L)$.

Note that the distance of the Reed-Solomon code used in our setting is $\gg 2$ even after fixing the message s. So, $\pi(c_{[N]})$ is a valid codeword if and only if the last $2m$ bits of $c_{[N]}$ and $\pi(c_{[N]})$ encode identical elements in $([n] \times \mathbb{F})$. Further, the independence of our Reed-Solomon code is $\gg 2$, therefore the distribution of $\langle n-1, p(f_{n-1})\rangle$ and $\langle n, p(f_n)\rangle$ are, respectively, identical to $\langle n-1, \mathbf{U}_{\mathbb{F}}\rangle$ and $\langle n, \mathbf{U}_{\mathbb{F}}\rangle$, irrespective of the value of the message s being encoded.

So, the probability σ of $\pi(c_{[N]})$ being a valid codeword and encoding the identical message is the probability of 0 in the following experiment:

1. Sample $a \sim \langle n-1, \mathbf{U}_{\mathbb{F}}\rangle$ and $b \sim \langle n, \mathbf{U}_{\mathbb{F}}\rangle$.
2. Encode a and b with the inner encoding scheme; the encodings are represented by $\tilde{a}$ and $\tilde{b}$ respectively.
3. Apply the tampering function π (restricted to the last two blocks) to $(\tilde{a}, \tilde{b})$.
4. If the tampered blocks a' and b' encode a and b, respectively, then output 1; otherwise output 0.

Note that σ is independent of the message s; and can be computed efficiently. We emphasize that a' need not be (bit-wise) identical to $\tilde{a}$ and b' need not be (bit-wise) identical to $\tilde{b}$; they need to encode the same message a and b, respectively.

We say that an outer codeword is "dirty," if it receives bits from at least two different blocks after the tampering function is applied. So, in this example, π has exactly two dirty outer codewords. An analogous argument suffice when the number of dirty outer codewords is "small."

Example 2. Now consider a tampering function π which has the following property: Each block receives at least one bit from every block of $c_{[N]}$.

In this case, we leverage the fact that any valid inner code has parity 0; and (on average) any bit of the encoding is 0 and 1 with at least a constant probability. So, if bits from two (or more) different blocks are copied into one block by the tampering function, then we expect that the resulting parity of the tampered block should have parity 1 with constant probability. We expect the following. Since a large number of outer codewords are dirty, with $1 - \mathsf{negl}(n)$ probability at least one block of the tampered codeword $\pi(c_{[N]})$ does not have parity 0. But this argument is not immediate because the parities of bits in the tampered blocks are correlated random variables.

First, note that the probability of any bit in $c_{[N]}$ being 1 is at least 1/3. One can extend this argument (despite correlations among wires) to claim that any subset of $1 \leqslant t \leqslant m$ wires in a block has parity 1 with a constant probability. Intuitively, given this observation, we would like to conclude that the probability that all dirty outer codewords have parity 0 is $\mathsf{negl}(n)$.

Despite correlations, using a careful analysis, we show that there exist (at least) $n/2$ outer blocks whose parity of bits are independently set to 1 with constant probability (and this is tight). This suffices to formally prove our intuitive claim. Thus, in this case, we can output $\sigma = 1$.

A generalization of this argument also works when the number of dirty outer codewords is "large."

Putting Things Together. When the number of dirty codewords is "small," we use the argument used in the first example. On the other hand, if the number of dirty codewords is "large" we use the argument of the second example. By appropriately choosing the parameters of our encoding scheme these cases are exhaustive.

Extension to Channels. If the number of dirty outer codewords is large, then an argument similar to the one mentioned above continues to work. But if the number of dirty outer codewords is small, then we need to make a small modification. Note that if the tampering function toggles two bits in the same block, then it preserves the parity. To counter this issue, we replace the inner code which encodes a secret $x \in \mathbb{Z}_{n|\mathbb{F}|}$ using the characteristic vector of a random subset of $[m]$ of size $\lceil m/3 \rceil + 3x$, where $m = 9n\,|\mathbb{F}| + 1$. Now, we can show that applying any number of non-constant channels to a block does not preserve the parity of the block with constant probability. Thus, generalizing the definition of "dirty" codewords, we can take care of non-constant channels as well.

A robust non-malleable code permits the tampering function to change the encoding but not the encoded message itself. Robust non-malleability is a strengthening of the traditional notion of non-malleability. Using algebraic geometric codes, instead of Reed-Solomon codes, we have the following theorem:

Theorem 1. *Let κ be the statistical security parameter and $L \in \mathbb{N}$. In the information theoretic setting, there exists an explicit robust non-malleable code with efficient encoding and decoding for messages in $\{0,1\}^L$ against the class of functions $\mathcal{C}_{\{0,1\}} \circ \mathcal{S}_N$, where codewords are in $\{0,1\}^N$, $N = \Theta(L^2)$ and $L \geqslant \lg^3 \kappa$.*

The basic non-malleable encoding scheme is summarized in Figure 6.

1.3.2 Main Construction

Let $\mathbb{F}$ be a suitably large field with characteristic 2. The message s is an L-bit message and is interpreted as an element in $\mathbb{F}^\ell$. This is shared using a Reed-Solomon secret sharing scheme with appropriately chosen parameters such that the number of shares $n \leqslant \ell + \ell^{1-c}$, where $c \in (0,1)$ is a constant. Let the shares be $c^{(1)} \in \mathbb{F}^n$. Next, we use the basic non-malleable encoding scheme to encode the message $\left(h, h\left(c^{(1)}\right)\right)$, where $h \xleftarrow{\$} \mathcal{H}$ and $\mathcal{H}$ is a suitably chosen (almost) universal hash function with weak collision resistance properties. We represent this as $c^{(2)}$ and shall be referred to as the "tag of $c^{(1)}$."

The final non-malleable encoding of the message s is the pair $c = \left(c^{(1)}, c^{(2)}\right)$. Parameters for the Reed-Solomon encoding and the hash family $\mathcal{H}$ can be appropriately chosen so that the rate $L/N = 1 - o(1)$ (see Figure 7).

Intuition of Non-Malleability Proof. We illustrate the main ideas underlying our analysis. Fix a tampering function $(\pi, f_1, \ldots, f_N)$. Let the tampered codeword be $\tilde{c} = (\tilde{c}^{(1)}, \tilde{c}^{(2)})$.

Case 1. If "too many" blocks in the tag $\tilde{c}^{(2)}$ receive bits from the code $c^{(1)}$. In this case, we show that all blocks have parity $0 \mod 3$ with $1 - \mathsf{negl}(\kappa)$ probability.

Case 2. If "too many" blocks in the tag $\tilde{c}^{(2)}$ receive bits from multiple blocks in $c^{(2)}$ itself. In this case, we use an analysis similar to "large number of dirty blocks" that all blocks have parity 0 mod 3 with $1 - \mathsf{negl}(\kappa)$ probability.

Case 3. In this case, the tampering function copies most blocks identically (except permuting bits within a block). This means that the outer codeword in the tag in $c^{(2)}$ and $\tilde{c}^{(2)}$ remains identical. Now, there are some cases to consider.

Case 3.a. If $c^{(1)}$ and $\tilde{c}^{(1)}$ are different then this yields a collision of the hash function, which happens only with $\mathsf{negl}(\kappa)$ probability. The reduction to collision resistance experiment can only be performed if the number of blocks in $c^{(2)}$ which are copied into $\tilde{c}^{(1)}$ is small.

Case 3.b. If large number of bits are not identically mapped in $c^{(1)}$ then the probability that $c^{(1)}$ and $\tilde{c}^{(1)}$ are equal is $\mathsf{negl}(\kappa)$.

Case 3.c. If the number of bits not identically mapped in $c^{(1)}$ are also small, then the probability that $c^{(1)} = \tilde{c}^{(1)}$ and the underlying outer codewords on $c^{(2)}$ and $\tilde{c}^{(2)}$ are equal is independent of the message being coded. This relies on the large distances of the Reed-Solomon code and the basic encoding scheme.

We can set parameters such that these cases are exhaustive. This provides our main result:

Theorem 2. *Let κ be the statistical security parameter and $L \in \mathbb{N}$. In the information theoretic setting, there exists an explicit robust non-malleable code with efficient encoding and decoding for messages in $\{0,1\}^L$ against the class of functions $\mathcal{C}_{\{0,1\}} \circ \mathcal{S}_N$, where codewords are in $\{0,1\}^N$, $N = L \times (1 + o(1))$ and $L \in [\lg^{11} \kappa, \kappa^{\lambda}]$, for a constant $\lambda \in \mathbb{N}$.*

The construction is summarized in Figure 7. The proof of this result, detailed in Section 5.1, closely follows the outline mentioned above. Our construction provides an alternate efficient explicit rate 1 AMD code [CDF+08].

2 Preliminaries

We denote the set $\{1, \dots, n\}$ by $[n]$. If $a \in [b - \varepsilon, b + \varepsilon]$, then we represent it as: $a = b \pm \varepsilon$. The set of all k-subsets of S is represented by $\binom{S}{k}$; and the set of all subsets of S is represented by 2^S.

Probability distributions are represented by bold capital alphabets, for example $\mathbf{X}$. The distribution $\mathbf{U}_S$ represents a uniform distribution over the set S. Given a distribution $\mathbf{X}$, $x \sim \mathbf{X}$ represents that x is sampled according to the distribution $\mathbf{X}$. And, for a set S, $x \xleftarrow{\$} S$ is equivalent to $x \sim \mathbf{U}_S$.

For a joint variable $\mathbf{X} = (\mathbf{X}_1, \dots, \mathbf{X}_n)$ and $S = \{i_1, \dots, i_{|S|}\} \subseteq [n]$, we define the random variable $\mathbf{X}_S = (\mathbf{X}_{i_1}, \dots, \mathbf{X}_{i_{|S|}})$. We use a similar notation for vectors as well, for example x_S represents the vector restricted to indices in the set S. For a function $f(\cdot)$, the random variable $\mathbf{Y} = f(\mathbf{X})$ represents the following distribution: Sample $x \sim \mathbf{X}$; and output $f(x)$. Further, $f(x_{[n]})$ represents the vector $f(x_1) \dots f(x_n)$. For example, $i + [n] = \{i + 1, \dots, i + n\}$.

The statistical distance between two distributions $\mathbf{S}$ and $\mathbf{T}$ over a finite sample space I is defined as:

$$\mathrm{SD}\left(\mathbf{S},\mathbf{T}\right):=\frac{1}{2}\sum_{i\in I}\left|\Pr_{s\sim\mathbf{S}}[s=i]-\Pr_{s\sim\mathbf{T}}[s=i]\right|$$

For a pair $z=\langle x,y\rangle$, define $\mathsf{first}(z){:=}x$ and $\mathsf{second}(z){:=}y$.

2.1 Classes of Tampering Functions

We shall consider the following set of tampering functions.

1. Family of Permutations. Let $\mathcal{S}_N$ denote the set of all permutations $\pi:[N]\to[N]$. Given an input codeword $x_{[N]}\in\{0,1\}^N$, tampering with function $\pi\in\mathcal{S}_N$ yields the following codeword: $x_{\pi^{-1}(1)}\dots x_{\pi^{-1}(N)}{=:}x_{\pi^{-1}([N])}$.
2. Family of Non-constant Channels. Let $\mathcal{C}_{\{0,1\}}$ denote the set of all binary channels f such that $\mathsf{Supp}(f(\mathbf{U}_{\{0,1\}}))=\{0,1\}$. The transition probabilities of these channels are constants. Given an input bit b, application of a channel f, produces the (randomized) output $f(b)$. Note that the following functions lie in this family: 1) $f(b)=1\oplus b$, i.e. negation of the input bit, 2) $f(b)=\mathbf{U}_{\{0,1\}}$, i.e. a uniform random bit irrespective of the input bit. We emphasize that the only channels whose transition probabilities are constants but are not included in this class are: $f(b)=0$ and $f(b)=1$. Note that this class contains infinitely many functions.

We can define more complex tampering function classes by composition of these function classes. For example, composition of $\mathcal{S}_N$ with $\mathcal{C}_{\{0,1\}}$ yields the following class of tampering functions. For any $\pi\in\mathcal{S}_N$ and $f_1,\dots,f_N\in\mathcal{C}_{\{0,1\}}$, it transforms a codeword $x_{[N]}$ into $f_1(x_{\pi^{-1}(1)})\dots f_N(x_{\pi^{-1}(N)}){=:}f_{1,\dots,N}(x_{\pi^{-1}([N])})$. This class is represented by: $\mathcal{C}_{\{0,1\}}\circ\mathcal{S}_N$. Our main result provides an efficient non-malleable code against the tampering class $\mathcal{C}_{\{0,1\}}\circ\mathcal{S}_N$.

2.2 Non-Malleable Codes Security Experiment

At a high level, our model describes a game between an honest challenger and a malicious adversary. In the first step, the adversary sends the tampering function f which will be applied to the codeword generated later. Next, the challenger, just seeing this tampering function, tries to estimate the failure probability σ_f of the adversary, which is the probability that the tampered codeword is invalid. For non-malleability, this failure probability should be independent of the message chosen by the adversary. Now, the adversary sends the message s to be encoded. The honest challenger computes the encoding (possibly randomized) of the message.

The adversary can succeed in the above game in two ways. First, it can try to make the failure probability σ_f depend on the message. Second, the adversary can also succeed if the tampered codeword is valid for some message $s'\neq s$ with non-negligible probability. We define two advantages $\mathsf{adv}_{1,\mathcal{A}}$ and $\mathsf{adv}_{2,\mathcal{A}}$, respectively, for the adversary in the above experiment.

The above intuituion is formalized in Figure 1. We say that the encoding scheme Enc is non-malleable against the class of tampering functions $\mathcal{F}$ if there exists $\nu=\mathsf{negl}(\kappa)$ such that $\mathsf{adv}_{1,\mathcal{A}}=\nu(\kappa)$ and $\mathsf{adv}_{2,\mathcal{A}}=\nu(\kappa)$.

Note that in our model any valid malleated codeword cannot encode any $s' \neq s$ with non-negligible probability. Hence, our notion of non-malleability is stronger than the standard notion of non-malleable codes considered in literature. We call it *robust*[5] non-malleability.

$\mathsf{Expt}_{\mathsf{Enc},A,\mathcal{F}}(1^\kappa)$:
It is an experiment between an honest challenger $\mathcal{H}$ and a malicious adversary $\mathcal{A}$, where $\mathsf{Enc} : \{0,1\}^L \to \mathcal{C} \subseteq \{0,1\}^N$ is the encoding scheme (possibly randomized) such that the message message is $\mathcal{M}{:=}\{0,1\}^L$ and $\mathcal{C} \subseteq \{0,1\}^N$ denotes the codeword space. $\mathcal{F}$ is the class of allowed tampering functions. Any $f \in \mathcal{F}$ maps $\mathcal{C}$ to an element in $\{0,1\}^N$.

1. $\mathcal{A}$ sends a tampering function $f \in \mathcal{F}$.
2. $\mathcal{H}$ runs the algorithm A to compute the failure probability σ_f.
3. $\mathcal{A}$ sends a message $s \in \mathcal{M}$.
4. $\mathcal{H}$ computes $c \leftarrow \mathsf{Enc}(m; r)$.

Define $\tilde{c} = f(c)$.
The adversary can succeed in the above experiment by two ways. Firstly, if the probability of $\tilde{c}$ being an invalid codeword is non-negligibly away from σ_f. Secondly, if $\tilde{c}$ is a valid codeword such that $\mathsf{Dec}(\tilde{c}) \neq m$ with non-negligible probability. Equivalently, these advantages, $\mathsf{adv}_{1,\mathcal{A}}$ and $\mathsf{adv}_{2,\mathcal{A}}$, respectively, can be defined as follows:

$$\begin{aligned} \mathsf{adv}_{1,\mathcal{A}} &= |\Pr[\tilde{c} \ \notin \ \mathcal{C}] - \sigma_f| \\ \mathsf{adv}_{2,\mathcal{A}} &= \Pr[\tilde{c} \ \in \mathcal{C} \ \wedge \ \mathsf{Dec}(\tilde{c}) \ \neq \ m] \end{aligned}$$

Both probabilities are taken over the randomness r used by the challenger to encode the messages.

Figure 1: Robust Non-Malleability Experiment

2.3 Hash Functions

Definition 1 (Almost Universal Hash Function Family)**.** *A family of hash functions $\mathcal{H}_\kappa : \mathcal{D}_\kappa \to \mathcal{R}_\kappa$ is an ε-almost universal hash function family, if for all $x, x' \in \mathcal{D}_\kappa$ such that $x \neq x'$ we have:*

$$\Pr_{h \xleftarrow{\$} \mathcal{H}_\kappa} \left[h(x) = h(x') \right] \leqslant \varepsilon$$

An $(1/|\mathcal{R}_\kappa|)$-almost universal hash function family is also called a universal hash function family. An ε-almost universal hash function family has weak collision resistance properties which shall suffice for our construction.

Let $T'(\cdot)$ be any function. Note that there exists a universal hash function family $\mathcal{H}'_\kappa : \{0,1\}^{2T'(\kappa)} \to$

[5] Our notion of robust non-malleability is intermediate to the notions of standard and strong non-malleability [DPW10]. In robust non-malleability it is acceptable if the underlying message does not change but the encoding itself changes.

$\{0,1\}^{T'(\kappa)}$ such that the description of the hashes in $\mathcal{H}'_\kappa$ is identical to the set $\{0,1\}^{3T'(\kappa)}$.[6]

Let $T^*(\cdot)$ be any function. Then there exists an ε^*-almost universal hash function family $\mathcal{H}^*_\kappa : \{0,1\}^{2T'(\kappa)T^*(\kappa)} \to \{0,1\}^{T'(\kappa)}$ such that the description of the hashes in $\mathcal{H}^*_\kappa$ is identical to $\{0,1\}^{3T'(\kappa)D(\kappa)}$ and $\varepsilon^* = D(\kappa) \cdot 2^{-\lg^3 \kappa}$, where $D(\kappa) = \lg 2T^*(\kappa)$. This hash function class is obtained by constructing a Merkle-tree using the hash function family $\mathcal{H}'_\kappa$ such that at each level of the Merkle-tree an independent hash function from $\mathcal{H}'_\kappa$ is used.

In particular, for any constant $\lambda \in \mathbb{N}$ and for any $\lg^3 \kappa \leqslant L \leqslant \kappa^\lambda$, there exists a hash function family $\mathcal{H}_\kappa : \{0,1\}^L \to \{0,1\}^{\lg^3 \kappa}$ with seed length at most $4\lambda \lg^4 \kappa$ and $\varepsilon^* = 4\lambda \lg \kappa \cdot 2^{-\lg^3 \kappa}$.

3 Building Blocks

In this section, we define encoding schemes (equivalently, secret sharing schemes) relevant to our construction.

Definition 2 (Secret Sharing Scheme)**.** *Consider alphabet sets $\Lambda_0, \Lambda_1, \ldots, \Lambda_m$ and a joint distribution $\mathbf{S} = (\mathbf{X}_0, \mathbf{X}_1, \ldots, \mathbf{X}_m)$ over the space $\Lambda_0 \times \Lambda_1 \times \cdots \times \Lambda_m$. The random variable $\mathbf{X}_0$ represents the secret being shared and $\mathbf{X}_i$ for $i \in [m]$ represents the i^{th} share. For $s \in \Lambda_0$ and set $T = \{i_1, \ldots, i_\ell\}$, the conditional distribution $(\mathbf{X}_T | \mathbf{X}_0 = s)$ is defined as the conditional distribution of $(\mathbf{X}_{i_1}, \ldots, \mathbf{X}_{i_\ell})$ under $\mathbf{S}$ when $\mathbf{X}_0 = s$. We define the following properties of the secret sharing schemes.*

1. *t-independence: For any $s \in \Lambda_0$, $T \subseteq [n]$ such that $|T| \leqslant t$, we have*

$$\mathrm{SD}\left((\mathbf{X}_T | \mathbf{X}_0 = s), \mathbf{U}_{\Lambda_T}\right) = 0$$

2. *t-privacy: For any $s_1, s_2 \in \Lambda_0$, $T \subseteq [n]$ such that $|T| \leqslant t$, we have*

$$\mathrm{SD}\left((\mathbf{X}_T | \mathbf{X}_0 = s_1), (\mathbf{X}_T | \mathbf{X}_0 = s_2)\right) = 0$$

3. *r-reconstruction: For any $s_1, s_2 \in \Lambda_0$, $T \subseteq [n]$ such that $|T| \geqslant r$, we have*

$$\mathrm{SD}\left((\mathbf{X}_T | \mathbf{X}_0 = s_1), (\mathbf{X}_T | \mathbf{X}_0 = s_2)\right) = 1$$

Consider a secret sharing scheme with r-reconstruction. Then any two different secrets s, s' have at least $m - r + 1$ different shares. Hence, we define the distance for this secret sharing scheme to be $m - r + 1$.

Secret Sharing Schemes. Below, we describe some secret sharing schemes which are relevant to our construction.

[6] The hash function family is the set of all random $T'(\kappa) \times 2T'(\kappa)$ Toeplitz matrices; and evaluation of the hash $h \in \{0,1\}^{T'(\kappa) \times 2T'(\kappa)}$ at $x \in \{0,1\}^{2T'(\kappa) \times 1}$ is a matrix multiplication $h \cdot x$. This family of hash function is, in fact, pairwise independent.

Secret Sharing Scheme $\mathbf{X}^{(\mathsf{RS},n,k,\ell,\mathbb{F})}$:

1. Sample space: $\Lambda_0 = \mathbb{F}^\ell$, $\Lambda_1 = \cdots = \Lambda_n = \mathbb{F}$.
2. Conditions: $|\mathbb{F}| \geqslant n + \ell$ and $n \geqslant k \geqslant \ell$.
3. Joint Distribution $(\mathbf{X}_0, \ldots, \mathbf{X}_n)$ is defined via the following sampling procedure: We assume that $\{f_{-\ell}, \ldots, f_{-1}, f_1, \ldots, f_n\} \subseteq \mathbb{F}$.
   (a) Pick a random polynomial: $p(x) = \sum_{i=0}^{k-1} a_i x^i$, where $a_i \xleftarrow{\$} \mathbb{F}$ and $i \in \{0\} \cup [k-1]$.
   (b) Define $x_0 = (p(f_{-1}), \ldots, p(f_{-\ell})) \in \mathbb{F}^\ell$.
   (c) Define $x_i = p(f_i)$, for $i \in [n]$.
   (d) Output $(x_0, x_{[n]})$.

Efficient Encoding and Decoding. Efficient sampling property for $\mathbf{X}^{(\mathsf{RS},n,k,\ell,\mathbb{F})}$ follows from the efficiency of Lagrange interpolation.

Figure 2: Basic Reed-Solomon based Secret Sharing.

Secret Sharing Scheme $\mathbf{X}^{(\mathsf{aRS},n,k,\ell,\mathbb{F})}$:

1. Sample space: $\Lambda_0 = \mathbb{F}^\ell$, $\Lambda_1 = \cdots = \Lambda_n = [n] \times \mathbb{F}$.
2. Conditions: $|\mathbb{F}| \geqslant n + \ell$ and $n \geqslant k \geqslant \ell$.
3. Joint Distribution $(\mathbf{X}_0, \ldots, \mathbf{X}_n)$ is defined via the following sampling procedure: We assume that $\{f_{-\ell}, \ldots, f_{-1}, f_1, \ldots, f_n\} \subseteq \mathbb{F}$.
   (a) Pick a random polynomial: $p(x) = \sum_{i=0}^{k-1} a_i x^i$, where $a_i \xleftarrow{\$} \mathbb{F}$ and $i \in \{0\} \cup [k-1]$.
   (b) Define $x_0 = (p(f_{-1}), \ldots, p(f_{-\ell})) \in \mathbb{F}^\ell$.
   (c) Define $x_i = \langle i, p(f_i) \rangle$, for $i \in [n]$.
   (d) Output $(x_0, x_{[n]})$.

Efficient Encoding and Decoding. Efficient sampling property for $\mathbf{X}^{(\mathsf{aRS},n,k,\ell,\mathbb{F})}$ follows from the efficiency of Lagrange interpolation.

Figure 3: Augmented Reed-Solomon based Secret Sharing.

Secret Sharing Scheme $\mathbf{X}^{(\mathsf{aAG},n,k,d,\ell,t,\mathbb{F}_q)}$:

1. Sample space: $\Lambda_0 = \mathbb{F}^\ell$, $\Lambda_1 = \cdots = \Lambda_n = [n] \times \mathbb{F}_q$.
2. Conditions: Let $\alpha, \beta, \alpha', \delta, q$ be constants such that:
   (a) $\alpha \in (0,1)$ and $\beta \in (0, 1-\alpha)$.
   (b) $\alpha' \in (0, \alpha)$ and $\delta \in (0, \alpha - \alpha')$.
   (c) $q^* = \max\left\{ \left(1 + \frac{1}{\beta}\right)^2, \left(1 + \frac{1}{\alpha - \alpha' - \delta}\right)^2, \left(1 + \frac{1}{1 - \alpha - \alpha'}\right)^2, 49 \right\}$
   (d) For all $q \geqslant q^*$ such that q is an even power of a prime, there are infinitely many $n' \in \mathbb{N}$ such that for $k = \alpha n'$, $d = (1 - \alpha - \beta)n'$, $\ell = \alpha' n'$ and $t = \delta n'$, there exists $[n', k, d]_q$ AG codes (see Figure 9). Set $n {:=} n' - \ell = (1 - \alpha')n'$.
3. Joint Distribution $(\mathbf{X}_0, \ldots, \mathbf{X}_n)$ is defined via the following sampling procedure:
   (a) Consider $[n', k, d]_q$ AG code over $\mathbb{F}_q$ as described in Figure 9.
   (b) Sample a random AG codeword $(z_{-\ell}, \ldots, z_{-1}, z_1, \ldots, z_n)$.
   (c) Define $x_0 = (z_{-\ell}, \ldots, z_{-1}) \in \mathbb{F}_q^\ell$.
   (d) Define $x_i = \langle i, z_i \rangle$, for $i \in [n]$.
   (e) Output $(x_0, x_{[n]})$.

Efficient Encoding and Decoding. Follows from [CC06] as described in Figure 9.

Figure 4: Augmented Algebraic-Geometric Code based Secret Sharing.

Secret Sharing Scheme $\mathbf{X}^{(\mathsf{unary},m,F,p)}$:

1. Sample space: $\Lambda_0 = \mathbb{Z}_{|F|}$, $\Lambda_1, \dots, \Lambda_m = \{0,1\}$.
2. Condition: $m = 3p\,|F| + 1$.
3. Joint Distribution $(\mathbf{X}_0, \dots, \mathbf{X}_m)$ is defined via the following sampling procedure: We assume that there exists a bijection from the set F to $\mathbb{Z}_{|F|}$, say $f \mapsto \mathsf{map}(f)$.
   (a) Pick $f \xleftarrow{\$} F$.
   (b) Define $x_0 = \mathsf{map}(f)$.
   (c) Sample $x_{[m]} \xleftarrow{\$} \binom{[m]}{\lceil m/3 \rceil + px_0}$. Here $x_{[m]}$ is the characteristic vector of the sampled subset.
   (d) Output $(x_0, x_{[m]})$.

Efficient Encoding and Decoding. It is easy to see that there exists efficient encoding and decoding schemes. In fact, it is also efficient to sample $x_S \sim (\mathbf{X}_S | \mathbf{X}_T = x_T)$, for any $S, T \subseteq \{0\} \cup [m]$.

Figure 5: Balanced Unary Secret-sharing Scheme.

1. Basic Secret Sharing scheme using Reed-Solomon codes $\mathbf{X}^{(\mathsf{RS},n,k,\ell,\mathbb{F})}$. This is a generalization of Massey secret sharing scheme [Mas95] and is commonly referred to as the "share-packing technique" for Reed-Solomon codes. This is an $[n,k]$ code over a field $\mathbb{F}$, such that $|\mathbb{F}| \geqslant n+\ell$. Let $\{f_{-\ell}, \dots, f_{-1}, f_1, \dots, f_n\} \subseteq \mathbb{F}$. The secret sharing of message $(s_1, \dots, s_\ell) \in \mathbb{F}^\ell$ is done by choosing a random polynomial $p(\cdot)$ of degree $< k$ conditioned on $(p(f_{-1}), \dots, p(f_{-\ell})) = (s_1, \dots, s_\ell)$. The shares $\{y_1, \dots, y_n\}$ are evaluations of $p(\cdot)$ at $\{f_1, \dots, f_n\}$ respectively. The formal description of the secret sharing scheme is provided in Figure 2. The field $\mathbb{F}$ will generally have characteristic 2 and this scheme will be used in our main construction presented in Section 5.

   The encoding has $(k-\ell)$-privacy (in fact, $(k-\ell)$ independence) and distance $d = n-k+1$.

2. Secret Sharing scheme using Reed-Solomon codes $\mathbf{X}^{(\mathsf{aRS},n,k,\ell,\mathbb{F})}$. Consider any $[n+\ell, k]$ code over finite field $\mathbb{F}$ such that $|\mathbb{F}| \geqslant n+\ell$. Given a message $s \in \mathbb{F}^\ell$, the secret sharing is performed as follows: Sample a random Reed-Solomon code conditioned on the fact that its first ℓ elements are identical to the message s. Let $y_1, \dots, y_n$ be the remaining elements in the codeword. The shares are defined to be $\langle 1, y_1 \rangle, \dots, \langle n, y_n \rangle$. It is known that efficient encoding and decoding exist using Lagrange interpolation. For formal description of this scheme refer to Figure 3.

   The encoding scheme has $(k-\ell)$-privacy and distance $d = n-k+1$.

3. Secret Sharing scheme using algebraic geometric codes $\mathbf{X}^{(\mathsf{aAG},n,k,\ell,\mathbb{F})}$ [CC06]. Consider $[n+\ell, k, d]_q$ AG code over finite field $\mathbb{F}_q$ for the choice of parameters specified in Figure 4. Given a message $s \in \mathbb{F}_q^\ell$, the secret sharing is performed as follows: Sample a random AG code conditioned on the fact that its first ℓ elements are identical to the message s. Let $y_1, \dots, y_n$ be the remaining elements in the codeword. The shares are defined to be $\langle 1, y_1 \rangle, \dots, \langle n, y_n \rangle$.

It is known that efficient encoding and decoding exist [CC06]. For formal description of this scheme refer to Figure 4.

The encoding scheme has $t = \delta n'$-privacy and $(\alpha n' + \frac{n'}{\sqrt{q}-1} + 1)$-reconstruction, where $n' = n + \ell$. For details on parameter setting for AG codes, see Appendix G.

4. Balanced unary secret sharing scheme $\mathbf{X}^{(\mathsf{unary},m,F,p)}$. Set $m{:=}3p\,|F|$, where F is the message space. Given a message $s \in F$, the secret sharing is performed as follows: Sample a random set S of $[m]$ of weight $\lceil m/3 \rceil + ps$. The shares are defined to be the characteristic vector of set S. Note that this scheme has efficient encoding and decoding. For a formal description refer to Figure 5.

   For any $s \in F$ and any set S used for encoding s, the weight of the final shares lie in $[m/3, 2m/3]$. Hence, the name balanced unary secret sharing scheme.

Definition 3 (Concatenation Codes.)**.** *Consider two encoding schemes, the outer encoding scheme* $\mathbf{X}^{(\mathsf{out})} = (\mathbf{X}_0^{(\mathsf{out})}, \mathbf{X}_1^{(\mathsf{out})}, \dots, \mathbf{X}_n^{(\mathsf{out})})$ *over* $\Lambda_0 \times \Lambda \times \cdots \times \Lambda$ *and the inner encoding scheme* $\mathbf{X}^{(\mathsf{in})} = (\mathbf{X}_0^{(\mathsf{in})}, \mathbf{X}_1^{(\mathsf{in})}, \dots, \mathbf{X}_m^{(\mathsf{in})})$ *over* $\Lambda \times \Lambda' \times \cdots \times \Lambda'$*. We define the concatenation code as the joint distribution* $\mathbf{X}^{(\mathsf{concat})} = (\mathbf{X}_0^{(\mathsf{concat})}, \mathbf{X}_1^{(\mathsf{concat})}, \dots, \mathbf{X}_{nm}^{(\mathsf{concat})})$ *over* $\Lambda_0 \times \Lambda' \times \cdots \times \Lambda'$*. Given a secret* $s \in \Lambda_0$*, sample* $\mathbf{x}_{[nm]} \sim (\mathbf{X}_{[nm]}^{(\mathsf{concat})} | \mathbf{X}_0^{(\mathsf{concat})} = s)$ *as follows: Sample* $\mathbf{x}_{[n]}^{(\mathsf{out})} \sim \left(\mathbf{X}_{[n]}^{(\mathsf{out})} \;\middle|\; \mathbf{X}_0^{(\mathsf{out})} = s \right)$*. Next, for each* $i \in [n]$*, sample* $\mathbf{x}_{(i-1)m+[m]} \sim \left(\mathbf{X}_{[m]}^{(\mathsf{in})} \;\middle|\; \mathbf{X}_0^{(\mathsf{in})} = \mathbf{x}_i^{(\mathsf{out})} \right)$*. Output* $\mathbf{x}_{[nm]}$*.*

Encoding and decoding procedures for concatenation codes are defined naturally using corresponding procedures for inner and outer encoding schemes. Note that the final encoding and decoding procedures are efficient if the corresponding procedures are efficient for inner and outer schemes.

Moreover, we emphasize that we do not focus on error correcting codes. In particular, if any of inner or outer decoding procedures fails, we output $\perp$ as the decoding of the overall code.

Suppose we have a codeword $c_{[n]}$ over $\mathbb{F}^n$ then c_i is referred to as the i^{th} element of the codeword. Now, consider a concatenation code where each element c_i is further encoded using an inner code over some field $(\mathbb{F}')^m$. The resultant codeword is $d_{[mn]} \in (\mathbb{F}')^{mn}$. The i^{th} block in $d_{[mn]}$ corresponds to the encoding of the i^{th} element of $c_{[n]}$.

4 Basic Non-Malleable Encoding Scheme

In this section, we give the construction for our non-malleable encoding scheme.

Construction. As a high level, our encoding scheme is a concatenation code (see Definition 3) which does the following: Given a message s, it samples an outer code according to augmented algebraic geometric code based secret sharing (see Figure 4). Then for each outer code element, it samples an inner codeword according to balanced unary secret sharing scheme (see Figure 5).

The choice of parameters for our scheme is as follows: Let κ be the statistical security parameter. As shown in Figure 4, the first step is to choose the constants $\alpha, \beta, \alpha', \delta, q$ satisfying certain conditions. We set these constants as follows: $\alpha{:=}3/7$, $\beta{:=}1/7$, $\alpha'{:=}1/7$, $\delta{:=}1/7$, $q{:=}64$. Note that these parameters satisfy all the conditions specified in Figure 4. Hence, there exists infinitely many $n' \in \mathbb{N}$ such that there exists $[n', k, d]_q$ AG codes for $k = \alpha n'$ and $d = (1 - \alpha - \beta)n'$.

Let $\mathbb{F}_q$ be a finite field with characteristic 2 and size 2^6. Choose smallest n' of the form $(\sqrt{q}-1)q^{u/2}$ such that $(1-\alpha')n' \geqslant \lg^3 \kappa$, where $u \in \mathbb{N}$. Define $\ell = \alpha' n'$ and $t = \delta n'$ and $n = n' - \ell \geqslant \lg^3 \kappa$. Note that $n \leqslant \sqrt{q} \lg^3 \kappa$.

For this choice of parameters, our message space is $\mathbb{F}_q^\ell$, which is identical to $\{0,1\}^{6\ell}$, i.e. $L = 6\ell$.

Our encoding scheme is formally defined in Figure 6. Our code is a subspace of $\{0,1\}^{nm}$, where $m = 9\,|F| + 1$, where $F = [n] \times \mathbb{F}_q$.

Let $N = n(9 \cdot 6 \cdot n + 1), n = L, 54L^2 + L$.

Property of outer encoding scheme. Due to the parameter setting in our scheme, the outer encoding scheme satisfies the following properties:

- $(\frac{n}{6})$-privacy.
- $(\frac{2n}{3} + 1)$-reconstruction and $(\frac{n}{3})$ distance.

Let $\{0,1\}^L$ be the message space such that $L \equiv 6 \cdot 8^u$ for some $u \in \mathbb{N}$.[a] Consider the following choice of parameters for the secret sharing scheme $\mathbf{X}^{(\mathsf{aAG}, n, k, d, \ell, t, \mathbb{F}_q)}$ (see Figure 4). Let $\alpha{:=}3/7$, $\beta{:=}1/7$, $\alpha'{:=}1/7$, $\delta{:=}1/7$, and $q{:=}64$. Note that these parameters satisfy all the conditions specified in Figure 4. Let $\mathbb{F}_q$ be finite field of characteristic 2 with $|\mathbb{F}_q| = 2^6$. Let $n' = 7 \cdot 8^u$. Let $k = \alpha n'$, $d = (1 - \alpha - \beta)n'$, $\ell = \alpha' n'$, $t = \delta n'$ and $n = n' - \ell$.
Note that $\mathbf{X}^{(\mathsf{aAG}, n, k, d, \ell, t, \mathbb{F})}$ exists for the above setting of parameters.

Let $\mathbf{X}^{(\mathsf{aAG}, n, k, d, \ell, t, \mathbb{F})}$ be the outer code $\mathbf{X}^{(\mathsf{out})}$; $\mathbf{X}^{(\mathsf{unary}, m, F, 3)}$ (see Figure 5) be the inner code $\mathbf{X}^{(\mathsf{in})}$ where $F = [n] \times \mathbb{F}_q$ and $m{:=}9\,|F| + 1$. Define $\mathbf{X}^{(\mathsf{basic})}$ as the concatenation of $\mathbf{X}^{(\mathsf{out})}$ with $\mathbf{X}^{(\mathsf{in})}$ (see Definition 3).

$\mathsf{Enc}_{\mathsf{basic}}(s \in \mathbb{F}^\ell)$:

1. Output $c_{[mn]} \sim (\mathbf{X}^{(\mathsf{basic})}_{[mn]} | \mathbf{X}^{(\mathsf{basic})}_0 = s)$.

$\mathsf{Dec}_{\mathsf{basic}}(c_{[mn]} \in \{0,1\}^{mn})$:

1. Decode $c_{[mn]}$ by decoding algorithm corresponding to $\mathbf{X}^{(\mathsf{basic})}$ code.

[a] If L is not of this form then consider the smallest number of form $6 \cdot 8^u$ greater than L. And pad the original message with sufficient 0s to encode it. This increases the length of the message by at most a multiplicative factor of 8.

Figure 6: Basic Non-malleable Code achieving rate $\geqslant 1/c^* L$, where $c^* = 440$.

Equivalence of codes for our scheme. We need the concept of equivalence of two codewords $g^{(\mathsf{nmc})}_{[nm]}$ and $h^{(\mathsf{nmc})}_{[nm]}$ which are equivalent if each block encodes identical outer codeword element. [7] Formally defined as follows:

Inner codes. Consider two inner codewords, $g^{(\mathsf{in})}_{[m]}$ and $h^{(\mathsf{in})}_{[m]}$. We say that $g^{(\mathsf{in})}_{[m]}$ and $h^{(\mathsf{in})}_{[m]}$ are equivalent codes if they encode the same message according to the inner code $\mathbf{X}^{(\mathsf{in})}$.

Non-Malleable codes. Consider two codewords $g^{(\mathsf{nmc})}_{[mn]}$ and $h^{(\mathsf{nmc})}_{[mn]}$. We say that $G^{(\mathsf{nmc})}_{[mn]} \cong h^{(\mathsf{nmc})}_{[mn]}$ if the following holds. Define $g^{(\mathsf{in})}_{i} = g^{(\mathsf{nmc})}_{(i-1)m+[m]}$ for all $i \in [n]$. Similarly, define $h^{(\mathsf{in})}_{i} = h^{(\mathsf{nmc})}_{(i-1)m+[m]}$ for all $i \in [n]$. Then, $g^{(\mathsf{in})}_{i} \cong h^{(\mathsf{in})}_{\pi(i)}$ for all $i \in [n]$.

4.1 Proof of Non-Malleability against $\mathcal{S}_N$

As a primer, in this section, we will consider the tampering functions $\mathcal{S}_N$, which is the set of all permutations $[N]$ to $[N]$. We show that our scheme described in Figure 6 is non-malleable against this class of tampering functions.

Proof Overview. Intuitively, a block in the tampered codeword is said to be dirty if it receives bits from at least two blocks of the given encoding. Formal definition is provided in Definition 5. Let n_{dirty} denote the number of such dirty codewords. We have the following two cases.

1. $n_{\mathsf{dirty}} < \lg^2 n$. We use high distance and high privacy of the outer encoding scheme compared to $\lg^2 n$ to argue the following. Failure probability is independent of the message being encoded and can be brute force calculated by assuming $s = 0^{\ell}$.
2. $n_{\mathsf{dirty}} \geqslant \lg^2 n$. Since the number of dirty codewords is large, we first fix the outer codeword and show for any fixing $\Pr[\tilde{c} \in \mathcal{C}] \leqslant \mathsf{negl}(n)$. We show this by first proving each dirty inner codeword is invalid with at least a constant probability and then showing that overall failure probability grows exponentially in n_{dirty}. Hence, we output failure probability as 1.

Formal Proof. In order to argue non-malleability of our scheme, we will describe an algorithm A specified in the security experiment which outputs the failure probability σ_f such that following conditions are satisfied. Let s be the message being encoded in the non-malleability security experiment. Let c be the code, and let $\tilde{c}$ be the malleated code.

$$\mathsf{adv}_{1,\mathcal{A}} = |\Pr[\tilde{c} \ \notin \ \mathcal{C}] - \sigma_f| \leqslant \mathsf{negl}(\kappa) \tag{1}$$

$$\mathsf{adv}_{2,\mathcal{A}} = \Pr[\tilde{c} \ \in \ \mathcal{C} \ \wedge \ \mathsf{Dec}_{n,k,\ell,\mathbb{F}}(\tilde{c}) \ \neq \ s] \leqslant \mathsf{negl}(\kappa) \tag{2}$$

Note that these conditions should hold independent of the message being encoded, which could have been chosen maliciously by $\mathcal{A}$.

[7] Note that we only insist that $g^{(\mathsf{nmc})}_{[nm]}$ and $h^{(\mathsf{nmc})}_{[nm]}$ not only encode the same message s but also every outer codeword element is identical.

Let the codeword given to the adversary $\mathcal{A}$ be $c_{[mn]} \sim (\mathbf{X}^{(\mathsf{nmc})}_{[mn]} | \mathbf{X}^{(\mathsf{nmc})}_0 = s)$. By the definition of concatenation codes, we can equivalently write it as $d^{(\mathsf{out})}_{[n]} \sim (\mathbf{X}^{(\mathsf{out})}_{[n]} | \mathbf{X}^{(\mathsf{out})}_0 = s)$ and $c_{[mn]} \sim (\mathbf{X}^{(\mathsf{nmc})}_{[mn]} | \mathbf{X}^{(\mathsf{out})}_{[n]} = d^{(\mathsf{out})}_{[n]})$. We also define $y^{(\mathsf{in})}_i = c_{(i-1)m+[m]}$. Note that $y^{(\mathsf{in})}_i$ refers to the i^{th} block of c corresponding to i^{th} outer codeword element, i.e. $d^{(\mathsf{out})}_i$.

Let the tampering function be $f \in \mathcal{F}$. Let $\tilde{c}_{[mn]} = f(c_{[mn]})$. For all $i \in [n]$, define $z^{(\mathsf{in})}_i = \tilde{c}_{(i-1)m+[m]}$. Here $z^{(\mathsf{in})}_i$ refers to the i^{th} block $\tilde{c}$.

We will model the attack as a weighted bipartite graph $G = (V_L, V_R, W)$ with partite sets $V_L = y^{(\mathsf{in})}_{[n]}$ and $V_R = z^{(\mathsf{in})}_{[n]}$ and a symmetric weight function $W : V_L \times V_R \to \{0\} \cup \mathbb{N}$, such that $W(u,v) = W(v,u)$. The weight on the edge between nodes $u \in V_L$ and $v \in V_R$, denoted by $w_{u,v}$, corresponds to the number of bits that have been copied from codeword u in $c_{[mn]}$ to codeword v in $\tilde{c}$. Note that these weights can be calculated by seeing the tampering function f alone. Also, since f is a permutation, this graph is regular, i.e. $n_R = n = n_L$ and $\deg_L(u) = \deg_R(v) = m$ for all $u \in V_L, v \in V_R$.

Now consider the connected components $\{C_i\}$ of graph G. A connected component C_i with two nodes, u, v such that $u \in V_L$, $v \in V_R$ and $w_{u,v} = m$ is called a *matching*. Matchings correspond to a copy of a block $y^{(\mathsf{in})}_u$ in c to a block $z^{(\mathsf{in})}_v$ in $\tilde{c}$. More formally, we define a matching as follows:

Definition 4 (Matching)**.** *For $i, j \in [n]$, $(y^{(\mathsf{in})}_i, z^{(\mathsf{in})}_j)$ is a matching if $y^{(\mathsf{in})}_i \cong z^{(\mathsf{in})}_j$. That is $y^{(\mathsf{in})}_i = z^{(\mathsf{in})}_j$ up to internal permutation of bits.*

Right nodes in a matching are always valid w.r.t. inner coding scheme $\mathbf{X}^{(\mathsf{unary},m,F,3)}$. Moreover, as mentioned above, any matching $(y^{(\mathsf{in})}_i, z^{(\mathsf{in})}_j)$ would result in an invalid code $\tilde{c}$ if $i \neq j$. This is because in outer encoding scheme (see Figure 3) we store the index of the outer codeword element. If there exists such a matching, we output $\sigma_f = 1$. Thus, without loss of generality, we assume that for any matching $(y^{(\mathsf{in})}_i, z^{(\mathsf{in})}_j)$, $i = j$. Let n_{match} denote the number of such matchings.

For components of size greater than 2 nodes, all nodes on the right receive bits from more than one blocks on the left. We call these *dirty* codewords. More formally,

Definition 5 (Dirty Codewords)**.** *For some $j \in [n]$, $z^{(\mathsf{in})}_j$ is said to be a dirty inner codeword if there is no $i \in [n]$ such that $(y^{(\mathsf{in})}_i, z^{(\mathsf{in})}_j)$ is a matching. In other words, there exists $i, i' \in [n]$ such that some bits have been copied from both $z^{(\mathsf{in})}_i$ and $z^{(\mathsf{in})}_{i'}$ to $y^{(\mathsf{in})}_j$.*

Let n_{dirty} be the number of dirty codewords in $\tilde{c}$. Note that this classification of blocks in $\tilde{c}$ depends solely on the tampering function f. Now we do a case analysis on n_{dirty} in order to analyze the validity of $\tilde{c}$.

- $n_{\mathsf{dirty}} < \lg^2 n$: In this case since the number of dirty codewords is “small", the non-malleability would hold using distance and privacy of the outer encoding scheme. This follows by the following two claims.

  Claim 1. $\tilde{c} \in \mathcal{C} \implies \tilde{c} \cong c$.

Proof. For the outer encoding scheme, we have distance, $d > \lg^2 n$. The claim follows by noting that in order to get a valid code for a different message, the adversary needs to change at least d outer codeword characters. □

Claim 2. *$\forall s_1, s_2 \in \mathcal{M}$ following holds*

$$\Pr[\tilde{c} \notin \mathcal{C} : c_{[mn]} \sim (\mathbf{X}^{(\mathsf{nmc})}_{[mn]} | \mathbf{X}^{(\mathsf{nmc})}_0 = s_1), \tilde{c} = f(c)]$$
$$= \Pr[\tilde{c} \notin \mathcal{C} : c_{[mn]} \sim (\mathbf{X}^{(\mathsf{nmc})}_{[mn]} | \mathbf{X}^{(\mathsf{nmc})}_0 = s_2), \tilde{c} = f(c)]$$

Proof. The claim follows by $\lg^2 n$-privacy of the outer encoding scheme.

For the class of permutations, the number of blocks of c which are used in creating dirty codewords in $\tilde{c}$ are exactly equal to n_{dirty}. But the outer encoding scheme has t-privacy for $t > \lg^2 n$. Hence, the outer code elements corresponding to $y_i^{(\mathsf{in})}$'s which are used to create dirty codewords are independent of the message. To calculate the probability in the claim above, we need to calculate the probability that permutations of these randomly chosen blocks (due to privacy) results in valid blocks which are consistent with the blocks that were copied directly (matchings). From the previous claim, it holds that these have to be equal to the outer codeword elements in c in order to be a valid code overall. In short, we need to calculate the probability that permutations blocks corresponding to certain randomly elements, results in the exactly same blocks (up to some natural allowed permutations of bits inside these blocks). Note that this probability would be independent of the message. □

In order to compute σ_f for this case, brute force compute the probability in Claim 4.1 by taking $s = 0^\ell$.

- $n_{\mathsf{dirty}} \geqslant \lg^2 n$: We will analyze this case for every fixed outer codeword elements but over the randomness used to generate the blocks. More precisely, given s the message being encoded, fix $d^{(\mathsf{out})}_{[n]} \in \mathsf{Supp}\left(\mathbf{X}^{(\mathsf{out})}_{[n]} \;\middle|\; \mathbf{X}^{(\mathsf{out})}_0 = s\right)$. Now our sample space is $\left(\mathbf{X}^{(\mathsf{in})}_{[nm]} \;\middle|\; \mathbf{X}^{(\mathsf{out})}_{[n]} = d^{(\mathsf{out})}_{[n]}\right)$.

  For a connected component which is not a matching, by Lemma 5, the probability that all the nodes on the right are valid blocks is at most $\mu^{n_{\mathsf{dirty},i}}$, where $n_{\mathsf{dirty},i}$ is the number of dirty codewords in C_i and $\mu \in (0,1)$ is a constant. Here, by a valid block we refer to blocks which have parity 0 mod 3.

  Since we had already fixed the outer codeword elements, these success probabilities over each component are independent of each other. Hence, we get

$$\Pr[\tilde{c} \in \mathcal{C}] \leqslant \left(\prod_{i : C_i \text{ is not a matching}} \mu^{n_{\mathsf{dirty,i}}} \right) \leqslant \mu^{n_{\mathsf{dirty}}} \leqslant \mathsf{negl}(n),$$

  where the last inequality follows by $n_{\mathsf{dirty}} > \lg^2 n$.

  In this case, we can set $\sigma_f = 1$.

4.2 Proof of Theorem 1

A formal proof is provided in Appendix C. We provide a high level proof below.

Intuitively, a block in the tampered codeword is said to be dirty if it receives bits from at least two blocks of the given encoding. Formal definition is provided in Definition 5. Let n_{dirty} denote the number of such dirty codewords. Moreover, a block in the tampered codeword is said to be dirty matching if it receives bits from one block of the codeword given but there is at least one bit which is passed through a non-constant channel. Note that simply copying a bit is a trivial channel and is ignored. For a formal definition, see Definition 12. Let $n_{\mathsf{match,dirty}}$ denote the number of such dirty matchings. Let $n_{\mathsf{D}} = n_{\mathsf{dirty}} + n_{\mathsf{match,dirty}}$. We have the following two cases.

1. $n_{\mathsf{D}} < \lg^2 n$. We use high distance and privacy of the outer encoding scheme compared to $\lg^2 n$ to argue the following. Failure probability is independent of the message being encoded and can be brute force calculated by assuming $s = 0^\ell$.

2. $n_{\mathsf{D}} \geqslant \lg^2 n$. Since the number of dirty codewords and dirty matchings is large, we first fix the outer codeword and show for any fixing $\Pr[\tilde{c} \in \mathcal{C}] \leqslant \mathsf{negl}(n)$. We first prove that a codeword obtained via a dirty matching is invalid with constant probability. We also show that each dirty codeword is invalid with a constant probability. Finally, we show that the failure probability grows exponentially in n_{D}. Hence, we output failure probability as 1.

5 Rate 1 Non-Malleable Encoding Scheme

Our main construction is presented in Figure 7.

Construction. Our encoding scheme does the following at an intuitive level. It encodes a given message $s \in \{0,1\}^L$ using Reed-Solomon secret sharing scheme (see Figure 2); and appends a suitable tag to it using the basic non-malleable encoding scheme presented in Figure 6.

We choose our parameters as follows. Suppose we are interested in encoding L bit messages such that $\lg^3 \kappa \leqslant L \leqslant \kappa^\lambda$, where $\lambda \in \mathbb{N}$ is a constant and κ is the statistical security parameter. We choose a characteristic 2 field $\mathbb{F}$ such that $\lg|\mathbb{F}| = \lceil \lg(2L + \lg^9 \kappa) \rceil$. Define $\ell = \left\lceil \frac{L}{\lg|\mathbb{F}|} \right\rceil$ and $n = k = \ell + \lg^9 \kappa$.

Let $\mathcal{H}_\kappa : \{0,1\}^{n \lg|\mathbb{F}|} \to \{0,1\}^{\lg^3 \kappa}$ be the ε^*-almost universal hash function family defined in Section 2.3. Such a hash function family has seed length $4\lambda \lg^4 \kappa$ and $\varepsilon^* = (3\lambda+1)\lg\kappa \cdot 2^{-\lg^3 \kappa} = \mathsf{negl}(\kappa)$. The weak collision resistance properties of this family suffices for our construction.

To encode a message $s \in \{0,1\}^L$, we interpret it as an element in $\mathbb{F}^\ell$. Next, we share this message according to the secret sharing scheme $\mathbf{X}^{(\mathsf{RS},n,k,\ell,\mathbb{F})}$. The resulting collection of shares is an element in $\mathbb{F}^n$ which is interpreted as a binary string $c^{(1)}_{[N^{(1)}]}$, where $N^{(1)} = n \lg|\mathbb{F}|$.

To, generate the tag we do the following. First, sample $h \xleftarrow{\$} \mathcal{H}_\kappa$ and compute $t = \left(h, h\left(c^{(1)}_{[N^{(1)}]}\right)\right)$. Number of bits in t is at most $4\lambda \lg^4 \kappa + \lg^3 \kappa \ll 5\lambda \lg^4 \kappa$. Encoding of t using our basic non-malleable

For message length L and statistical security parameter κ, let $\mathbb{F}$ be a characteristic 2 field such that: $\lg|\mathbb{F}| = \lceil \lg(2L + \lg^9 \kappa) \rceil$. Let $\ell = \left\lceil \frac{L}{\lg|\mathbb{F}|} \right\rceil$. Let $n = k = \ell + \lg^9 \kappa$. Suppose $L \leqslant \kappa^\lambda$, where $\lambda \in \mathbb{N}$ is a constant. Define $N^{(1)} = n \lg|\mathbb{F}|$, $N^{(2)}$ be the (bit) length of non-malleable encoding of a message of length $5\lambda \lg^4 \kappa$ according to $\mathbf{X}^{(\mathsf{basic})}$, and $N = N^{(1)} + N^{(2)}$. Let $\mathcal{H}_\kappa : \{0,1\}^{n \lg|\mathbb{F}|} \to \{0,1\}^{\lg^3 \kappa}$ be the ε^*-almost universal hash function family defined in Section 2.3. It has seed length $4\lambda \lg^4 \kappa$ and $\varepsilon^* = \lambda \lg \kappa \cdot 2^{-\lg^3 \kappa} = \mathsf{negl}(\kappa)$.

$\mathsf{Enc}(s \in \{0,1\}^L)$:

1. Let $c^{(1)}_{[N^{(1)}]} \sim \left(\mathbf{X}^{(\mathsf{RS},n,k,\ell,\mathbb{F})}_{[n]} | \mathbf{X}^{(\mathsf{RS},n,k,\ell,\mathbb{F})}_0 = s \right)$. Here we interpret s as an element in $\mathbb{F}^\ell$; and the shares $\in \mathbb{F}^n$ as element in $\{0,1\}^{N^{(1)}}$.
2. Draw $h \xleftarrow{\$} \mathcal{H}_\kappa$ and define $t = \left(h, h\left(c^{(1)}_{[N^{(1)}]}\right)\right)$. Let $c^{(2)}_{[N^{(2)}]} \sim \left(\mathbf{X}^{(\mathsf{basic})}_{[N^{(2)}]} | \mathbf{X}^{(\mathsf{basic})}_0 = t \right)$.
3. Output $\left(c^{(1)}_{[N^{(1)}]}, c^{(2)}_{[N^{(2)}]}\right)$.

$\mathsf{Dec}(c \in \{0,1\}^N)$:

1. Let $\left(c^{(1)}_{[N^{(1)}]}, c^{(2)}_{[N^{(2)}]}\right) \equiv c$.
2. Decode $c^{(2)}_{[N^{(2)}]}$ by the decoding algorithm of $\mathbf{X}^{(\mathsf{basic})}$ to obtain (h, z).
3. If $h\left(c^{(1)}_{[N^{(1)}]}\right) = z$, then: Output $s \in \{0,1\}^L$ obtained by decoding $c^{(1)}_{[N^{(1)}]}$ according to the decoding algorithm of $\mathbf{X}^{(\mathsf{RS},n,k,\ell,\mathbb{F})}$.

Figure 7: Main Non-malleable Code achieving rate 1, if $L \geqslant \lg^{11} \kappa$.

scheme shall produce a code of length $N^{(2)} \leqslant 25c^*\lambda^2 \lg^8 \kappa$, where $c^* = 440$ as defined in Figure 6. We represent this encoding as $c^{(2)}_{[N^{(2)}]}$.

Note that $N^{(1)} \leqslant (L+\lg|\mathbb{F}|)+\lg|\mathbb{F}|\lg^9\kappa \leqslant L+\Theta(\lg^{10}\kappa)$ and $N^{(2)} \leqslant 25c^*\lambda^2\lg^8\kappa$. So, for $L \geqslant \lg^{11}\kappa$ our construction is rate 1.

To decode our main non-malleable encoding scheme, we decode the tag and obtain (h, z). Next, check whether $h\left(c^{(1)}_{[N^{(1)}]}\right) = z$ or not. If yes, then output the decoding of $c^{(1)}_{[N^{(1)}]}$.

In the parameter setting done above, we have ensured that: $(k-\ell) \geqslant 2N^{(2)} + \lg^2\kappa$ and bit length of t is at least $\lg^3\kappa$ (which is required to keep our basic encoding scheme secure).

5.1 Proof of Theorem 2

Consider a tampering function $(\pi, f_1, \dots, f_N) \in \mathcal{C}_{\{0,1\}} \circ \mathcal{S}_N$. The tampered code is $\tilde{c}_{[N]} \equiv \left(\tilde{c}^{(1)}_{[N^{(1)}]}, \tilde{c}^{(2)}_{[N^{(2)}]}\right)$.

Given the tampering function $(\pi, f_1, \dots, f_N)$, we define the following sets:

1. Let P be the set of indices $i \in [N^{(1)}]$ such that $\pi(i) \notin [N^{(1)}]$.
2. Let Q be the set of indices $i \in [N] \setminus [N^{(1)}]$ such that $\pi(i) \in [N^{(1)}]$. Define $\langle Q \rangle$ as the set of blocks such that an index in the block belongs to Q.
3. Let $\overline{Q}$ be the set of indices in $i \in [N] \setminus [N^{(1)}]$ such that $\pi^{-1}(i) \in [N^{(1)}]$. Define $\langle \overline{Q} \rangle$ as the set of blocks such that an index in the block belongs to $\overline{Q}$.
4. Let M be the set of indices $i \in [N^{(1)}]$ such that $\pi(i) \neq i$ or f_i is not the message forwarding channel.
5. Let R' be the set of blocks in $c^{(2)}_{[N^{(2)}]}$ such that there exists an index i in the block such that $\pi^{-1}(i)$ is not in the block or f_i is a not a message forwarding channel. Let R be the set of all the indices corresponding to the blocks in R'.

Let $\tau = \lg^2\kappa$ be a threshold parameter. We consider the following case analysis.

Case 1. $\left|\langle \overline{Q} \rangle\right| > \tau$. Consider a block indexed by $\overline{Q}$. We define the "bundle" corresponding to this block as the set of all indices which maps from $[N^{(1)}]$ to this block. We shall show that the parity mod 3 of each bundle is unpredictable. Since there are large number of bundles, the probability that all of them have $0 \mod 3$ parity is $\mathsf{negl}(\kappa)$. The actual proof for this case is slightly complicated by the fact that the message being encoded by the basic encoding scheme is dependent upon the encoding $c^{(1)}_{[N^{(1)}]}$. The full proof is provided in Appendix D.1.

Case 2. $\left|R' \setminus \langle \overline{Q} \rangle\right| > \tau$. There are large number of blocks in the tag which are dirty and receive all their bits from the tag only. We shall show that in this case, the probability that $\tilde{c}^{(2)}_{[N^{(2)}]}$ is a valid codeword is $\mathsf{negl}(\kappa)$. This case is similar in spirit to the case where there are large number of "dirty" codewords in our basic encoding scheme. The full proof for this case is provided in Appendix D.2.

Case 3. $\left|\langle\overline{Q}\rangle\right| \leqslant \tau$ and $\left|R' \setminus \langle\overline{Q}\rangle\right| \leqslant \tau$. First note that, we have $|\langle Q\rangle| \leqslant 2\tau$. Suppose not. Let the number of blocks in our basic encoding scheme be $n^{(2)}$. Then the number of blocks which do not receive bits from other blocks and have message forwarding channel applied to each of its indices is $n^{(2)} - \left|\langle\overline{Q}\rangle\right| - \left|R' \setminus \langle\overline{Q}\rangle\right|$. This is the same as the number of blocks all of whose indices are mapped inside their block and the channel applied at each destination index is the message forwarding channel. So, we get: $n^{(2)} - \left|\langle\overline{Q}\rangle\right| - \left|R' \setminus \langle\overline{Q}\rangle\right| = n^{(2)} - |\langle Q\rangle| - |R' \setminus \langle Q\rangle|$. Then $\left|R' \setminus \langle\overline{Q}\rangle\right| \geqslant |\langle Q\rangle| - \left|\langle\overline{Q}\rangle\right| > \tau$; which is a contradiction.

Further, note that the distance of the outer code of the basic encoding scheme is $\Theta(\lg^3 \kappa)$. If only $|R'| \leqslant \left|R' \setminus \langle\overline{Q}\rangle\right| + \left|\langle\overline{Q}\rangle\right| \leqslant 2\tau$ blocks are changed then the only way $\tilde{c}^{(2)}_{[N^{(2)}]}$ can be a valid codeword is when: the underlying outer codeword of $c^{(2)}_{[N^{(2)}]}$ and $\tilde{c}^{(2)}_{[N^{(2)}]}$ are identical.

After these two observations, this case is further subdivided into cases.

Case 3.a. $c^{(1)}_{[N^{(1)}]} \neq \tilde{c}^{(1)}_{[N^{(1)}]}$. Here the message being encoded in the basic encoding scheme does not change. So, we want to leverage the collision resistance of our hash function family. Note that $|\langle Q\rangle| \leqslant 2\tau$. So, the distribution over those blocks is independent of the message being encoded in the tag. So, we can construct $c^{(1)}_{[N^{(1)}]}$ and then simulate the construction of $\tilde{c}^{(1)}_{[N^{(1)}]}$ (by simulating the distribution over the blocks indexed by $\langle Q\rangle$). This shall violate the collision resistance property of the hash function. Therefore, this happens only with $\mathsf{negl}(\kappa)$ probability.

Case 3.b. $c^{(1)}_{[N^{(1)}]} = \tilde{c}^{(1)}_{[N^{(1)}]}$ and $M > 2N^{(2)} + \tau$. Note that at most $N^{(2)}$ elements in M are mapped to indices in the tag; and at most $N^{(2)}$ elements in M receive maps from indices in the tag. So, there is a set M' in M of size τ such that: a) For any index $i \in M'$ we have $\pi(i) \in M$, and b) For any index $i \in M'$ we have $\pi^{-1}(i) \in M$. Note that M' and $\pi(M')$ are both subsets of M and the size of their union is at most 2τ. Since the independence of the Reed-Solomon code is much large than 2τ, the distribution over $c^{(1)}_{M' \cup \pi(M')}$ is uniform. So, the probability that $c^{(1)}_{\pi(M')} = \tilde{c}^{(1)}_{\pi(M')}$ is at most $\mathsf{negl}(\kappa)$.[8]

Case 3.c. $c^{(1)}_{[N^{(1)}]} = \tilde{c}^{(1)}_{[N^{(1)}]}$ and $M \leqslant 2N^{(2)} + \tau$. Note that the probability of: "$c^{(1)}_{[N^{(1)}]} = \tilde{c}^{(1)}_{[N^{(1)}]}$ and decryptions of $c^{(2)}_{[N^{(2)}]}$ and $\tilde{c}^{(2)}_{[N^{(2)}]}$ are identical" is same as the probability of: "$c^{(1)}_{M} = \tilde{c}^{(1)}_{M}$ and the underlying (augmented) $\mathbb{F}_{64}$ element encoded by blocks $c^{(2)}_{R}$ and $\tilde{c}^{(2)}_{R}$ are identical." This is because the distance of the basic encoding scheme is $\Theta(\lg^3 \kappa)$.

[8] The argument that $c^{(1)}_{\pi(M')} = \tilde{c}^{(1)}_{\pi(M')}$ is at most $\mathsf{negl}(\kappa)$ is slightly complicated by the fact that there are dependencies among these equations. But we can show that the probability of these equations being satisfied is at most $2^{-\tau/2}$. To see this, consider the following graph over the vertex set $\{0\} \cup M' \cup \pi(M')$. The directed edge (i, j) exists if $j = \pi(i)$. If $\pi(i)$ is outside $M' \cup \pi(M')$, then we add a directed edge $(i, 0)$. Note that the in- and out-degree of this graph (except vertex 0) is 1; and vertex 0 has no out-degree. So, we can decompose this graph into vertex disjoint cycles and paths. The probability that the equations corresponding to a cycle C are satisfied is $2^{-|C|+1}$, where $|C|$ is the size of the cycle C. The probability that the equations corresponding to a path P are satisfied is $2^{-|P|+1}$, where $|P|$ is the number of vertices in the path P. This gives an overall bound that $c^{(1)}_{\pi(M')} = \tilde{c}^{(1)}_{\pi(M')}$ with probability at most $2^{-|M'|/2}$.

Since independence of the Reed-Solomon code is higher than $|M|$, the distribution of $c_M^{(1)}$ is independent of the message s being encoded. Further, the privacy of the basic encoding scheme is $\Theta(\lg^3 \kappa)$ which is larger than $|R'|$. So, the distribution over $c_R^{(2)}$ does not depend on the message begin encoded.

Therefore, this probability can be computed by first sampling $c_M^{(1)}$ and $c_R^{(2)}$ and then checking whether $c_M^{(1)} = \tilde{c}_M^{(1)}$ and the underlying (augmented) $\mathbb{F}_{64}$ elements encoded in $c_R^{(2)}$ and $\tilde{c}_R^{(2)}$ are identical.

Computation of Failure Probability. Finally, to compute the failure probability σ_f, for a tampering function $f \in \mathcal{C}_{\{0,1\}} \circ \mathcal{S}_N$, check whether $\left|\left\langle \overline{Q} \right\rangle\right| \leqslant \tau$ and $\left|R' \setminus \left\langle \overline{Q} \right\rangle\right| \leqslant \tau$ and $M \leqslant 2N^{(2)} + \tau$. If this is the case, then compute the probability that $c_M^{(1)} = \tilde{c}_M^{(1)}$ and the underlying (augmented) $\mathbb{F}_{64}$ codeword represented by $c_R^{(2)}$ and $\tilde{c}_R^{(2)}$ are identical, when $c_M^{(1)}$ is picked uniformly at random and $\tilde{c}_R^{(1)}$ is picked based on the distribution as induced by the basic encoding scheme. If the check mentioned above is not true then output $\sigma_f = 1$. This completes the proof of Theorem 2.

References

[ADL13] Divesh Aggarwal, Yevgeniy Dodis, and Shachar Lovett. Non-malleable codes from additive combinatorics. *Electronic Colloquium on Computational Complexity (ECCC). To appear STOC-2014.*, 20:81, 2013. 4

[Can00] Ran Canetti. Security and composition of multiparty cryptographic protocols. *J. Cryptology*, 13(1):143–202, 2000. 38

[Can01] Ran Canetti. Universally composable security: A new paradigm for cryptographic protocols. In *FOCS*, pages 136–145. IEEE Computer Society, 2001. 38

[CC06] Hao Chen and Ronald Cramer. Algebraic geometric secret sharing schemes and secure multi-party computations over small fields. In Cynthia Dwork, editor, *CRYPTO*, volume 4117 of *Lecture Notes in Computer Science*, pages 521–536. Springer, 2006. 13, 14, 15, 44, 45

[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 Nigel P. Smart, editor, *EUROCRYPT*, volume 4965 of *Lecture Notes in Computer Science*, pages 471–488. Springer, 2008. 3, 8

[CG14a] Mahdi Cheraghchi and Venkatesan Guruswami. Capacity of non-malleable codes. In Moni Naor, editor, *ITCS*, pages 155–168. ACM, 2014. 3, 4

[CG14b] Mahdi Cheraghchi and Venkatesan Guruswami. Non-malleable coding against bit-wise and split-state tampering. In Lindell [Lin14], pages 440–464. 3

[Chv79] Vasek Chvátal. The tail of the hypergeometric distribution. *Discrete Mathematics*, 25(3):285 – 287, 1979. 41

[CKM11] Seung Geol Choi, Aggelos Kiayias, and Tal Malkin. Bitr: Built-in tamper resilience. In Dong Hoon Lee and Xiaoyun Wang, editors, *ASIACRYPT*, volume 7073 of *Lecture Notes in Computer Science*, pages 740–758. Springer, 2011. 4

[CKO14] Nishanth Chandran, Bhavana Kanukurthi, and Rafail Ostrovsky. Locally updatable and locally decodable codes. In Lindell [Lin14], pages 489–514. 4

[DBL10] *51th Annual IEEE Symposium on Foundations of Computer Science, FOCS 2010, October 23-26, 2010, Las Vegas, Nevada, USA*. IEEE Computer Society, 2010. 26, 27

[DDN91] Danny Dolev, Cynthia Dwork, and Moni Naor. Non-malleable cryptography (extended abstract). In *STOC*, pages 542–552. ACM, 1991. 4, 37

[DKO13] Stefan Dziembowski, Tomasz Kazana, and Maciej Obremski. Non-malleable codes from two-source extractors. In Ran Canetti and Juan A. Garay, editors, *CRYPTO (2)*, volume 8043 of *Lecture Notes in Computer Science*, pages 239–257. Springer, 2013. 4

[DPW10] Stefan Dziembowski, Krzysztof Pietrzak, and Daniel Wichs. Non-malleable codes. In Andrew Chi-Chih Yao, editor, *ICS*, pages 434–452. Tsinghua University Press, 2010. 3, 10, 37

[FMNV14] Sebastian Faust, Pratyay Mukherjee, Jesper Buus Nielsen, and Daniele Venturi. Continuous non-malleable codes. In Lindell [Lin14], pages 465–488. 4

[FMVW13] Sebastian Faust, Pratyay Mukherjee, Daniele Venturi, and Daniel Wichs. Efficient non-malleable codes and key-derivation for poly-size tampering circuits. *IACR Cryptology ePrint Archive. To appear Eurocrypt-2014.*, 2013:702, 2013. 1, 3, 4

[FV11] Lance Fortnow and Salil P. Vadhan, editors. *Proceedings of the 43rd ACM Symposium on Theory of Computing, STOC 2011, San Jose, CA, USA, 6-8 June 2011*. ACM, 2011. 26

[Gop81] Valerii Denisovich Goppa. Codes on algebraic curves. In *Soviet Math. Dokl*, pages 170–172, 1981. 45

[Goy11] Vipul Goyal. Constant round non-malleable protocols using one way functions. In Fortnow and Vadhan [FV11], pages 695–704. 4, 39

[GS96] Arnaldo Garcia and Henning Stichtenoth. On the asymptotic behaviour of some towers of function fields over finite fields. *Journal of Number Theory*, 61(2):248–273, 1996. 45

[GS10] Venkatesan Guruswami and Adam Smith. Codes for computationally simple channels: Explicit constructions with optimal rate. In *FOCS* [DBL10], pages 723–732. 4

[HO08] Brett Hemenway and Rafail Ostrovsky. Public-key locally-decodable codes. In Wagner [Wag08], pages 126–143. 4

[KMO10] Eike Kiltz, Payman Mohassel, and Adam O'Neill. Adaptive trapdoor functions and chosen-ciphertext security. In Henri Gilbert, editor, *EUROCRYPT*, volume 6110 of *Lecture Notes in Computer Science*, pages 673–692. Springer, 2010. 38

[Lin14] Yehuda Lindell, editor. *Theory of Cryptography - 11th Theory of Cryptography Conference, TCC 2014, San Diego, CA, USA, February 24-26, 2014. Proceedings*, volume 8349 of *Lecture Notes in Computer Science*. Springer, 2014. 25, 26

[Lip94] Richard J. Lipton. A new approach to information theory. In Patrice Enjalbert, Ernst W. Mayr, and Klaus W. Wagner, editors, *STACS*, volume 775 of *Lecture Notes in Computer Science*, pages 699–708. Springer, 1994. 4

[LL12] Feng-Hao Liu and Anna Lysyanskaya. Tamper and leakage resilience in the split-state model. In Reihaneh Safavi-Naini and Ran Canetti, editors, *CRYPTO*, volume 7417 of *Lecture Notes in Computer Science*, pages 517–532. Springer, 2012. 4

[LP11] Huijia Lin and Rafael Pass. Constant-round non-malleable commitments from any one-way function. In Fortnow and Vadhan [FV11], pages 705–714. 4

[Mas95] James Lee Massey. Some applications of coding theory in cryptography. In *Codes and Ciphers: Cryptography and Coding IV*, pages 33–47, 1995. 14, 44

[MPSW05] Silvio Micali, Chris Peikert, Madhu Sudan, and David A. Wilson. Optimal error correction against computationally bounded noise. In Joe Kilian, editor, *TCC*, volume 3378 of *Lecture Notes in Computer Science*, pages 1–16. Springer, 2005. 4

[MS09] Steven Myers and Abhi Shelat. Bit encryption is complete. In *FOCS*, pages 607–616. IEEE Computer Society, 2009. 4

[OPS07] Rafail Ostrovsky, Omkant Pandey, and Amit Sahai. Private locally decodable codes. In Lars Arge, Christian Cachin, Tomasz Jurdzinski, and Andrzej Tarlecki, editors, *ICALP*, volume 4596 of *Lecture Notes in Computer Science*, pages 387–398. Springer, 2007. 4

[O'S95] M. E. O'Sullivan. Decoding of codes defined by a single point on a curve. *IEEE Transactions on Information Theory*, 41(6):1709–1719, 1995. 46

[PPV08] Omkant Pandey, Rafael Pass, and Vinod Vaikuntanathan. Adaptive one-way functions and applications. In Wagner [Wag08], pages 57–74. 38

[PR07] Manoj Prabhakaran and Mike Rosulek. Rerandomizable rcca encryption. In Alfred Menezes, editor, *CRYPTO*, volume 4622 of *Lecture Notes in Computer Science*, pages 517–534. Springer, 2007. 38

[SAK$^+$01] Kenneth W. Shum, Ilia Aleshnikov, P. Vijay Kumar, Henning Stichtenoth, and Vinay Deolalikar. A low-complexity algorithm for the construction of algebraic-geometric codes better than the gilbert-varshamov bound. *IEEE Transactions on Information Theory*, 47(6):2225–2241, 2001. 46

[Wag08] David Wagner, editor. *Advances in Cryptology - CRYPTO 2008, 28th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 17-21, 2008. Proceedings*, volume 5157 of *Lecture Notes in Computer Science*. Springer, 2008. 26, 27

[Wee10] Hoeteck Wee. Black-box, round-efficient secure computation via non-malleability amplification. In *FOCS* [DBL10], pages 531–540. 39

A Ensuring Independence

Definition 6 (Weighted Bipartite Graph)**.** *Let $G = (V_L, V_R, W)$ be a weighted bipartite graph with partite sets V_L and V_R and a symmetric weight function $W : V_L \times V_R \to \{0\} \cup \mathbb{N}$, such that $W(u, v) = W(v, u)$. The weight on the edge between nodes $u \in V_L$ and $v \in V_R$ is denoted by $w_{u,v}$.*

Let $n_L = |V_L|$ and $n_R = |V_R|$. Without loss of generality we assume that $V_L = [n_L]$ and $V_R = [n_R]$. The graph G is *connected* if there exists a path from every $u \in V_L$ to $v \in V_R$ via edges with positive weights.

The *left degree* of a vertex $u \in V_L$ is denoted by $\deg_L(u) := \sum_{j \in V_R} w_{u,j}$. Similarly, the *right degree* of a vertex $v \in V_R$ is denoted by $\deg_R(v) := \sum_{i \in V_L} w_{i,v}$. An m-regular bipartite graph has $\deg_L(u) = m = \deg_R(v)$, for all $u \in V_L$ and $v \in V_R$. Note that for an m-regular graph, $n_L = n_R$.

An *ordering* of the right partite set is defined by a permutation $\pi : [n_R] \to [n_R]$ on V_R.

Let $G = (V_L, V_R, W)$ be a weighted connected bipartite graph. An edge (u, v) is *k-blue* w.r.t. an ordering π if $w_{u,v} > 0$ and the following conditions are satisfied:

1. $\sum_{j \in V_R : \pi(j) \leqslant \pi(v)} w_{u,j} < \deg_L(u)$
2. $\sum_{j \in V_R : \pi(j) < \pi(v)} w_{u,j} < \deg_L(u) - k$

And edge (u, v) is *k-red* w.r.t. an ordering π if $w_{u,v} > 0$ and it is not k-blue w.r.t. the ordering π. Further, a node $v \in V_R$ is *k-blue* w.r.t. an ordering π if there exists an edge incident on it which is k-blue w.r.t. π.

We emphasize that the classification of an edge as k-blue or k-red edge depends on the ordering π of the nodes in V_R.

Property 1. *For all $u \in V_L$ there exists $v_1 \neq v_2$ such that $w_{u,v_1} > 0$ and $w_{u,v_2} > 0$.*

Observation 1. *Given a weighted bipartite graph G satisfying Property 1 and an ordering π, for any $u \in V_L$ define $\mathsf{first}_\pi(u)$ as the unique $v^* \in V_R$ such that $w_{u,v^*} > 0$ and $\forall v \in V_R$, if $w_{u,v} > 0$ then $\pi(v^*) \leqslant \pi(v)$. Note that if $\deg_L(u) > k$ then the edge (u, v^*) is always k-blue w.r.t. π.*

Claim 3. *Let G be a weighted bipartite graph satisfying Property 1 such that $\deg_L(u) \geqslant 2k$ for all $u \in V_L$. If an edge (u, v) is k-red w.r.t. an ordering $\pi = (\pi_1, \ldots, \pi_{n_R})$ then (u, v) is k-blue w.r.t. $\pi_{\mathsf{rev}} := (\pi_{n_R}, \ldots, \pi_1)$.*

Proof. We have the following two cases for edge (u, v).

1. $\sum_{j \in V_R : \pi(j) \leqslant \pi(v)} w_{u,j} = \deg_L(u)$: Note that $v_u^* = v$ w.r.t. π_{rev}. Since G satisfies Property 1, (u, v) is k-blue w.r.t. π_{rev} by Observation 1.
2. $\sum_{j \in V_R : \pi(j) < \pi(v)} w_{u,j} \geqslant \deg_L(u) - k$: Since $\deg_L(u) \geqslant 2k$, $\sum_{j \in V_R : \pi(j) \geqslant \pi(v)} w_{u,j} \leqslant k$. Thus, $\sum_{j \in V_R : \pi_{\mathsf{rev}}(j) < \pi_{\mathsf{rev}}(v)} w_{u,j} < k < \deg_L(u) - k$. Hence, (u, v) is k-blue w.r.t. π_{rev}.

□

Lemma 1. *Let G be a connected weighted bipartite graph satisfying Property 1 such that $\deg_L(u) \geqslant 2k$ for all $u \in V_L$. There exists an ordering π such that at least $n/2$ vertices in V_R are k-blue w.r.t. the ordering π.*

In particular, $\pi \in \{\pi_1 = (1, \ldots, n_R), \pi_2 = (n_R, \ldots, 1)\}$.

Proof. Here we will prove the second statement in the theorem. More precisely, we will show that a vertex $v \in V_R$ is k-blue w.r.t. at least π_1 or π_2. In particular, wlog if $v \in V_R$ is not k-blue w.r.t. π_1, then it is k-blue w.r.t. π_2. Since v is not k-blue in π_1, all the edges incident on v are k-red. Since G is connected, there is at least one edge incident on v. By Claim 3, this edge is k-blue w.r.t. π_2. Hence, v is k-blue w.r.t. π_2.

The lemma follows by an averaging argument. □

B Unpredictability

Given a distribution $\mathcal{D}$ over a set S and a function $f : S \to R$, define the distribution $f(\mathcal{D})$ over set R by the following sampling procedure: Sample $x \sim \mathcal{D}$. Output $f(x)$.

Definition 7 (δ-Balanced)**.** *A distribution $\mathcal{D}$ over a set S is δ-balanced if*

$$\forall s \in S, \left(\Pr_{x\sim\mathcal{D}}(x = s) > 0\right) \implies \left(\Pr_{x\sim\mathcal{D}}(x = s) \in [\delta, 1-\delta]\right)$$

Definition 8 (α-Unpredictability)**.** *A distribution $\mathbf{D}$ over sample space $\mathcal{S}$ is said to be α-*unpredictable*, if there exists $s_0, s_1 \in \mathcal{S}$ such that $\Pr_{s\sim\mathbf{D}}[s = s_0]$, $\Pr_{s\sim\mathbf{D}}[s = s_1] \geqslant \alpha$ and $s_0 \neq s_1$.*

Definition 9 (Weight, Density, Dense, Sparse)**.** *For an n-bit binary string $x_{[n]}$, its* weight *(represented as $\mathsf{wt}(x_{[n]})$) is the number of 1s in it. Its* density *is defined to be $\mathsf{wt}(x_{[n]})/n$. It is α-*dense *if its density is at least α; and it is α-*sparse *if its density is at most α.*

Now consider the weighted bipartite graph G as described in the Appendix A. Moreover, let G be a m-regular bipartite graph with $n_L = n = n_R$ and $\deg_L(u) = \deg_R(v) = m$ for all $u \in V_L, v \in V_R$. Next, we label the vertices in V_L by elements in Λ_0. More precisely, let $\mathsf{map} : V_L \to \Lambda_0$ mapping vertices in V_L to Λ_0. The label on $u \in V_L$ is denoted by $\mathsf{map}(u)$. For the rest of the analysis, fix any labeling map for the vertices V_L.

We emphasize that the analysis holds for any arbitrary labeling.

For each $u \in V_L$, $\mathsf{map}(u)$ is encoded using the encoding scheme $\mathbf{X}^{(\mathsf{unary},m,\Lambda_0,3)}$ (see Figure 5). Note that $m = 9|\Lambda_0|$. Also, we will choose the parameter used in red/blue labeling of edges and vertices in Appendix A as $k = m/2$.

Next, $[m]$ is sequentially partitioned into $S_{u,1}, \ldots, S_{u,n}$ such that $[m] = S_{u,1} \cup \ldots \cup S_{u,n}$ and $|S_{u,j}| = w_{u,j}, \forall j \in [n]$. For any vertex $u \in V_L$, sample $\mathbf{x}_{u,[m]} \sim (\mathbf{X}_{[m]} | \mathbf{X}_0 = m(u))$. For $j \in [n_R]$ with $w_{u,j} = 0$, define $\mathbf{B}_{u,j} = 0$. For each $j \in [n_R]$ with $w_{u,j} > 0$, we define $\mathbf{B}_{u,j} = \sum_{i \in S_{u,j}} \mathbf{X}_{u,i}$. In other words, $\mathbf{B}_{u,j}$ is the random variable representing $\mathsf{wt}(\mathbf{X}_{u,S_{u,j}})$.

Now consider an ordering π of V_R such that the number of vertices which are k-blue w.r.t. π are at least $n/2$. Such a ordering π is guaranteed to exist by Lemma 1. Let the set of k-blue vertices w.r.t. π be $\mathsf{S}^{\pi}_{\mathsf{blue}}$ such that $|\mathsf{S}^{\pi}_{\mathsf{blue}}| \geqslant n/2$. We share process these vertices in ascending order as induced by π, i.e. a vertex $v \in \mathsf{S}^{\pi}_{\mathsf{blue}}$ is processed before $v' \in \mathsf{S}^{\pi}_{\mathsf{blue}}$ if and only if $\pi(v) < \pi(v')$.

Consider the next vertex $v \in \mathsf{S}^{\pi}_{\mathsf{blue}}$ w.r.t. π. Then there is an edge, say (u, v), incident on v which is k-blue w.r.t. π. We shall analyze the distribution on vertex v given the fixings of all the edges to vertices $v' \in V_R$ such that $\pi(v') < \pi(v)$. Let $\mathbf{Y}_v = \cup_{i \in V_L} \cup_{j \in V_R : \pi(j) < \pi(v)} \mathbf{X}_{S_{i,j}}$. Also, let all the edges incident on v apart from (u, v) be $\mathbf{G}_v = \cup_{i \neq u} \mathbf{X}_{S_{i,v}}$.

We partition the outgoing edges from vertex u into three sets $F, S, T \subset [m]$ as follows: $F = \cup_{j : \pi(j) < \pi(v)} S_{u,j}$, $S = S_{u,v}$ and $T = \cup_{j : \pi(j) > \pi(v)} S_{u,j}$. Note that by definition of a k-blue edge, $T \neq \emptyset$.

Figure 8 shows the various sets of edges defined above.

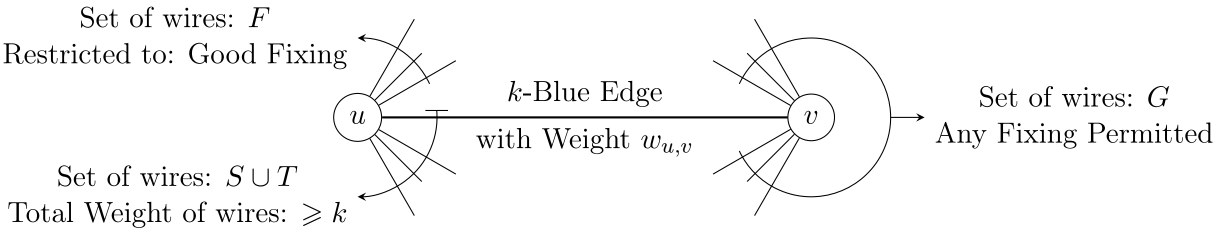

Figure 8: Argument about "Why Blue Edges are Unpredictable?"

For the analysis below, we begin by making the following observation. Though we need to condition the analysis of edge (u, v) on $(\mathbf{Y}_v, \mathbf{G}_v)$, it is sufficient to condition on $\mathbf{X}_F, \mathbf{G}_v$. In particular, we claim the following:

Claim 4. $(\mathbf{B}_{u,v} \bmod 3 | \mathbf{Y}_v = y_v, \mathbf{G}_v = g_v) \equiv (\mathbf{B}_{u,v} \bmod 3 | \mathbf{X}_F = x_F, \mathbf{G}_v = g_v)$, *where x_F is restriction of y_v to the set F.*

Let $\mathbf{Z}_v$ be the random variable bit string at the node v in this graph. Let $\mathbf{P}_v = \mathsf{wt}(\mathbf{Z}_v) \bmod 3$. In order to show that $\mathbf{Z}_v$ is a valid encoding according to Mod_{Λ_0} conditioned on $\mathbf{Y}_v$ with at most a constant probability, we do the following: We show that $(\mathbf{P}_v|\mathbf{Y})$ is β-unpredictable for some constant $\beta \in (0, 1)$ (Lemma 4). In this direction, we first prove that $(\mathbf{B}_{u,v} \bmod 3 | \mathbf{X}_F, \mathbf{G}_v)$ is α-unpredictable for some constant $\alpha \in (0, 1)$ (Lemma 3).

We prove these lemmas conditioned on the fact that $\mathbf{X}_F$ comes from a good distribution. Hence, we begin by defining a good fixing for $\mathbf{X}_F$. A good fixing intuitively means that even after setting the edges from the vertex u which go to prior vertices, there are a sufficient number of both $0s$ and $1s$ in $\mathbf{X}_{S_{u,v} \cup T}$. More precisely, we define it as follows:

Definition 10. *(c-Good Fixing) Let x^*_F be a fixing for the variable $\mathbf{X}_F$. We say that x^*_F is a c-good fixing if for all $x \in \mathsf{Supp}(\mathbf{X}_{S \cup T} | \mathbf{X}_F = x^*_F)$, x is c-dense and $(1 - c)$-sparse.*

We emphasize that above definition is independent of the weight of the edge (u, v).

Lemma 2. *Sample $\mathbf{x}_{u,[m]} \sim (\mathbf{X}_1, \ldots, \mathbf{X}_m | \mathbf{X}_0 = \mathsf{map}(u))$. Then, $\forall c \in (0, 1/3)$, $\exists \nu = \mathsf{negl}(\kappa)$ such that $\Pr[\mathbf{x}_{u,F}$ is a c-good fixing$] \geqslant 1 - \nu$, where the probability is taken over the randomness of the sampling procedure.*

Proof. This follows from Lemma 8 by noting that $\mathsf{wt}(\mathbf{x}_{u,[m]}) \in [m/3, 2m/3]$ and $|S_{u,v}| + |T| > m/2$. □

Lemma 3 (Unpredictability of $\mathbf{B}_{u,v} \bmod 3$)**.** *Let $c \in (0,1)$ be a constant such that x^*_F be a c-good fixing for $\mathbf{X}_F$. Let $\mathbf{G} = g_v$. Then, there exists a constant $\alpha > 0$ such that $(\mathbf{B}_{u,v} \bmod 3 | \mathbf{X}_F = x^*_F, \mathbf{G}_v = g_v)$ is α-unpredictable.*

Proof. Since x^*_F is a c-good fixing, $\mathbf{X}_{S_{u,v} \cup T}$ is c-dense and $(1-c)$-sparse. By Lemma 9, there exists a constant $\alpha > 0$ such that $(\mathbf{B}_{u,v} \bmod 3 | \mathbf{X}_F = x^*_F, \mathbf{G}_v = g_v)$ is α-unpredictable.

□

Lemma 4 (Unpredictability of $(\mathbf{P}_v | \mathbf{Y}_v = y_v)$)**.** *There exists a constant $\beta \in (0,1)$ and $\nu = \mathsf{negl}(\kappa)$ such that $(\mathbf{P}_v | \mathbf{Y}_v)$ is β-unpredictable with probability $1 - \nu(\kappa)$.*

Proof. Sample $\mathbf{x}_{u,[m]} \sim (\mathbf{X}_1, \ldots, \mathbf{X}_m | \mathbf{X}_0 = \mathsf{map}(u))$. Then by Lemma 2, $\forall c \in (0, 1/3)$, $\exists \nu = \mathsf{negl}(\kappa)$ such that $\Pr[\mathbf{x}_{u,F}$ is a $c-$good fixing$] \geqslant 1 - \nu$. We call this a good event. Now, given such a c-good fixing x^*_F and any fixing g_v of $\mathbf{G}_v$, by Lemma 3, $(\mathbf{B}_{u,v} \bmod 3 | \mathbf{X}_F = x^*_F, \mathbf{G}_v = g_v)$ is α-unpredictable for a constant $\alpha \in (0,1)$. This implies that $(\mathbf{B}_{u,v} \bmod 3 | \mathbf{Y}_v = y_v, \mathbf{G}_v = g_v)$ is α-unpredictable (Claim 4). Hence, $(\mathbf{P}_v | \mathbf{Y}_v = y_v, \mathbf{G}_v = g_v)$ is α-unpredictable.

Since there are $\binom{3}{2}$ pairs of possible parity values, by an averaging argument over g_v, $(\mathbf{P}_v | \mathbf{Y}_v = y_v)$ is β-unpredictable for $\beta = \alpha \binom{3}{2}^{-1}$ conditioned on the good event. □

Lemma 5. $\Pr[\forall j \in V_R,\ \mathbf{P}_j \equiv 0 \bmod 3] \leqslant (1-\gamma)^{n_R/2}$ *for some constant $\gamma \in (0,1)$ with probability* $1 - \mathsf{negl}(\kappa)$*. Further, this holds even for a graph with multiple connected components.*

Proof.

$$\begin{aligned}
\Pr[\forall j \in V_R,\ \mathbf{P}_j \equiv 0 \bmod 3] &\leqslant \Pr[\forall j \in \mathsf{S}^\pi_{\mathsf{blue}},\ \mathbf{P}_j \equiv 0 \bmod 3] \\
&= \prod_{j \in \mathsf{S}^\pi_{\mathsf{blue}}} \Pr[\mathbf{P}_j \equiv 0 \bmod 3 | \mathbf{Y}_j = y_j] \\
&\leqslant (1 - \beta + \nu(\kappa))^{|\mathsf{S}^\pi_{\mathsf{blue}}|},
\end{aligned}$$

where $\nu = \mathsf{negl}(\kappa)$. The last inequality follows from Lemma 4. The lemma follows by noting that $|\mathsf{S}^\pi_{\mathsf{blue}}| \geqslant n_R/2$.

Note that the above analysis was done for a given mapping map. Thus, analysis of each components can be done independently. Hence, this analysis extends naturally to any bipartite graph G. □

C Proof of Theorem 1

In this section, we consider the family of tampering functions in the composition of bit level permutations and non-constant channels. We show that our scheme described in Figure 6 is non-malleable against this class of tampering functions. In particular, we will describe an algorithm A specified in the security experiment which outputs the failure probability σ_f such that following conditions are satisfied. Let s be the message being encoded in the non-malleability security experiment.

$$\mathsf{adv}_{1,\mathcal{A}} = |\Pr[\tilde{c} \notin \mathcal{C}] - \sigma_f| \leqslant \mathsf{negl}(\kappa) \tag{3}$$

$$\mathsf{adv}_{2,\mathcal{A}} = \Pr[\tilde{c} \in \mathcal{C} \wedge \mathsf{Dec}_{n,k,\ell,\mathbb{F}}(\tilde{c}) \neq s] \leqslant \mathsf{negl}(\kappa) \tag{4}$$

Note that these conditions should hold independent of the message being encoded, which could have been chosen maliciously by $\mathcal{A}$.

Let $f_{\mathsf{copy}} \in \mathcal{C}_{\{0,1\}}$ be a non-constant channel such that $f_{\mathsf{copy}}(b) = b, \forall b \in \{0,1\}$. In this section, for ease of analysis, we will not consider f_{copy} in the class $\mathcal{C}_{\{0,1\}}$. Since this channel leaves the bit b unchanged, equivalently, this will be seen as a simple copy of the bit.

In this section, we consider tampering functions in the composition of $\mathcal{S}_{mn}$ and $f_{i_1}, \ldots, f_{i_z} \in \mathcal{C}_{\{0,1\}}$ for some $\mathcal{I} = \{i_1, \ldots, i_z\}$ such that $\mathcal{I} \subseteq [nm]$.

Let s be the message being encoded. Then, $c_{[mn]} \sim (\mathbf{X}^{(\mathsf{nmc})}_{[mn]} | \mathbf{X}^{(\mathsf{nmc})}_0 = s)$ be the codeword generated. By the definition of concatenation codes, we can equivalently write it as $d^{(\mathsf{out})}_{[n]} \sim (\mathbf{X}^{(\mathsf{out})}_{[n]} | \mathbf{X}^{(\mathsf{out})}_0 = s)$ and $c_{[mn]} \sim (\mathbf{X}^{(\mathsf{nmc})}_{[mn]} | \mathbf{X}^{(\mathsf{out})}_{[n]} = d^{(\mathsf{out})}_{[n]})$. We also define $y^{(\mathsf{in})}_i = c_{(i-1)m+[m]}$. Note that $y^{(\mathsf{in})}_i$ refers to the i^{th} block of c corresponding to i^{th} outer codeword element, i.e. $d^{(\mathsf{out})}_i$.

Let the tampering function be F. Let $\tilde{c}_{[mn]} = F(c_{[mn]})$. For all $i \in [n]$, define $z^{(\mathsf{in})}_i = \tilde{c}_{(i-1)m+[m]}$. Here $z^{(\mathsf{in})}_i$ refers to the i^{th} block $\tilde{c}$.

Given the tampering function F, similar to the case of permutations, we define the concept of a matching and a dirty codeword for tampered blocks. Recall that a block in $\tilde{c}$ is said to be made by a matching if it copies all the bits from one block in c (see Definition 4). Otherwise, it is called a dirty codeword (see Definition 5).

In this section, we will further classify the type of matchings. Consider a matching $(y^{(\mathsf{in})}_i, z^{(\mathsf{in})}_j)$ for some $i, j \in [n]$. We will have two kinds of matching: pure and dirty. We call a matching a "pure" matching if bits have been copied directly to bits in a single block of $\tilde{c}$ without using any non-constant channel. A matching is called a "dirty" matching if it is not a pure matching. More formally, we define them as follows:

Definition 11 (Pure Matching). *For $i, j \in [n]$, a matching $(y^{(\mathsf{in})}_i, z^{(\mathsf{in})}_j)$ is said to be pure if for all $k \in ((j-1)m + [m])$, $k \notin \mathcal{I}$, where $\mathcal{I} \subseteq [mn]$ refers to the set of indices to which non-constant channels have been applied.*

Moreover, as mentioned above, any pure matching $(y^{(\mathsf{in})}_i, z^{(\mathsf{in})}_j)$ would result in an invalid code $\tilde{c}$ if $i \neq j$. This is because in outer encoding scheme (see Figure 3) we store the index of the outer code elements. If there exists such a matching, we output $\sigma_f = 1$. Thus, without loss of generality, we assume that for any pure matching $(y^{(\mathsf{in})}_i, z^{(\mathsf{in})}_j)$, $i = j$.

Definition 12 (Dirty Matching). *For $i, j \in [n]$, a matching $(y_i^{(\mathsf{in})}, z_j^{(\mathsf{in})})$ is said to be dirty if $\exists k \in ((j-1)m + [m])$ such that $k \in \mathcal{I}$, where $\mathcal{I} \subseteq [mn]$ refers to the set of indices to which non-constant channels have been applied.*

Define $n_{\mathsf{match,pure}}$ and $n_{\mathsf{match,dirty}}$ to be the number of pure and dirty matchings, respectively. Define n_{dirty} as the number of dirty codewords. Finally, define $n_{\mathsf{D}} := n_{\mathsf{match,dirty}} + n_{\mathsf{dirty}}$.

Depending on the tampering function F we have the following two cases:

○ $n_{\mathsf{D}} < \lg^2 n$: In this case, we prove the following claims:

  Claim 5. $\tilde{c} \in \mathcal{C} \implies \tilde{c} \cong c$.

  Proof. The claim follows from the fact that for the outer encoding scheme, we have distance, $d > \lg^2 n$. □

  Claim 6. $\forall s_1, s_2 \in \mathcal{M}$ *following holds*

$$\Pr[\tilde{c} \notin \mathcal{C} : c_{[mn]}^{(\mathsf{nmc})} \sim (\mathbf{X}_{[mn]}^{(\mathsf{nmc})} | \mathbf{X}_0^{(\mathsf{nmc})} = s_1), \tilde{c} = F(c)]$$
$$= \Pr[\tilde{c} \notin \mathcal{C} : c_{[mn]}^{(\mathsf{nmc})} \sim (\mathbf{X}_{[mn]}^{(\mathsf{nmc})} | \mathbf{X}_0^{(\mathsf{nmc})} = s_2), \tilde{c} = F(c)]$$

  Proof. The claim follows by $\lg^2 n$-privacy of the outer encoding scheme.

  Note that for this class of tampering functions, the number of blocks of c which are used in creating dirty matchings and dirty codewords in $\tilde{c}$ are exactly equal to n_{D}. But the outer encoding scheme has t-privacy where $t \gg \lg^2 n$. Hence, the outer code elements corresponding to $y_i^{(\mathsf{in})}$'s which are used to create dirty matchings and dirty codewords are independent of the message. To calculate the probability in the claim above, we need to calculate the probability that dirty matchings and (dirty) permutations of the blocks for these randomly chosen elements (due to privacy) results in valid blocks which are consistent with the blocks that were copied directly (matchings). From previous claim, it holds that these have to be equal to the blocks in c in order to be a valid code overall. In short, we need to calculate the probability that dirty matchings and dirty permutations of blocks for certain randomly chosen elements results in the exactly same blocks (up to some natural allowed permutations of bits within blocks). Note that this probability would be independent of the message. □

  In order to compute σ_f for this case, brute force compute the probability in Claim 6 by taking $s = 0^\ell$.

○ $n_{\mathsf{D}} \geqslant \lg^2 n$. We will analyze this case for every fixed outer code elements but over the randomness used to generate the blocks. More precisely, given s the message being encoded, fix $d_{[n]}^{(\mathsf{out})} \in \mathsf{Supp}\left(\mathbf{X}_{[n]}^{(\mathsf{out})} \,\middle|\, \mathbf{X}_0^{(\mathsf{out})} = s\right)$. Now our sample space is $\left(\mathbf{X}_{[nm]}^{(\mathsf{in})} \,\middle|\, \mathbf{X}_{[n]}^{(\mathsf{out})} = d_{[n]}^{(\mathsf{out})}\right)$.

  Next, consider the bipartite graph $G = (V_L, V_R, W)$ corresponding to the dirty matchings and dirty codewords as described in Appendix B. Let $\{C_i\}$ be the connected components of G. Now, we will consider each component separately. For a component, if it consists of a dirty matching $(y_i^{(\mathsf{in})}, z_j^{(\mathsf{in})})$, then by Lemma 10, there exists a constant $\mu_1 \in (0,1)$ such that the parity (mod 3) of $z_j^{(\mathsf{in})}$ is μ_1-unpredictable. Since all valid blocks have parity (0 mod 3), $z_j^{(\mathsf{in})}$ will be an invalid block with probability at least μ_1.

For a connected component which is not a matching, an analysis similar to Appendix B and using Lemma 11, will show that there exists a constant $\mu_2 \in (0,1)$ such that the probability that all the nodes on the right are valid is at most $\mu_2^{n_{\mathsf{dirty},i}}$, where $n_{\mathsf{dirty},i}$ is the number of dirty codewords in C_i.

Since we had already fixed the outer code elements, these success probabilities over each component are independent of each other. Hence, we get

$$\Pr[\tilde{c} \in \mathcal{C}] \leqslant \left(\prod_{i: C_i \text{ is a matching}} (1-\mu_1) \right) \left(\prod_{i: C_i \text{ is not a matching}} \mu_2^{n_{\mathsf{dirty,i}}} \right) \leqslant \mu^{n_{\mathsf{D}}} \leqslant \mathsf{negl}(n),$$

where $\mu \in (0,1)$ is some constant and the last inequality follows by $n_{\mathsf{D}} > \lg^2 n$.

In this case, we can set $\sigma_f = 1$.

D Proof of Theorem 2

The only proofs which need to be presented are proofs for Case 1 and 2 as presented in Section 5. These two proofs are presented below.

D.1 Proof of Case 1

Here we will prove the following lemma:

Lemma 6. *If* $\left|\langle \overline{Q} \rangle\right| > \tau$, $\Pr[\tilde{c}^{(2)}$*is a valid codeword according to basic scheme*$] \leqslant \mathsf{negl}(\kappa)$, *where the probability is over the randomness used to pick the codeword for the message* s.

Proof. In order to analyse this case, we will pick a codeword for message s using following steps.

1. Pick $c_P^{(1)}$ uniformly at random.
2. Pick a code $c_{[N^{(1)}]}^{(1)}$ for message s consistent with $c_P^{(1)}$. Note that this is possible since independence of $c_{[N^{(1)}]}^{(1)}$ is more than $N^{(2)}$.
3. Pick a hash function h uniformly at random. Then $t = (h, h(c_{[N^{(1)}]}^{(1)}))$. Recall that the basic non-malleable encoding scheme is a concatenation code. Pick an outer codeword $d_{[n]}^{(\mathsf{out})}$ for message t.
4. Pick an inner code for each outer code element in $d_{[n]}^{(\mathsf{out})}$. We will denote the length of each block by m.

Without loss of generality, let $1, \ldots, \left|\langle\overline{Q}\rangle\right|$ be the blocks of $d^{(\mathsf{out})}_{[n]}$ which are in $\langle\overline{Q}\rangle$. For any i consider the bundle of indices from $c^{(1)}_P$ which are being copied to block i. This may be followed by application of non-constant channels. Let $X_{i,1}, \ldots, X_{i,c(i)}$ be the set of indices such that $c(i)$ is the number of indices being copied to block i. We classify the blocks in $\langle\overline{Q}\rangle$ into two types. If a block i has $c(i) = m$, we call it is a full block. Let otherwise the block is called non-full block. Let n_{full} be the number of full blocks and $n_{\mathsf{non-full}}$ be the number of non-full blocks.

We have the following two exhaustive cases cases:

1. $n_{\mathsf{full}} > \tau/2$: Consider a full block. In $\tilde{c}$, this complete block is made by copying indices from $c^{(1)}_P$. The validity of this block is independent of the choice of the outer codeword element of this block in $d^{(\mathsf{out})}_{[n]}$. Hence, we will analyze this block over the randomness used to pick bits in $c^{(1)}_P$.

   Note that non-constant channels may be applied to some or all of these indices $X_{i,j}$. Let $X'_{i,j} = f_{i,j}(X_{i,j})$. Also, let $Y_i = \sum_j X'_{i,j} \mathsf{mod} 3$. We claim the following:

   Claim 7. *Let $Y_i = \sum_j X_{i,j} \mathsf{mod} 3$. Then Y_i is α-unpredictable, for some constant $\alpha > 0$.*

   Proof. $Y_i = \sum_j X'_{i,j} \mathsf{mod} 3 = \left(f_{i,1}(X_{i,1}) + \sum_{j>1} X'_{i,j}\right) \mathsf{mod} 3$. Now, for any fixing of $X_{i,2}, \ldots, X_{i,m}$ and any fixing of the randomness of $f_{i,2}, \ldots, f_{i,m}$, Y_i is α_i-unpredictable because $X_{i,1} \equiv \mathbf{U}_2$ and $f_{i,1}$ is a non-constant channel. Here $\alpha_i > 0$ is some constant. Note that it is okay that α_i might depend on the channels applied in this block. □

   From above claim, we get that parity mod 3 of this block is μ-unpredictable. Since only parity 0 mod 3 blocks are valid, Pr[Block i is valid] $\leqslant (1 - \alpha_i)$.

   Since the bits in $c^{(1)}_P$ which are copied to different full blocks are chosen independently, we have the following:

   $$\Pr[\tilde{c}^{(2)} \text{is a valid codeword according to basic scheme}] \leqslant (1-\alpha)^{\tau/2} \leqslant \mathsf{negl}(\kappa),$$

   where $\alpha = \min_i \alpha_i$.

2. $n_{\mathsf{non-full}} > \tau/2$: We will analyse this case for a fixing of bits in $c^{(1)}_P$ which are copied to this block, the fixing of the randomness of the non-constant channels applied to these bits and a fixing of the outer codeword $d^{(\mathsf{out})}_{[n]}$ as shown above. The analysis will be over the randomness used to pick the inner codeword block given the outer code element in $d^{(\mathsf{out})}_{[n]}$.

   Consider a non-full block i. Let $f_{i,j}$ be the non-constant channel applied to $X_{i,j}$. Let $X'_{i,j} = f_{i,j}(X_{i,j})$. We fix the randomness for the channels $f_{i,j}$. This fixes the value $Z_i = \sum_j X'_{i,j} \mathsf{mod} 3$.

   Now consider the bits in this block which have been copied from different blocks of $c^{(2)}[N^{(2)}]$. Also, note that non-constant channels may have been applied to potentially all these bits. By repeated application of Lemma 11 on different blocks from which bits have been copied, we get that the parity mod 3 of this collection of bits is μ-unpredictable for some constant $\mu > 0$.

   Now combining the above two observations, we get that parity mod 3 of this block is μ-unpredictable. Since only parity 0 mod 3 blocks are valid, Pr[Block i is valid] $\leqslant (1 - \mu_i)$.

Note that this analysis can be done independently for each non-full block in $\langle \overline{Q} \rangle$. Using this observation, we get

$$\Pr[\tilde{c}^{(2)}\text{is a valid codeword according to basic scheme}] \leqslant (1-\mu_i)^{\tau/2} \leqslant \mathsf{negl}(\kappa),$$

where $\gamma = \min_i \mu_i$.

□

D.2 Proof of Case 2

Here we will prove the following lemma.

Lemma 7. *If* $\left|R' \setminus \langle \overline{Q} \rangle\right| > \tau$, $\Pr[\tilde{c}^{(2)}$*is a valid codeword according to basic scheme*$] \leqslant \mathsf{negl}(\kappa)$*, where the probability is over the randomness used to pick the codeword for the message* s*.*

Proof. We will analyse this case for a fixed outer codeword for the tag t. Let the outer codeword be $d^{(\mathsf{out})}_{[n]}$). Let the length of each inner code word block be m, i.e. $N^{(2)} = mn$. Let $c^{(2)}_{[N^{(2)}]} = c^{(2)}_{[mn]} \sim (\mathbf{X}^{(\mathsf{basic})}_{[mn]} | \mathbf{X}^{(\mathsf{out})}_{[n]} = d^{(\mathsf{out})}_{[n]})$. We also define $y^{(\mathsf{in})}_i = c^{(2)}_{(i-1)m+[m]}$. Note that $y^{(\mathsf{in})}_i$ refers to the i^{th} block of $c^{(2)}$ corresponding to i^{th} outer codeword element, i.e. $d^{(\mathsf{out})}_i$.

Here we will only consider tampering function correspoding to the blocks in $R' \setminus \langle \overline{Q} \rangle$. This tampering function would be a composition of some permutation of bits in R followed by non-constant channels $f_{i_1}, \dots, f_{i_z} \in \mathcal{C}_{\{0,1\}}$ for some $\mathcal{I} = \{i_1, \dots, i_z\}$ such that $\mathcal{I} \subseteq [nm]$. Only interesting channels to consider are non-message forwarding channels.

Given $\tilde{c}^{(2)}$, for all $i \in [n]$, define $z^{(\mathsf{in})}_i = \tilde{c}^{(2)}_{(i-1)m+[m]}$. Here $z^{(\mathsf{in})}_i$ refers to the i^{th} block $\tilde{c}^{(2)}$.

Recall that each block in $R' \setminus \langle \overline{Q} \rangle$ receives all its bits from $c^{(2)}_{[N^{(2)}]}$. Note that each block in $R' \setminus \langle \overline{Q} \rangle$ either a dirty matching or a dirty codeword. For completion, we refine these terms below.

Definition 13 (Dirty Matching)**.** *For* $i, j \in [n]$*, a matching* $(y^{(\mathsf{in})}_i, z^{(\mathsf{in})}_j)$ *is said to be dirty if* $\exists k \in ((j-1)m + [m])$ *such that* $k \in \mathcal{I}$*, where* $\mathcal{I} \subseteq [mn]$ *refers to the set of indices to which non-constant channels have been applied.*

Definition 14 (Dirty Codewords)**.** *For some* $j \in [n]$*,* $z^{(\mathsf{in})}_j$ *is said to be a dirty inner codeword if there is no* $i \in [n]$ *such that* $(y^{(\mathsf{in})}_i, z^{(\mathsf{in})}_j)$ *is a matching. In other words, there exists* $i, i' \in [n]$ *such that some bits have been copied from both* $z^{(\mathsf{in})}_i$ *and* $z^{(\mathsf{in})}_{i'}$ *to* $y^{(\mathsf{in})}_j$*.*

Since we have fixed the outer codeword $d^{(\mathsf{out})}_{[n]}$, our sample space is $\left(\mathbf{X}^{(\mathsf{in})}_{[nm]} \;\middle|\; \mathbf{X}^{(\mathsf{out})}_{[n]} = d^{(\mathsf{out})}_{[n]}\right)$.

Next, consider the bipartite graph $G = (V_L, V_R, W)$ corresponding to the dirty matchings and dirty codewords as described in Appendix B. We will restrict our analysis to $R' \setminus \langle \overline{Q} \rangle$. So, we pick $V_R = R' \setminus \langle \overline{Q} \rangle$ and V_L consists of the blocks in $c^{(2)}_{[N^{(2)}]}$.[9]

[9] For the sake of completeness of the degree of the edges in V_L, we can consider an additional sink node in V_R and all the additional edges are mapped to this sink node.

Let $\{C_i\}$ be the connected components of G. Now, we will consider each component separately. For a component, if it consists of a dirty matching $(y_i^{(\mathsf{in})}, z_j^{(\mathsf{in})})$, then by Lemma 10, there exists a constant $\mu_1 \in (0,1)$ such that the parity (mod 3) of $z_j^{(\mathsf{in})}$ is μ_1-unpredictable. Since all valid blocks have parity (0 mod 3), $z_j^{(\mathsf{in})}$ will be an invalid block with probability at least μ_1.

For a connected component which is not a matching, an analysis similar to Appendix B and using Lemma 11, will show that there exists a constant $\mu_2 \in (0,1)$ such that the probability that all the nodes on the right are valid is at most $\mu_2^{n_{\mathsf{dirty},i}}$, where $n_{\mathsf{dirty},i}$ is the number of dirty codewords in C_i.

Since we had already fixed the outer code elements, these success probabilities over each component are independent of each other. Hence, we get

$$\Pr[\tilde{c}^{(2)} \text{ is valid}] \leqslant \left(\prod_{i:C_i \text{ is a matching}} (1-\mu_1) \right) \left(\prod_{i:C_i \text{ is not a matching}} \mu_2^{n_{\mathsf{dirty,i}}} \right) \leqslant \mu^\tau \leqslant \mathsf{negl}(n),$$

where $\mu \in (0,1)$ is some constant and the last inequality follows by $\tau = \lg^2 n$.

□

E Application to Non-malleable Commitments

In their original paper, Dziembowski et. al. [DPW10] suggest that non-malleable codes can be useful in obtaining tamper-proof security as follows. Consider a cryptographic device viewed as a pair $\langle G, s\rangle$ where G is a public cryptographic algorithm, encoded in a tamper-proof hardware; for example G could be the signing algorithm of a digital signature scheme. String s represents the secret contents of the memory, such as the signing-key along with some other state. State s might evolve over time. An adversary attacking the system can issue output queries on (appropriate) inputs x of its choice, and learn the output of $G(s,x)$. In addition, it may also issue *tamper* queries, represented as functions from the allowed class of tampering-function $f \in \mathcal{F}$. In response, the state of the device is set to $s' \leftarrow f(s)$. The adversary can issue these queries in any order for an appropriate number of times; such an adversary is said to have *tamper-access* to the device. Dziembowski et. al. argue that if we encode the secret s using a non-malleable code secure against the class $\mathcal{F}$, then the resulting device, denoted $\langle \widetilde{G}, \widetilde{s}\rangle$ [10] becomes secure against such tampering attacks: any tampering function $f \in \mathcal{F}$ will either leave s unchanged or result in an invalid encoding.

Non-malleable *string* commitments. The above approach is also useful in the setting of non-malleable commitments [DDN91]. Specifically, we show that non-malleable codes can be combined with with non-malleable *bit* commitment schemes to obtain a *simple* and *efficient* constructions for non-malleable *string* commitment schemes.

For simplicity, first consider "idealized commitment" models in which commitments can be implemented via physical means such as locked-boxes, hardware tokens, or sealed envelopes. Concretely, to commit to a bit b, the committer "writes" the value of b inside a box, locks the box using a

[10] String $\widetilde{s}$ is the non-malleable codeword of s; algorithm $\widetilde{G}$ first decodes $\widetilde{s}$ to s and then applies G.

physical lock, and sends it over to the receiver; to open the commitment, it simply sends across the key for the lock. We assume that the box implements an "idealized commitment": it perfectly hides the bit when locked, and opens to the same bit b that was written when it was first locked. This scheme is also a "non-malleable" commitment: a man-in-the-middle adversary can either choose to forward the locked-box to an honest receiver, or it can abort.[11]

Using non-malleable codes (secure against permutation attacks) we can construct a *string* commitment scheme as follows. The resulting scheme will be *non-malleable* against the class of *rearranging* attacks in which the adversary is only allowed to rearrange or permute the order of locked boxes it receives.

To commit to a string x of length n, the committer first encodes x using a non-malleable code secure against *permutation* attacks to obtain the codeword w of length n'. It then sends n' locked boxes in the order $B_1, \ldots, B_{n'}$ where box B_i is a commitment to w_i for $i \in [n']$. This scheme remains non-malleable against any rearrangement of the boxes: any permutation of the boxes does not change the committed string x. Indeed, suppose that after receiving the boxes, man-in-the-middle chooses a permutation $\pi : [n'] \to [n']$ and applies them to the n' boxes to define its commitment $(B'_1, \ldots, B'_{n'})$. Let w' be the resulting codeword inside these boxes where w'_i is the bit inside box B'_i. By construction, for every i there exists a unique j such that $B'_i = B_{\pi(j)}$, and therefore $w'_i{=}w_{\pi(j)}$. Hence, w' is a permutation of the original codeword w. Since the non-malleable code is resistant to permutation attacks, both w' and w encode the *same* string x.

Instead of working in a model with physical boxes, one can also obtain a similar result in the framework of universal composition [Can00, Can01]. For example, consider the UC-style formulation of rerandomizable RCCA encryption by Prabhakaran and Rosulek [PR07] where, roughly speaking, the trusted third party returns a randomly chosen "pointer" or a "handle" to represent a ciphertext (or commitment). This pointer acts as an ideal commitment and can be used by various parties to refer to the committed value in a larger computation.

Construction in the plain model. We now show that the above approach also works in the plain model provided that we use appropriate bit commitment and non-malleable code schemes. We will demonstrate this via an example. We will use Naor's commitment based on *adaptive* PRGs and non-malleable codes secure against class $\mathcal{F}^*$, as explained below.

Let $G : \{0,1\}^n \to \{0,1\}^{3n}$ be a pseudorandom generator (PRG) where n is the security parameter. In Naor's scheme, the receiver sends a random string $r \in \{0,1\}^{3n}$; to commit to 0, the committer sends $z = G(s)$ and to commit to 1 it sends $z \oplus r$ where $\oplus$ denotes bitwise exclusive-or and $s \in \{0,1\}^n$ is a random seed. If the PRG is an *adaptive* PRG [12] then this scheme is a non-malleable bit commitment scheme. This is because, given access to the inversion oracle $\mathcal{O}_G$, A can learn the value it commits to on "right"; if the scheme were not non-malleable, it compromises the hiding of commitment on "left", contradicting the adaptive security of the PRG G.

We will stick to a slightly weaker definition of non-malleable commitments, called *non-malleability*

[11] The probability of abort is independent of the bit in the received box since the bit is information-theoretically hidden.

[12] Roughly speaking, following [PPV08, KMO10], G is said to be adaptively secure if no PPT adversary A can tell if $y = G(s)$ for a random $s \in \{0,1\}^n$ or y is uniform *even if* A has access to a special *inversion* oracle $\mathcal{O}_G$; on query a string z of length $3n$, the oracle tells whether z is in the range of G or not. A is not allowed to query the challenge string y.

w.r.t. replacement [Goy11, Wee10]. Roughly speaking, this is the same as usual definition of non-malleability except that whenever the man-in-the-middle A sends a commitment for which no valid decommitment exists, the definition considers A to have "admitted defeat." When this happens, the definition allows a "simulator" to *replace* the invalid value (denoted by the abort symbol $\bot$) by *any* arbitrary value of its choice (that helps maintain an indistinguishable distribution). It has been shown in [Goy11, Wee10] that this definition suffices for almost all applications of NM commitments that we know.

There are 4 ways in which A can "admit defeat" as above. These cases are listed below, and we say that A acts as a defeat channel on the received commitment:

(1) when A receives commitment to a bit 0 on left, it commits to 0 on right, but if it receives commitment to 1, it commits to $\bot$; we denote this by $\mathsf{defeat}_{0\to 0,1\to\bot}$.

(2) opposite of the first case, denoted by $\mathsf{defeat}_{1\to 1,0\to\bot}$.

(3) "toggle" variant of case (1) where A commits to 1 if it receives a commitment to 0 and $\bot$ in the other case; this is denoted by $\mathsf{defeat}_{0\to 1,1\to\bot}$.

(4) opposite of (3), denoted by: $\mathsf{defeat}_{1\to 0,0\to\bot}$.

We need a non-malleable code which, in addition to tolerating permutation attacks, can also handle attacks $\mathsf{set}, \mathsf{reset}, \mathsf{toggle}$, (duplicate) copy, and all four defeat attacks. More precisely, let $\mathcal{F}^*$ be a class of tampering functions where every function $f \in \mathcal{F}^*$ is fully specified by a string of n' actions where $\mathsf{action} \in \{\mathsf{set}, \mathsf{reset}\} \cup \{\mathsf{copy}_i\}_{i\in[n']} \cup \{\mathsf{toggle}_i\}_{i\in[n']} \cup \mathsf{defeat}$, where $\mathsf{defeat} := \{\mathsf{defeat}_{0\to 0,1\to\bot}, \mathsf{defeat}_{1\to 1,0\to\bot}, \mathsf{defeat}_{0\to 1,1\to\bot}, \mathsf{defeat}_{1\to 0,0\to\bot}\}$. On input a codeword $w \in \{0,1\}^{n'}$, the output $w' = f(w)$ corresponding to $f := \{\mathsf{action}_i\}_{i=1}^{n}$ is defined as follows. For every $i \in [n']$, if $\mathsf{action}_i = \mathsf{set}$, then $w'_i = 1$; if $\mathsf{action}_i = \mathsf{reset}$, then $w'_i = 0$; if $\mathsf{action}_i = \mathsf{copy}_j$, then $w'_i = w_j$; if $\mathsf{action}_i = \mathsf{toggle}_j$, then $w'_i = 1 - w_j$; finally, if $\mathsf{action}_i \in \mathsf{defeat}$ bit w'_i is defined to be either 0 or $\bot$, according to items (1)–(4) above, depending on the value of w_j and the type of the "defeat" action.

We note that in this class, a particular bit of input codeword, say w_j, may be copied into more than one location of the output codeword w'. In contrast, this duplication is not allowed in "permutation only" attacks. That is, the class of permutation attacks corresponds to f which contain copy_j action exactly *once* in their description for every $j \in [n]$; in addition, they do not contain $\mathsf{set}/\mathsf{reset}$.

To commit to a string x of length, say n, our commitment scheme first encodes x using a non-malleable code that is secure against $\mathcal{F}^*$. It then commits to each bit of the resulting codeword using Naor's bit commitment scheme (instantiated using an adaptive PRG G). These commitments can be done in parallel. The receiver accepts a string commitment as follows. Let (r_i, t_i) be the two messages of i-th bit commitment; by construction $t_i = z_i$ or $z_i \oplus r_i$ for some string z_i in the range of G. Note that z_i is always well defined for honestly generated commitments. Define set $S_i := \{z_i, z_i \oplus r_i\} = \{t_i, t_i \oplus r_i\}$. Then, the receiver accepts the commitment if for all distinct i, i' it holds that $S_i \cap S_{i'} = \emptyset$. It is easy to check that this holds with high probability for honestly created commitments.

We claim that the above scheme is a non-malleable string commitment scheme. We argue this in two parts. First, we show that the attack by adversary A translates to a tampering attack f on the underlying codeword for some $f \in \mathcal{F}^*$. That is, we show that if c is the commitment for codeword

w received by A on left, and $c' \leftarrow A(c)$ is the commitment sent by A on right then there exists a function $f \in \mathcal{F}^*$ such that c' is a commitment to a codeword $w' = f(w)$ except with negligible probability. In the second part, we show that the distribution of $w'_0 = f_0(w_0)$ is computationally indistinguishable from $w'_1 = f_1(w_1)$ where w_b is a codeword for one of the two message x_0, x_1 chosen a-priori by A. This will prove non-malleability of our string commitment scheme.

To prove the first claim, we show how to construct f given c, c' and access to the inversion oracle $\mathcal{O}_G$ (corresponding to the adaptive PRG G) without compromising the hiding property of left commitment c. Let $c = \{c_i\}_{i\in[n']} = \{(r_i, t_i)\}_{i\in[n']}$ be the commitment given to A on left and $c' = \{c'_j\}_{j\in[n']} = \{(r'_j, t'_j)\}_{j\in[n']}$. Our goal is to construct $f \in \mathcal{F}^*$ *without* violating the hiding of the commitments on left. To do this, we have to be careful to not query the oracle $\mathcal{O}_G$ on any value which might violate the hiding of commitments on left. Indeed, A might carefully select string r_i sent on left, or, t'_j sent on right which reveal useful information via the answers of $\mathcal{O}_G$. Therefore, we will ensure that we never query such strings. These strings of interest are: t'_j, $t'_j \oplus r'_j$, $t'_j \oplus r_i$, $t'_j \oplus r'_j \oplus r_i$ for every j and i in $[n']$.[13]

Recall that we defined set $S_i = \{t_i, t_i \oplus r_i\}$ for commitments on left; define $S'_j = \{t_j, t_j \oplus r_j\}$ analogously for commitments on right. Further recall that $S_i \cap S_{i'} = \emptyset$ for all distinct i, i' w.h.p. for honestly generated commitments, and the commitment on right is accepted if and only if $S'_j \cap S'_{j'} = \emptyset$ for all distinct j, j'. Observe that except with negligible probability, it holds that for every j there do not exist distinct indices i, i' (corresponding to left commitments) such that $S'_j \cap S_i \neq \emptyset$ *and* $S'_j \cap S_{i'} \neq \emptyset$. This is because S_i and $S_{i'}$ do not intersect, and therefore if the claim were false, we must have, w.l.o.g., $t'_j \in S_i$ and $t'_j \oplus r_j \in S_{i'}$; but the later can only happen with negligible probability since r_j and $t_{i'}$ are honestly generated.

This allows us to define the parent of every right commitment j as follows. If there exists an i-th on left such that $S_i \cap S'_j \neq \emptyset$, define $\mathsf{parent}(j) = i$. Note that by the argument above, for every j, if there exists a parent then such a parent is *unique* with high probability.

Given c', we now construct $f = \{\mathsf{action_j}\}_{j\in[n']}$ as follows. For every j on the right:

1. if $\mathsf{parent}(j)$ does not exist, query the oracle $\mathcal{O}_G$ on all strings in the set $X_j := S'_j \cup (\cup_i S_i \oplus r_j)$.[14] It is easy to check that the j-th commitment sent by A on right appears in X_j; let b_j be the bit committed to in this commitment. Since all strings in X_j are sent to $\mathcal{O}_G$, the value of b_j is also known. Define $\mathsf{action}_j = \mathsf{set}$ if $b_j = 0$ and reset otherwise. Observe that no strings in set X_j are likely to appear in any of the sets S_i on left, for all i, with high probability.
2. if $\mathsf{parent}(j) = i$, define action_j as follows.
   (a) IF $r_i = r'_j$ then: (1) if $t'_j = t_i$, define $\mathsf{action}_j = \mathsf{copy}_i$, (2) if $t'_j = t_i \oplus r_i$ let , define $\mathsf{action}_j = \mathsf{toggle}_i$;
   (b) ELSE: (in this case it will be one of the defeat channels as follows:) (1) if $t'_j = t_i$ then $\mathsf{action}_j = \mathsf{defeat}_{0\to 0, 1\to\perp}$, (2) if $t'_j \oplus r'_j = t_i \oplus r_i$ then $\mathsf{action}_j = \mathsf{defeat}_{0\to\perp, 1\to 1}$, (3) if $t'_j = t_i \oplus r_i$ then $\mathsf{action}_j = \mathsf{defeat}_{0\to\perp, 1\to 0}$, (4) if $t'_j \oplus r'_j = t_i$ then $\mathsf{action}_j = \mathsf{defeat}_{0\to 1, 1\to\perp}$.

[13] Of course, A is also free to commit to its own values by sending appropriate strings in/out of the range of G; however, such strings will be "easy cases": they will simply translate to *fix types of attacks* on the underlying codeword, as we describe shortly.

[14] $S_i \oplus r_j$ means the set where each element of S_i is XORed with r_j.

This completes the description of our f. Therefore, with the help of the inversion oracle, attack on the outer commitment has been translated to an attack on the inner codeword w. Further, note that in defining this attack $f \in \mathcal{F}^*$, no string that belong to any of the left sets S_i for all i were queried to the inversion oracle. Therefore, it is possible to learn f with the help of the oracle *without* compromising the hiding of left commitment to w. Therefore, it holds that for every two messages x_0, x_1, if we let w_0, w_1 their respective non-malleable codewords sampled uniformly, c_0, c_1 honestly generated commitments to w_o, w_1 respectively, $c'_b \leftarrow A(c_b)$ the commitment produced by A on input c_b for $b \in \{0, 1\}$, and $f_b \in \mathcal{F}^*$ the tampering functions corresponding to c'_b; then by the adaptive security of G it holds that:

$$f_0 \;\approx_c\; f_1$$

Using this observation, we can design a simple hybrid argument to show that $f_0(w_0) \;\approx_c\; f_1(w_1)$. To see this, define the following $\mathsf{Game}_{a,b}$ for bits a, b: the game samples codewords w_0 and w_1 uniformly corresponding to messages x_0 and x_1 respectively; the game then samples a commitment c_a to the codeword w_a and gives it to A who produces a commitment c'. Let f_a be the tampering function w.r.t. c' and c_a (defined according to the procedure above). Function f is applied to w_b to yield $w' = f_a(w_b)$. The game decodes w' and outputs the resulting message, denoted $X_{a,b}$.

Note that $X_{0,0}$ and $X_{1,1}$ are identical to $f_0(w_0)$ and $f_1(w_1)$ respectively (defined above), and we want to show that $X_{0,0} \approx_c X_{1,1}$. First, observe that due to computational hiding of commitments, $X_{0,0} \approx_c X_{1,0}$ and $X_{1,1} \approx_c X_{0,1}$. We now argue that $X_{1,0} \approx_c X_{0,1}$ to conclude the argument.

Towards this end, recall that f_0, f_1 are well defined distribution over tampering functions and that $f_0 \approx_c f_1$. Let $Y_{0,0}$ be the output of the following experiment: the experiment samples f_0; it then samples a codeword w_0^* *independently* of f_0, and then outputs the decoding of the codeword $f_0(w_0^*)$. By computational hiding of the commitment scheme, we have that $X_{1,0} \approx_c Y_{0,0}$. Further, from the *non-malleability* of our code against functions in $\mathcal{F}^*$, it follows that $Y_{0,0} \approx_c X_{0,1}$. Therefore: $X_{1,0} \approx_c Y_{0,0} \approx_c X_{0,1}$. This completes the sketch of our proof.

F Mathematical Tools

In this section we prove some useful mathematical tools relevant for our results.

Lemma 8 (Density Lemma [Chv79])**.** *Let $c \in (0, 1)$ be a constant, $m, n \in \mathbb{N}$ and $m \in [cn, (1-c)n]$. Let $\mathbf{X}_{[n]} = \mathbf{U}_{\binom{[n]}{m}}$. For every $t \in \mathbb{N}$, we have:*

$$\Pr_{x_{[n]} \sim \mathbf{X}_{[n]}}\left(\sum_{i \in [t]} X_i = t\left(\frac{m}{n} \pm \varepsilon\right)\right) \leqslant 2\exp\left(-\mathrm{D}_{\mathrm{KL}}\left(\frac{m}{n} + \varepsilon, \frac{m}{n}\right) \cdot t\right) \leqslant 2\exp(-\varepsilon^2 t/3),$$

where $\mathrm{D}_{\mathrm{KL}}(\alpha, \beta) := \alpha \ln \frac{\alpha}{\beta} + (1-\alpha)\ln\frac{1-\alpha}{1-\beta}$. *In particular:*

$$\Pr_{x_{[n]} \sim \mathbf{X}_{[n]}}\left(\sum_{i \in [t]} X_i \in [(c-\varepsilon)t, (1-c-\varepsilon)t]\right) \leqslant 2\exp(-\varepsilon^2 t/3)$$

Lemma 9 (Unpredictability Lemma)**.** *Let $c \in (0,1)$ be a constant, $m, n \in \mathbb{N}$ and $m \in [cn, (1-c)n]$. Define $\mathbf{X}_{[n]} = \mathbf{U}_{\binom{[n]}{m}}$. Let $p \in \mathbb{N}$ be a constant. Given any $t \in [n-1]$, let $\mathsf{parity}_{n,m,t,p}$ be the random variable: $\sum_{i \in [t]} X_i \mod p$. Then, there exists a constant $\mu \in (0, c)$ such that $\mathsf{parity}_{n,m,t,p}$ is μ-unpredictable.*

Proof. We consider two cases.

Case $(n - t + 1) \geqslant n/2$. A random sample of $\mathbf{X}_{[n]}$, satisfies the following condition with $1 - \nu(n)$ probability (where, $\nu(n) = \mathsf{negl}(n)$): The random variable $\mathbf{X}_{[n] \setminus [t-1]}$ is $(c-\varepsilon)$-dense and $(1-c+\varepsilon)$-sparse (by Lemma 8), for any constant $\varepsilon \in (0,1)$. Lets call this a good event. This implies that the random variable $\mathbf{X}_t$ is $(c-\varepsilon)$-balanced. Therefore, conditioned on a good event, $\mathsf{parity}_{n,m,t,p}$ is $(c-\varepsilon)$-unpredictable.

Since there are $\binom{p}{2}$ pairs of parity values, by an averaging argument, $\mathsf{parity}_{n,m,t,p}$ is $(c-\varepsilon)(1-\nu(n))\binom{p}{2}^{-1}$-unpredictable. Any constant $\mu < (c-\varepsilon)\binom{p}{2}^{-1}$ suffices.

Case $(n - t + 1) < n/2$. This implies that $t - 1 > n/2$. With $1 - \nu(n)$ probability, where $\nu(n) = \mathsf{negl}(n)$, $\mathbf{X}_{[t+1]}$ is $(c-\varepsilon)$-dense and $(1-c+\varepsilon)$-sparse (by Lemma 8), for any constant $\varepsilon \in (0,1)$. Lets call this a good event. This implies that the random variable $\mathbf{X}_{t+1}$ is $(c-\varepsilon)$-balanced. Therefore, conditioned on a good event, parity of last $(n-t)$ bits is $(c-\varepsilon)$-unpredictable. Which implies that conditioned on a good event, $\mathsf{parity}_{n,m,t,p}$ is $(c-\varepsilon)$-unpredictable.

Consequently, $\mathsf{parity}_{n,m,t,p}$ is $(c-\varepsilon)(1-\nu(n))\binom{p}{2}^{-1}$-unpredictable. Any constant $\mu < (c-\varepsilon)\binom{p}{2}^{-1}$ suffices. □

Lemma 10 (Unpredictability of Dirty Matchings)**.** *Let $c \in (0,1)$ be a constant, $m, n \in \mathbb{N}$ and $m \in [cn, (1-c)n]$. Define $\mathbf{X}_{[n]} = \mathbf{U}_{\binom{[n]}{m}}$. Consider non-constant channels $f_{i_1}, \dots, f_{i_z}$, $\mathcal{I} = \{i_1, \dots, i_z\} \subseteq [n]$. Define $\mathbf{Y}$ as follows: For all $i \in [n]$, define $Y_i = f_i(X_i)$ if $i \in \mathcal{I}$, else $Y_i = X_i$. Consider the random variable $\mathbf{P}_Y = \sum_{i \in [n]} Y_i \mod 3$. Then there exists a constant $\mu \in (0,1)$ such that $\mathbf{P}_Y$ is μ-unpredictable.*

Proof. A channel f is "confusing" if there exists $b \in \{0,1\}$ such that $\mathsf{Supp}(f(b)) = \{0,1\}$.

Case 1. There exists a confusing channel in $\{f_{i_1}, \dots, f_{i_z}\}$. Without loss of generality assume that f_1 is confusing and $\mathsf{Supp}(f(0)) = \{0,1\}$. We have $\Pr[X_1 = 0] \in [c, 1-c]$. Fix a setting of $\mathbf{X}$ conditioned on $X_1 = 0$. Fix the internal randomness of all remaining channels. This fixes $Y_2 + \cdots + Y_n \mod 3$. Now, we have $\Pr(Y_1 = 0)$ and $\Pr(Y_1 = 1)$ are at least a constant conditioned on these fixings. So, overall we can set $\mu = \Pr(X_1 = 0) \cdot \min\{\Pr[Y_1 = 0 | X_1 = 0], \Pr[Y_1 = 1 | X_1 = 0]\}$.

Case 2. There are no confusing channels. This implies that all channels are "toggle" channels. Without loss of generality assume that $\{i_1, \dots, i_z\} = [z]$. If f_i is a toggle then note that $Y_i = 1 - X_i$ over $\mathbb{Z}_3$. So,

$$\begin{aligned} Y_1 + \dots + Y_n \mod 3 &= z - (X_1 + \dots + X_z) + (X_{z+1} + \dots + X_n) \mod 3 \\ &= z + m - 2(X_1 + \dots + X_z) \mod 3 \\ &= z + m + (X_1 + \dots + X_z) \mod 3 \end{aligned}$$

Note that $X_1 + \dots + X_z$ is μ unpredictable, for some constant μ Lemma 9. Hence, $Y_1 + \dots + Y_n$ mod 3 is μ unpredictable. □

Lemma 11 (Unpredictability w.r.t. Channels)**.** *Let $c \in (0,1)$ be a constant, $m, n \in \mathbb{N}$ and $m \in [cn, (1-c)n]$. Define $\mathbf{X}_{[n]} = \mathbf{U}_{\binom{[n]}{m}}$. For any $t \in [n-1]$, let $f_1, \dots, f_t$ be non-constant channels; and define $Y_i = f_i(X_i)$, for all $i \in [t]$. Let $\mathsf{parity}_{n,m,t,3}$ be the random variable: $\sum_{i \in [t]} Y_i \mod 3$. Then, there exists a constant $\mu \in (0,c)$ such that $\mathsf{parity}_{n,m,t,3}$ is μ-unpredictable.*

Further, for $t = n$ and $(n-m) \neq 0 \mod 3$, if there exists f_i such that it is not the identity mapping (i.e. it is not $f(b) = b$), then $\Pr[\mathsf{parity}_{n,m,t,3} \neq 0]$ is at least a constant.

Proof. When $t \in [n-1]$, the proof follows by combining the proofs of Lemma 9 and Lemma 10. We just show the proof for the case $(n - t + 1) \geqslant n/2$. The final case $(n - t + 1) < n/2$ follows analogously.

A channel f is "confusing" if there exists $b \in \{0,1\}$ such that $\mathsf{Supp}(f(b)) = \{0,1\}$.

Case 1. Suppose there exists a confusing channel in $\{f_1, \dots, f_t\}$. Without loss of generality assume that f_t is confusing and $\mathsf{Supp}(f(0)) = \{0,1\}$.

We know that with probability $1 - \mathsf{negl}(n)$, the random variable $\mathbf{X}_{[n] \setminus [t-1]}$ is $(c-\varepsilon)$-dense and $(1-c+\varepsilon)$-sparse (by Lemma 8), for any constant $\varepsilon \in (0,1)$. Let us call this a good event. This implies that the random variable $\mathbf{X}_t$ is $(c-\varepsilon)$ balanced. Therefore, conditioned on a good event $\Pr[\mathbf{X}_t = 0] \geqslant (c - \varepsilon)$.

Conditioned on $\mathbf{X}_t = 0$ and the good event, choose a fixing of $\mathbf{X}_{[n]}$ and also fix the internal randomness of all channels $\{f_1, \dots, f_{t-1}\}$. Now, it is clear that $\mathsf{parity}_{n,m,t,3}$ is at least $(c-\varepsilon) \cdot \min\{\Pr[Y_t = 0 | X_t = 0], \Pr[Y_t = 1 | X_t = 0]\} - \mathsf{negl}(n)$.

Case 2. Suppose there are no confusing channels; that is all channels are toggle channels. Again we condition on the good event mentioned above and mimic the proof of Lemma 10 for the corresponding case.

Let us consider the case of $t = n$. Suppose there exists a confusing channel. Without loss of generality assume that f_1 is a confusing channel with $\mathsf{Supp}(f(0)) = \{0,1\}$. Then we know that $\Pr[X_1 = 0] \in [c, 1-c]$. By fixing a choice of $X_{2,\dots,n}$ and internal randomness of $f_2, \dots, f_t$ we get that $\mathsf{parity}_{n,m,t,3}$ is constant unpredictable.

Suppose all channels are toggles. In this case, we use the fact that $Y_i = 1 - X_i$, for each $i \in [t]$.

Now we have:

$$\begin{aligned}\sum_{i\in[n]} Y_i \mod 3 &= \sum_{i\in[t]} 1 - X_i \mod 3\\ &= (n-m) \mod 3\end{aligned}$$

□

G Algebraic Geometric Codes Parameters

This section contains explicit AG code constructions over $\mathbb{F}_q$ and constructing secret sharing schemes based on it (similar to Massey's secret sharing [Mas95]). These results are roughly based on [CC06]. For completeness, we include these results.

Secret Sharing Scheme $\mathbf{X}^{(\mathsf{AG},n,k,d,\ell,t,\mathbb{F}_q)}$:

1. Sample space: $\Lambda_0 = \mathbb{F}_q^\ell$, $\Lambda_1 = \cdots = \Lambda_{n-\ell} = \mathbb{F}_q$.
2. Conditions: Let $\alpha, \beta, \alpha', \delta, q$ be constants such that:
   (a) $k = \alpha n$, $d = (1-\alpha-\beta)n$, $\ell = \alpha' n$ and $t = \delta n$.
   (b) $\alpha \in (0,1)$ and $\beta \in (0, 1-\alpha)$.
   (c) $\alpha' \in (0,\alpha)$ and $\delta \in (0, \alpha - \alpha')$.
   (d) $q \geqslant \max\left\{\left(1+\frac{1}{\beta}\right)^2, \left(1+\frac{1}{\alpha-\alpha'-\delta}\right)^2, \left(1+\frac{1}{1-\alpha-\alpha'}\right)^2, 49\right\}$ and q is an even power of a prime.
3. Joint Distribution $(\mathbf{X}_0, \ldots, \mathbf{X}_{n-\ell})$ is defined via the following sampling procedure:
   (a) Consider the $[n,k,d]_q$ AG code over $\mathbb{F}_q$ as described in Theorem 3.
   (b) Sample a random AG codeword $(x_{-\ell}, \ldots, x_{-1}, x_1, \ldots, x_{n-\ell})$.
   (c) Define $x_0 = (x_{-\ell}, \ldots, x_{-1}) \in \mathbb{F}_q^\ell$.
   (d) Output $(x_0, x_{[n-\ell]})$.

Efficient Encoding and Decoding. Follows from [CC06]. The scheme is t-private and $n\left(\alpha + \frac{1}{\sqrt{q}-1}\right)$-reconstructible.

Figure 9: Algebraic-Geometric Code based Secret Sharing.

Theorem 3 (Asymptotic AG Code Parameter Choices [CC06])**.** *For every n, $k = \alpha n$, $d = (1-\alpha-\beta)n$, $\ell = \alpha' n$, $t = \delta n$ and q such that:*

1. α, β, α' and δ are constants.

2. $\alpha \in (0,1)$, $\beta \in (0, 1-\alpha)$.

3. *$\alpha' \in (0, \alpha)$ and $\delta \in (0, \alpha - \alpha')$*

Then, there exists a constant q^* such that, for all $q \geqslant q^*$ and q is an even power of a prime, there exists an $[n, k, d]_q$ linear code over $\mathbb{F}_q$.

Further, using the share-packing technique in Figure 9, it is possible to pack $\mathbb{F}_q^\ell$ such that reconstruction is possible while still ensuring t-privacy.

Proof. We choose q as an even prime power such that:

$$q \geqslant q^* := \max\left\{\left(1 + \frac{1}{\beta}\right)^2, \left(1 + \frac{1}{\alpha - \alpha' - \delta}\right)^2, \left(1 + \frac{1}{1 - \alpha - \alpha'}\right)^2, 49\right\}$$

Let $\{C_m\}_{m \in \mathbb{N}}$ be the infinite family of Garcia and Stichtenoth curves [GS96]. For any curve C_m we have:

1. The number of $\mathbb{F}_q$-rational points on C_m is: $\#C_m(\mathbb{F}_q) \geqslant (\sqrt{q} - 1)q^{m/2} =: n$, and
2. The genus of the curve C_m is: $g = g(C_m) \leqslant q^{m/2}$.

Goppa codes [Gop81, CC06] using the projective non-singular curve C_m over $\mathbb{F}_q$ and divisor D on C_m such that $\deg D = \alpha n + (q^{m/2} - 1)$, provide $[n, k, d]_q$ linear codes, such that:

1. $k = \deg D + 1 - g \geqslant \alpha n$, and
2. $d \geqslant n - \deg D = n - k - g + 1 = n - \alpha n - \beta n + 1 > n(1 - \alpha - \beta)$.

This code is $(\deg D + 1)$-reconstructible.

We use the share packing technique of [CC06] and pack $\mathbb{F}_q^\ell$. So, we get $(k - \ell - g)$-privacy for this scheme [CC06].

1. Privacy. Note that we have:

$$\begin{aligned}(k - \ell - g) &\geqslant \alpha n - \alpha' n - q^{m/2}\\ &= \alpha n - \alpha' n - n/(\sqrt{q} - 1)\\ &\geqslant n[\alpha - \alpha' - (\alpha - \alpha' - \delta)]\\ &= \delta n\end{aligned}$$

   So, we have δn-privacy.

2. Reconstruction. Note that we have $\deg D + 1$ reconstruction and, therefore, we must have: $(n - \ell) \geqslant (\deg D + 1)$.

$$\begin{aligned}(n - \ell) - (\deg D + 1) &= n - \alpha' n - \alpha n - q^{m/2}\\ &= n(1 - \alpha - \alpha') - \frac{n}{\sqrt{q} - 1}\\ &\geqslant n(1 - \alpha - \alpha') - n(1 - \alpha - \alpha') = 0\end{aligned}$$

The generator matrix can be efficiently constructed [SAK+01], when $q \geqslant 49$ is an even power of a prime. Efficient decoding up to half the distance is provided by [O'S95]. □