Four-state Non-malleable Codes with Explicit Constant Rate

Bhavana Kanukurthi * Sai Lakshmi Bhavana Obbattu ** Sruthi Sekar***

Indian Institute Of Science, Bangalore

Abstract. Non-malleable codes (NMCs), introduced by Dziembowski, Pietrzak and Wichs (ITCS 2010), generalize the classical notion of error correcting codes by providing a powerful guarantee even in scenarios where error correcting codes cannot provide any guarantee: a decoded message is either the same or completely independent of the underlying message, regardless of the number of errors introduced into the codeword. Informally, NMCs are defined with respect to a family of tampering functions $\mathcal{F}$ and guarantee that any tampered codeword either decodes to the same message or to an independent message, so long as it is tampered using a function $f \in \mathcal{F}$.

Nearly all known constructions of NMCs are for the t-split-state family, where the adversary tampers each of the t *states* of a codeword, arbitrarily but independently. Cheraghchi and Guruswami (TCC 2014) obtain a Rate-1 non-malleable code for the case where $t = \mathcal{O}(n)$ with n being the codeword length and, in (ITCS 2014), show an upper bound of $1 - 1/t$ on the best achievable rate for any $t-$split state NMC. For $t = 10$, Chattopadhyay and Zuckerman (FOCS 2014) achieve a constant rate construction where the constant is unknown. In summary, there is no known construction of an NMC with an explicit constant rate for any $t = o(n)$, let alone one that comes close to matching Cheraghchi and Guruswami's lowerbound!

In this work, we construct an efficient non-malleable code in the t-split-state model, for $t = 4$, that achieves a constant rate of $\frac{1}{3+\zeta}$, for any constant $\zeta > 0$, and error $2^{-\Omega(\ell/log^{c+1}\ell)}$, where ℓ is the length of the message and $c > 0$ is a constant.

* Department of Computer Science and Automation, Indian Institute Of Science, Email: bhavana@iisc.ac.in. Research supported, in part, by Department of Science and Technology Inspire Faculty Award.

** Department of Computer Science and Automation, Indian Institute Of Science, Email: oslbhavana@gmail.com

*** Department of Mathematics, Indian Institute Of Science, Email: sruthi.sekar1@gmail.com.

1 Introduction

Error correcting codes allow for the correction of errors introduced in data. However, their applicability is limited by the fact that they can only correct a bounded number of errors. When data is completely overwritten, no protection can be guaranteed. Non-malleable codes, introduced in the work of Dziembowski, Pietrzak and Wichs [16], guarantee that, errors caused to the data will render it either independent of the underlying message or leave it unchanged.

Non-malleable codes are parameterized by a family of *tampering* functions, $\mathcal{F}$, and they guarantee non-malleability only when $m^* = \mathsf{Dec}(f(\mathsf{Enc}(m)))$ where $f \in \mathcal{F}$ and $\mathsf{Enc}, \mathsf{Dec}$ are the encode and decode functions respectively. (In other words, there is no guarantee when $f \notin \mathcal{F}$.) Informally, given a tampering family $\mathcal{F}$, a non-malleable code $(\mathsf{Enc}, \mathsf{Dec})$ encodes a given message m into a codeword $c \leftarrow \mathsf{Enc}(m)$ s.t. if c is modified to $\tilde{c} = f(c)$ by some $f \in \mathcal{F}$, then the message $\tilde{m} = \mathsf{Dec}(\tilde{c})$ contained in the modified codeword, is either the original message m or is "unrelated" to and "independent" of m.

To understand the motivation of studying non-malleable codes, consider their application to cryptography. In any standard cryptographic security game, security is typically guaranteed even when the adversary has access to some permissible input-output behaviour on the secret key sk. [1] If the adversary is allowed to observe input-output behaviour with respect to some modified sk^*, we can no longer guarantee *any* security with respect to the original key sk. Consider a situation where sk^*, if different from sk, is guaranteed to be independent of sk. In such a case, no input-output behaviour on sk^* can help break the security with respect to sk. (After all, if obtaining information about an independent sk^* can help break the security with respect to sk, then an adversary for sk can generate this information on his own.) If sk is encoded with a non-malleable code, then non-malleability will prevent sk^* from ever taking a related value and the scheme will continue to remain secure with respect to sk.

It is no surprise that, since their introduction, non-malleable codes have found many applications such as in securing functionalities against physical –leakage and tampering– attacks [16,23], domain extension of CCA secure encryption [9] and non-malleable commitments [18]. Additionally, non-malleable codes have inspired an impressive line of theoretical research drawing connections across topics such as non-malleable extractors, additive combinatorics and so on. Researchers have been fascinated with two aspects of non-malleable codes:
(a.) the richness of the tampering function family which NMCs can protect against and
(b.) the *rate* $(= \frac{\mathsf{message\ length}}{\mathsf{codeword\ length}})$ they achieve.

Our work too falls into this domain with a specific focus on the rate.

[1] For example, this input-output behaviour may be decryption of ciphertexts in the case of Chosen Ciphertext Security of Encryption or signatures of messages in the case of Digital Signatures.

1.1 Related Work

In [16], Dziembowski et al. observe that it is impossible to build non-malleable codes which are secure with respect to the class of *all* functions. The intuition behind this is that, this class would contain the function which decodes $\mathsf{Enc}(m)$ and re-encodes it into a related value m^*. Further, [16] proves an existential result for non-malleable codes w.r.t tampering families of size less than 2^{2^n}.

A natural but restricted class of tampering functions is the class of *bit-wise tampering functions* which modify each bit of the codeword independently. Dziembowski et al. [16] presented a construction of non-malleable codes with respect to this family. Their construction used a composition of Linear error correcting secret sharing scheme (LECSS)[2] and Algebraic Manipulation Detection codes (AMD codes)[3]. Following this, Cheraghchi and Guruswami [8] gave an explicit construction of an *optimal rate* NMC w.r.t. bit-wise tampering family. Their construction combines the properties of a LECSS scheme, a sub-optimal NMC for small messages and pseudorandom permutations.

A natural generalization of the bit-wise tampering family is the split-state tampering family, where a codeword is split into blocks (typically of equal length though not necessarily) and each block of the codeword, called a *state*, is tampered independently. A $t-$split-state family consists of a family of t functions acting independently on a state of length n/t, where n is the codeword length [4].

Improving on an existential result due to Dziembowski et al. [16], in [7], Cheraghchi and Guruswami show that for a t-split state family, with each state of codeword containing n/t bits, the upper bound on best achievable rate is $1-1/t$. Both [16] and [7] give a Monte-Carlo construction of non-malleable codes for the 2-split-state model which show the existence of such codes in the random oracle model. The work of [8] also makes an elegant connection between seedless t-source non-malleable extractors and non-malleable codes in the t-split-state model.

In spite of the progress on bit-wise tampering function family, the first efficient constructions of split-state non-malleable codes made strong assumptions such as the random oracle model [16] or were in the computational setting [23][5].

[2] LECSS ensures that the bits of codeword are t-wise independent and detects tampering if the codeword is modified by an offset Δ, when Δ is not a valid codeword of the scheme.

[3] AMD codes detect tampering attacks that add some pre-determined offset Δ to the codeword.

[4] This tampering family captures other tampering attacks such as bit-wise tampering, identity function, constant function etc. A motivation to study this model comes from practical applications like cloud storage, where a single file may be stored in t parts at t different locations and an adversary tampers each of these parts independent of the other. It is therefore both of theoretical as well as practical interest to obtain non-malleable codes for $t-$split state family where $t > 1$ is as small as possible.

[5] Specifically, Liu and Lysyanskaya [23] present a computational non-malleable code w.r.t. split-state tampering functions in the common reference string (CRS) model, using number theoretic assumptions and assuming existence of robust non-interactive zero-knowledge proof systems for an appropriate NP language.

Dziembowski, Kazana and Obremski [15] were the first to present an explicit construction of a non-malleable code for the split-state model. Specifically, they used the inner product extractor to construct a non-malleable code for 1-bit messages in the 2-split-state model. Improving upon this result, Aggarwal, Dodis and Lovett [3] gave the first information theoretic construction for k-bit messages in 2-split-state model, achieving rate $\Omega(n^{-6/7})$. This construction relies on an elegant property of inner-product functions, which is obtained using results from additive combinatorics, including the *Quasi-polynomial Freiman-Ruzsa Theorem.*

NMC with improved rates: All of the above works focused on improving the richness of tampering functions which NMCs can tolerate. However, none of them, barring the codes of [8] for the bit-wise tampering family, achieve optimal rate. Chattopadhyay and Zuckerman [6] were the first to construct an efficient constant rate non-malleable code in 10-split-state model. Unfortunately, the rate they achieve is an *unknown* constant which is typically undesirable while building information-theoretic primitives. Additionally, as observed in [2], the rate is likely to be a small (i.e., poor) constant due to their use of additive combinatorics.

For the 2-split-state model, the construction by Li in [22] achieves the best known rate to date, of $\Omega(1/\log n)$. Both these works use the connection between seedless t-source non-malleable extractors and non-malleable codes in t-split-state model, due to [8]. The work of Aggarwal et al. [2] gives beautiful connections between various split-state models. Unfortunately, due to a subtle error pointed by Li [22], their proposed construction of a 2-split state, constant rate, non-malleable code no longer holds, making Li's result the best known for the 2-split state model. However, there are two *conjectured* constant-rate NMC constructions. Specifically, in [3], under an inner product conjecture, the authors get a constant-rate 2-split-state scheme. Further, while [2], as it stands, gives a linear-rate code using existing methods, it gives a constant-rate 2-split-state under an appropriate conjecture.

We know the following, about the best achievable rate, from [7] :

Lemma 1 (Section 1.1,[7]). *For non-malleable codes in the t-split-state model, with each state of equal length, the best achievable rate is $1 - \frac{1}{t}$.*

While Cheraghchi et al. in [8], obtain a Rate-1 (optimal) NMC for $t = \mathcal{O}(n)$, there is no known construction for $t = o(n)$, which achieves the optimal rate $1 - \frac{1}{t}$, for t-split-state family. In this work, we construct a non-malleable code with rate $\frac{1}{3+\zeta}$, for any constant $\zeta > 0$ in the 4-split-state model.

Computational setting: If we resort to computational assumptions, Aggarwal et al.[1] show that a NMC with the best possible rate as well as the least restricted of the t-split-state families can obtained. Concretely, they obtain a rate 1 computational NMC w.r.t. 2-split-state tampering function family. Unfortunately, despite significant efforts, there has been a large gap between the rates of the best known constructions in the computational setting and the information-theoretic setting.

We give an overview of the Rate-1 construction in the computational setting due to Aggrawal et al. [1] and then highlight the challenges of building such codes in the information-theoretic setting. Their construction works by choosing a key k_{ae} to a computational authenticated encryption scheme. It encodes this key with a poor-rate 2-split-state non-malleable code to get states c_1, c_2. This key is used to compute an authenticated encryption ciphertext (c_3) of the message to be encoded. This gives a three state nmc: (c_1, c_2, c_3). (They obtain a two-split-state construction by using an enhanced notion of "*augmented*" non-malleable codes. They also prove that the 2-state construction of [3] achieves augmented non-malleability.) The key behind the optimality of the rate is the observation that the length of the key for authenticated encryption (in the computational setting) can be short (and independent of the message length).
We have summarized the prior work on NMCs for the t-split-state model in Figure 1.

Work by	**No. of states**	**Rate**
Dziembowski, Pietrzak and Wichs (ICS 2010) **Introduced NMCs**	n	• Existential result for family of size $\log(\log(\lvert\mathcal{F}\rvert)) < n$ • Rate < 1 (explicit const.)
Cheraghchi and Guruswami (TCC 2014)	n	$1 - o(1)$ (optimal rate)
Cheraghchi and Guruswami (ITCS 2014)	t	$1 - 1/t$ (Existential, optimal achievable rate)
Dziembowski, Kazana, Obremski (CRYPTO 2013)	2	$\Omega(n^{-1})$ (for 1-bit messages)
Aggarwal, Dodis and Lovett (STOC 2014)	2	$\Omega(n^{-6/7})$
Chattopadhyay and Zuckerman (FOCS 2014)	10	$\Omega(1)$ (explicit constant not given)
Li (STOC 2017)	2	$\Omega(1/\log\ n)$
Aggarwal, Agrawal, Gupta, Maji, Pandey and Prabhakaran (TCC 2016) **Computational NMC**	2	$1 - o(1)$ (optimal rate computational NMC)
OUR RESULT (Kanukurthi, Obbattu, Sekar)	**4**	**1/3 (first explicit constant rate)**

Fig. 1: Summarizing prior work on t-split-state family

1.2 Our Result

Informally, in this work, we obtain information-theoretic constant-rate non-malleable codes in the 4-split-state model. The fact that we make no compu-

tational assumptions brings up some unique challenges in both the construction as well as the proof, which we now highlight. As a starting point, consider the same construction [1] described above but replace the computational authenticated encryption scheme with an information-theoretic one: we would still obtain a secure non-malleable code. However, for an information-theoretic encryption scheme to be secure, we require the length of the key to be as much as the length of the message. This means that to obtain good rate, the split-state non-malleable code used as a building block should have good rate as it is encoding a key that is as long as the message – this is precisely the problem we are trying to solve!

To resolve this chicken-and-egg problem, we observe that an authenticated encryption scheme can be modularly decomposed into an authentication scheme and an encryption scheme: namely, encrypting a message first with a generic (one-time) encryption scheme and then authenticating it with a one-time message authentication code, gives us a construction of an (one-time) authenticated encryption scheme. The good news is that message authentication codes only require short keys (informally, as long as the security parameter) and can, therefore, be encoded using a non-malleable code without compromising on the rate. This leads to the following approach: *can we leverage the non-malleability of authentication key to non-malleably encode larger messages?*

We shall motivate our construction by discussing some incorrect constructions. Consider the following attempt: $c_1 = (\mathsf{Enc}_{k_e}(m), \mathsf{Tag}_{k_a}(\mathsf{Enc}_{k_e}(m)))$; $c_2 = k_e$; $(c_3, c_4) = \mathsf{NMEnc}(k_a)$ where Enc is just a one-time pad encryption, $\mathsf{MAC} = (\mathsf{Tag}, \mathsf{Vrfy})$ is a one-time message authentication code, NMEnc is a 2-split-state non-malleable code and $\{c_i\}_{i\in[4]}$ are all stored in separate states. A fundamental problem with this proposal is that the encryption key is not encoded with a non-malleable code. By simply changing the key k_e and leaving the rest of the encoding unchanged, the adversary can relate the tampered message $\tilde{m}$ to the underlying message m. We can fix this problem by requiring the encryption key to be authenticated as well. Let $c_1 = (\mathsf{Enc}_{k_e}(m), \mathsf{Tag}_{k_a}(\mathsf{Enc}_{k_e}(m)||k_e))$; $c_2 = k_e$; $(c_3, c_4) = \mathsf{NMEnc}(k_a)$. While the authenticity of k_e may no longer be an issue, this introduces another problem: c_1 contains a MAC value computed on the key k_e and could reveal some information about k_e and therefore, the ciphertext c_1 may no longer be secure. The standard definition of a one-time MAC does not guarantee privacy of the underlying message. (We could consider specific MACs which do guarantee privacy as well but such information-theoretic MACs cannot have short keys, which we require, as mentioned above.) Let us try to remove this dependency by encoding the tag using the non-malleable code. Let $c_1 = (\mathsf{Enc}_{k_e}(m))$; $c_2 = k_e$; $(c_3, c_4) = \mathsf{NMEnc}(k_a, \mathsf{Tag}_{k_a}(\mathsf{Enc}_{k_e}(m)||k_e))$.
This leads to the following candidate construction to encode a message m:

1. Choose a key k_e for one-time pad encryption (Enc) and a key k_a for a one-time message authentication code (MAC).
2. Compute $c_1 = \mathsf{Enc}_{k_e}(m)$, tag $t = \mathsf{Tag}_{k_a}(c_1||k_e)$ and set $c_2 = k_e$.
3. Compute $(c_3, c_4) \leftarrow \mathsf{NMEnc}(k_a, t)$, using a 2-split-state non-malleable code with poor rate.

4. Output c_1, c_2, c_3, c_4 as the four states of the non-malleable code.

Intuitively, this might seem secure as the encryption key k_e is authenticated and its' tag is non-malleably encoded. Therefore, at best, the tampering function can make the tampered $\tilde{k_a}, \tilde{t}$ become independent of the underlying values. Assuming that the MAC verifies on the tampered key and tag, one might like to believe that it guarantees independence of $\tilde{k_e}$ and, therefore, of the underlying message $\tilde{m}$ as well. Unfortunately, this reasoning is not true for message authentication codes with short tags. Specifically, when tags are much shorter than the message, there will necessarily be collisions in the tag space of a given key – i.e., on a given key, there could be multiple message that map to the same tag value. [6] As we describe in the attack below, these "collisions" can be exploited to make the code "malleable".

Attack on the Candidate Construction: To describe an attack, we need to specify tampering functions f_1, f_2, f_3, f_4. We use $x[0]$ to denote the least significant bit of the binary string x in the description below.

1. Choose constants k_0, k_1 from encryption key space, ct_0, ct_1 from ciphertext space such that $ct_0[0] = k_0[0] = 0$, $ct_1[0] = k_1[0] = 1$ and a tag t^* and a key k_a^* such that $\mathsf{Tag}_{k_a^*}(ct_0||k_0) = \mathsf{Tag}_{k_a^*}(ct_0||k_1) = \mathsf{Tag}_{k_a^*}(ct_1||k_0) = \mathsf{Tag}_{k_a^*}(ct_1||k_1) = t^*$. Observe that these values are all independent of the message as well as the randomness of the encoding scheme described above. Now we describe the four tampering functions.
2. $f_1(c_1)$: If $c_1[0] = 0$, set $c_1^* = ct_0$ otherwise $c_1^* = ct_1$.
3. $f_2(c_2)$: If $c_2[0] = 0$, set $c_2^* = k_0$ otherwise $c_2^* = k_1$.
4. Compute $c_3^*, c_4^* = \mathsf{NMEnc}(k_a^*, t^*)$
5. $f_3(c_3) = c_3^*$
6. $f_4(c_4) = c_4^*$

Carefully working through our choice of the various constants will show us that the tampered message will retain the least significant bit of the underlying message i.e., $\tilde{m}[0] = m[0]$, where $\tilde{m}$ is the tampered message. Furthermore, collisions in the MAC scheme have been exploited to ensure that tag of the tampered message and key will always verify. Thus any tampering is undetected and reveals information about the underlying message, thus violating non-malleability.

Analyzing the intuition behind the attack, we observe that the main challenge is that, even though the key and the ciphertext are tampered independently, jointly they may retain information about the original message. To overcome this issue, we modify the construction to ensure that the tampered key is never related to the original key. Ensuring this independence proves to be our major bottleneck. We are able to overcome this bottleneck through a somewhat surprising use of (strong) randomness extractors.

[6] This problem does not arise with a MAC such as $ax + b$ where (a, b) is the MAC key and x is the underlying message. There, for a fixed key and fixed tag, there is a unique message which satisfies the linear equation.

Using Randomness Extractors to "Amplify" Non-Malleability: Informally, randomness extractors allow us to transform non-uniform randomness into uniform randomness. Here we use randomness extractors to generate the key k_e i.e., $k_e = \mathsf{Ext}(w; s)$ where w and s are uniformly random string of appropriate lengths. At the outset, this might seem completely pointless: after all, extractors are typically used in settings where one does not have perfect randomness. This is clearly not the case here: indeed, the encoding scheme is allowed to choose its' own randomness! *How can choosing k_e as the output of an extractor be of any help?* Showing how the randomness extractor helps in this scenario is the crux of our proof. We consider the following cases:

1. s, w are both unchanged: In this case, the extracted encryption key remains unchanged. While it remains unclear how to argue non-malleability in this scenario, for now, it suffices to note that the attack described above is no longer relevant and, therefore, we defer a discussion on this case to later.
2. s is changed to an independent seed $\tilde{s}$: [7] In this case, $\tilde{k_e}$ is independent of k_e, regardless of how $\tilde{w}$ depends on w. As mentioned earlier, here too the attack described above is no longer relevant.
3. s is unchanged but w is changed in a related manner: In this case, $\tilde{k_e}$ could contain information about k_e.

Case 3 seems to still retain our original bottleneck and we handle it by ensuring that, in our construction, whenever the source w is changed, the seed s also needs to be changed. This reduces it to Case 2. What remains, is to show how we can ensure this through a delicate use of randomness extractors, message authentication codes and non-malleable codes.

Overview of Our construction: We use the following tools in our construction: a) A non-malleable code for 2-split-state model achieving rate $\Omega(1/\log\ n)$ ([22]) where n is the block-length ; b) a one-time information theoretic message authentication code; c) an average-case strong randomness extractor; d) a perfectly secure encryption scheme, like One Time Pad.

Step I: We use a randomness extractor (which typically have short seeds) to extract the encryption key.
Step II: We encrypt the message using the extracted key.
Step III: To detect modification of the source (used to extract the key) and the ciphertext, we authenticate them using two different one time MACs [8]
Step IV: Finally, we encode the authentication keys and tags along with the seed (used to extract the key) using a 2-state non-malleable code. We output the 2-state codeword, the source and the ciphertext.

The non-malleable encoding in Step IV ties various key components of our construction together and is crucial in overcoming the challenge described in Case 3.

[7] We ensure this by encoding s using a non-malleable code

[8] It is crucial to authenticate them separately as, a construction where we do not authenticate them separately is insecure. This is brought out in the security proof later.

Proof techniques: We prove non-malleability via a series of statistically-close hybrids which take us from the tampered game to a simulated game. But some non-trivial challenges arise in our proof: firstly, there are dependencies across states (e.g.: we include the source in one state and the encoding of its tag in another). So, even though the states are modified independently, the modifications will be interlinked through this dependency. Secondly, even though in our encoding, we choose the source uniformly at random, the decode process reveals information about the source. This will prevent us from using extractor security directly. The trick that helps us here is that we capture all the information learnt via the decoding using auxiliary information that is *independent* of the seed. This will allow us to use the crucial extractor security in our proof.

1.3 Organization of the Paper

We describe preliminaries and building blocks of the construction in Sections 2 and 3, respectively. We then give the main construction in Section 4.2, followed by the proof in Section 4.3. We then give a detailed analysis of the rate and error in Sections 4.4, 4.5, 4.6 and 4.7.

2 Preliminaries

Notation. κ denotes security parameter throughout. $s \in_R S$ denotes uniform sampling from set S. $x \leftarrow X$ denotes sampling from a probability distribution X. $x||y$ represents concatenation of two binary strings x and y. $|x|$ denotes length of binary string x. U_l denotes the uniform distribution on $\{0,1\}^l$. All logarithms are base 2.

Statistical distance and Entropy. Let X_1, X_2 be two probability distributions over some set S. Their *statistical distance* is

$$\mathbf{SD}\left(X_1, X_2\right) \stackrel{\text{def}}{=} \max_{T \subseteq S} \{\Pr[X_1 \in T] - \Pr[X_2 \in T]\} = \frac{1}{2} \sum_{s \in S} \left| \Pr_{X_1}[s] - \Pr_{X_2}[s] \right|$$

(they are said to be ε-close if $\mathbf{SD}\left(X_1, X_2\right) \leq \varepsilon$ and denoted by $X_1 \approx_\varepsilon X_2$). The *min-entropy* of a random variable W is $\mathbf{H}_\infty(W) = -\log(\max_w \Pr[W = w])$. For a joint distribution (W, E), define the (average) conditional min-entropy of W given E [13] as

$$\widetilde{\mathbf{H}}_\infty(W \mid E) = -\log(\mathop{\mathbf{E}}_{e \leftarrow E}(2^{-\mathbf{H}_\infty(W|E=e)}))$$

(here the expectation is taken over e for which $\Pr[E = e]$ is nonzero). For a random variable W over $\{0,1\}^n$, $W|E$ is said to be an (n,t) - source if $\widetilde{\mathbf{H}}_\infty(W|E) \geq t$.

Proposition 1. *Let $A_1, ..., A_n$ be mutually exclusive and exhaustive events. Then, for probability distributions X_1, X_2 over some set S, we have:*

$$\mathbf{SD}\left(X_1, X_2\right) \leq \sum_{i=1}^{n} \Pr[A_i].\mathbf{SD}\left(X_1|A_i, X_2|A_i\right)$$

where $X_j|A_i$ is the distribution of X_j conditioned on the event A_i.

Proof.

$$\begin{aligned}
2\mathbf{SD}\left(X_1, X_2\right) &= \sum_{s\in S} \Big| \Pr[X_1 = s] - \Pr[X_2 = s] \Big| \\
&= \sum_{s\in S} \left| \sum_{i=1}^{n} \Big(\Pr[A_i] \Pr[X_1 = s|A_i] - \Pr[A_i] \Pr[X_2 = s|A_i] \Big) \right| \\
&\leq \sum_{s\in S} \sum_{i=1}^{n} \Pr[A_i] \Big| \Pr[X_1 = s|A_i] - \Pr[X_2 = s|A_i] \Big| \\
&= \sum_{i=1}^{n} \Pr[A_i] \sum_{s\in S} \Big| \Pr[X_1 = s|A_i] - \Pr[X_2 = s|A_i] \Big| \\
&= 2\sum_{i=1}^{n} \Pr[A_i] \mathbf{SD}\left(X_1|A_i, X_2|A_i\right)
\end{aligned}$$

Lemma 2. *For any random variables A, B, C if $(A, B) \approx_\epsilon (A, C)$, then $B \approx_\epsilon C$*

Lemma 3. *For any random variables A, B if $A \approx_\epsilon B$, then for any function f, $f(A) \approx_\epsilon f(B)$*

Lemma 4. *Let A, B be correlated random variables over $\mathcal{A}, \mathcal{B}$. For randomized functions $F : \mathcal{A} \to \mathcal{X}$, $G : \mathcal{A} \to \mathcal{X}$ (randomness used is independent of B) if $\forall\ a \in \mathcal{A}, F(a) \approx_\epsilon G(a)$, then $(B, A, F(A)) \approx_\epsilon (B, A, G(A))$*

Proof. $2\mathbf{SD}\left((B, A, F(A)), (B, A, G(A))\right)$

$$= \sum_{b\in\mathcal{B},a\in\mathcal{A},x\in\mathcal{X}} \Big| \Pr[B=b \wedge A=a \wedge F(A)=x] - \Pr[B=b \wedge A=a \wedge G(A)=x] \Big|$$

$$= \sum_{b\in\mathcal{B},a\in\mathcal{A},x\in\mathcal{X}} \Pr[B=b] \Big| \Pr[A=a \wedge F(A)=x | B=b] - \Pr[A=a \wedge G(A)=x | B=b] \Big|$$

$$= \sum_{b\in\mathcal{B},a\in\mathcal{A},x\in\mathcal{X}} \Pr[B=b] \Pr[A=a|B=b].$$
$$\Big| \Pr[F(A)=x|A=a,B=b] - \Pr[G(A)=x|A=a,B=b] \Big|$$

$$= \sum_{b\in\mathcal{B},a\in\mathcal{A},x\in\mathcal{X}} \Pr[B=b] \Pr[A=a|B=b].$$
$$\Big| \Pr[F(a)=x|B=b] - \Pr[G(a)=x|B=b] \Big|$$

$$= \sum_{b\in\mathcal{B},a\in\mathcal{A},x\in\mathcal{X}} \Pr[B=b] \Pr[A=a|B=b] \Big| \Pr[F(a)=x] - \Pr[G(a)=x] \Big|$$

$$= \sum_{b\in\mathcal{B},a\in\mathcal{A}} \Pr[A=a \wedge B=b] \sum_{x\in\mathcal{X}} \Big| \Pr[F(a)=x] - \Pr[G(a)=x] \Big|$$

$$\leq \sum_{b\in\mathcal{B},a\in\mathcal{A}} \Pr[A=a \wedge B=b] \cdot 2\epsilon = 2\epsilon$$

We also use the following lemma [13, Lemma 2.2b], which says that average min-entropy of a random variable does not decrease by more than the length of the correlated random variable.

Lemma 5. *If B has at most 2^λ possible values, then $\widetilde{\mathbf{H}}_\infty(A \mid B) \geq \mathbf{H}_\infty(A,B) - \lambda \geq \mathbf{H}_\infty(A) - \lambda$. and, more generally, $\widetilde{\mathbf{H}}_\infty(A \mid B,C) \geq \widetilde{\mathbf{H}}_\infty(A,B \mid C) - \lambda \geq \widetilde{\mathbf{H}}_\infty(A \mid C) - \lambda$*

2.1 Definitions

Definition 1. *[8] A (possibly randomized) function* $\mathsf{Enc} : \{0,1\}^l \to \{0,1\}^n$ *and a deterministic function* $\mathsf{Dec} : \{0,1\}^n \to \{0,1\}^l \cup \{\perp\}$ *is said to be a* coding scheme *if* $\forall\ m \in \{0,1\}^l,\ \Pr[\mathsf{Dec}(\mathsf{Enc}(m)) = m] = 1$. *$l$ is called the*

message length and n is called the block length or the codeword length. Rate of a coding scheme is given by $\frac{l}{n}$.

Definition 2. *[8] A coding scheme* $(\mathsf{Enc}, \mathsf{Dec})$ *with message and codeword spaces as* $\{0,1\}^l, \{0,1\}^n$ *respectively, is* ϵ*- non-malleable with respect to a function family* $\mathcal{F} \subseteq \{f : \{0,1\}^n \to \{0,1\}^n\}$ *if* $\forall\ f \in \mathcal{F}$, $\exists$ *a distribution* Sim_f *over* $\{0,1\}^l \cup \{same^*, \perp\}$ *such that* $\forall\ m \in \{0,1\}^l$

$$\mathsf{Tamper}_f^m \approx_\epsilon \mathsf{Copy}_{Sim_f}^m$$

where Tamper_f^m *denotes the distribution* $\mathsf{Dec}(f(\mathsf{Enc}(m)))$ *and* $\mathsf{Copy}_{Sim_f}^m$ *is defined as*

$$\tilde{m} \leftarrow Sim_f$$

$$\mathsf{Copy}_{Sim_f}^m = \begin{cases} m\ if\ \tilde{m} = same^* \\ \tilde{m}\ \mathsf{otherwise} \end{cases}$$

Sim_f should be efficiently samplable given oracle access to $f(.)$.

3 Building blocks

We use information-theoretic message authentication codes, strong average case extractor and an existing 2-split state non-malleable codes construction by Li [22], as building blocks to our construction. We briefly discuss about these building blocks below.

3.1 One-Time Message Authentication Codes [14]

A family of pair of functions $\{\mathsf{Tag}_{k_a} : \{0,1\}^\gamma \to \{0,1\}^\delta, \mathsf{Vrfy}_{k_a} : \{0,1\}^\gamma \times \{0,1\}^\delta \to \{0,1\}\}_{k_a \in \{0,1\}^\tau}$ is said to be a $\mu -$ secure one time MAC if

1. For $k_a \in_R \{0,1\}^\tau$, $\forall\ m \in \{0,1\}^\gamma$, $\Pr[\mathsf{Vrfy}_{k_a}(m, \mathsf{Tag}_{k_a}(m)) = 1] = 1$
2. For any $m \neq m', t, t'$, $\Pr\limits_{k_a}[\mathsf{Tag}_{k_a}(m) = t | \mathsf{Tag}_{k_a}(m') = t'] \leq \mu$ for $k_a \in_R \{0,1\}^\tau$

3.2 Extractors

Extractors [24] yield a close-to-uniform string from a random variable with high min-entropy, using a uniformly random seed i as a kind of catalyst. Strong extractors are ones in which the extracted string looks random even in the presence of the seed. We will use only strong extractors in this paper and thus sometimes omit the adjective "strong."

Definition 3. *Let* $\mathsf{Ext} : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^l$ *be a polynomial time probabilistic function that uses d bits of randomness. We say that* Ext *is an* $(n, t, d, l, \varepsilon)$-strong extractor *if for all random variables W over* $\{0,1\}^n$ *such that* $\mathbf{H}_\infty(W) \geq t$, *we have* $\mathbf{SD}\left((\mathsf{Ext}(W; X), X), (U_l, X)\right) \leq \varepsilon$, *where X is the uniform distribution over* $\{0,1\}^d$.

Universal hash functions are perhaps the simplest extractors, allowing $t = \ell - 2 + 2\log\frac{1}{\varepsilon}$ (see [25, Theorem 8.1], [20, Lemma 4.8], and references therein).

If an extractor works when the guarantee on W is for conditional min-entropy rather than min-entropy, it is called an *average-case* extractor. This notation was introduced in [13, Section 2.5]. Vadhan [26, Problem 6.8] showed that all extractors are average-case extractors with a slight loss of parameters: namely, any (t, ε)-extractor for $t \leq n - 1$ is also a $(t, 3\varepsilon)$-average-case extractor. Some extractors—namely, universal hash function [4]—don't lose parameters at all in the average case [13, Section 2.5] (in fact, almost universal hash functions [25] work as well [12]).

3.3 Li's construction of 2-split state Non-malleable code

Lemma 6. *[22, Theorem 7.12] For any $\beta \in \mathbb{N}$ there exists an explicit non-malleable code with efficient encoder/decoder in 2-split state model with block length 2β, rate $\Omega\left(\frac{1}{\log\ \beta}\right)$ and error $= 2^{-\Omega\left(\frac{\beta}{\log\ \beta}\right)}$*

Let the message length be α for the non-malleable code in Lemma 6. By Lemma 6, we have

$$\frac{\alpha}{2\beta} = \Omega\left(\frac{1}{\log \beta}\right)$$

$$\Rightarrow \alpha = \Omega\left(\frac{\beta}{\log(\beta)}\right)$$

By Lemma 10, we have

$$\beta = \mathcal{O}(\alpha.\log(\alpha)) \tag{1}$$

4 Construction

Before we present our construction, we briefly summarize some important points that we discussed in Section 1. We observe that a non-malleable code is unlikely to be secure if the message m is revealed in the clear in any of the states. If it did, then the tampering function for that state could choose whether or not to tamper depending on the information it learns. It is for this reason that, in our construction, we need to encrypt the message (using a one-time pad) and then store the key as well as the ciphertext in separate states. To prevent the adversary from tampering with these in a related manner, we authenticate it using a key k_a. We encode k_a as well as the tags using a non-malleable code to ensure that any non-trivial tampering will render these independent of the underlying k_a and tags. However, as described in Section 1, if we store the encryption key k in the clear, then using the collisions in MAC, we can tamper the key and the ciphertext in a related way, hence leading to a related tampered

message. We observe that if we are able to relate the tampered cipher-text but not the tampered encryption key $\tilde{k}$ to k, then the attack described in Section 1 no longer holds. Therefore, a key concern we address as we design our scheme is the following: *can we ensure the independence of any tampered encryption key $\tilde{k}$ from the underlying encryption key k?*

We show that a use of randomness extractors to generate k, combined with a careful use of message authentication codes helps us achieve this independence.

4.1 Notation

- $\mathsf{NMEnc}, \mathsf{NMDec}$ be an ε_1-secure two split state non-malleable code over message and codeword spaces as $\{0,1\}^{\alpha}$, $\{0,1\}^{\beta_1} \times \{0,1\}^{\beta_2}$ respectively (as in Lemma 6), with the message length α and the length of the two states, β_1, β_2, respectively. $\mathsf{NMTamper}^{\mathsf{m}}_{\mathsf{f}_1,\mathsf{f}_2}, \mathsf{NMSim}_{\mathsf{f}_1,\mathsf{f}_2}$ denote the tampered message distribution of m and the simulator of $\mathsf{NMEnc}, \mathsf{NMDec}$ with respect to tampering functions f_1, f_2
- $\mathsf{Tag}, \mathsf{Vrfy}$ be an information theoretic ε_2 secure one time MAC (as in Lemma 9) over key, message, tag spaces as $\{0,1\}^{\tau_1}, \{0,1\}^{n}, \{0,1\}^{\delta_1}$ respectively.
- $\mathsf{Tag}', \mathsf{Vrfy}'$ be an information theoretic ε_3 secure one time MAC (as in Lemma 9) over key, message, tag spaces as $\{0,1\}^{\tau_2}, \{0,1\}^{l}, \{0,1\}^{\delta_2}$ respectively.
- Ext be an $(n, t, d, l, \varepsilon_4)$ average case strong extractor.

The parameters will be chosen such that $\alpha = \tau_1 + \tau_2 + \delta_1 + \delta_2 + d$ and $n > 1 + \tau_2 + \delta_2 + l + t$. (Refer to Section 4.5 for details)

4.2 Construction Overview

We now define a construction for l bit messages in the four split state model. The idea is to use a randomness extractor (which typically have short seeds) to extract the key and then encode the seed using the underlying non-malleable code. Further, the source and the ciphertext are stored in separate parts of the codeword. We then authenticate the source and the ciphertext using two different one time MAC schemes and then encode the authentication keys and tags using the underlying non-malleable code. In other words, we define an encoder, which sends the ciphertext, the source (used to extract the encryption key) and the 2-state codeword encoding the two pairs of authentication keys and tags and the seed. The construction is described below:

$\mathsf{Enc}(m)$:

- $w \in_R \{0,1\}^n$, $s \in_R \{0,1\}^d$
- $k_{a_1} \in_R \{0,1\}^{\tau_1}, k_{a_2} \in_R \{0,1\}^{\tau_2}$
- $k = \mathsf{Ext}(w, s)$
- $C = m \oplus k$
- $t_1 = \mathsf{Tag}_{k_{a_1}}(w)$, $t_2 = \mathsf{Tag}'_{k_{a_2}}(C)$
- $(L, R) = \mathsf{NMEnc}(k_{a_1}||k_{a_2}||t_1||t_2||s)$
- output $:(L, R, w, C)$

$\mathsf{Dec}(L, R, w, C)$:

- $k_{a_1}||k_{a_2}||t_1||t_2||s = \mathsf{NMDec}(L, R)$
- If $k_{a_1}||k_{a_2}||t_1||t_2||s = \perp$ output $\perp$
- else if $\mathsf{Vrfy}_{k_{a_1}}(w, t_1) = 1$
  $\wedge\ \mathsf{Vrfy}'_{k_{a_2}}(C, t_2) = 1$
  output $C \oplus \mathsf{Ext}(w, s)$
- else output $\perp$

Theorem 1. *Let* $\mathsf{NMEnc}, \mathsf{NMDec}$ *be an* ε_1*-secure two split state non-malleable code,* $\mathsf{Tag}, \mathsf{Vrfy}$ *be an information theoretic* ε_2 *secure one time* MAC *and* $\mathsf{Tag}', \mathsf{Vrfy}'$ *be an information theoretic* ε_3 *secure one time* MAC *as given above. Let* Ext *be an* $(n, t, d, l, \varepsilon_4)$ *average case strong extractor. Let* $\alpha = \tau_1 + \tau_2 + \delta_1 + \delta_2 + d$ *and* $n > 1 + \tau_2 + \delta_2 + l + t$.

For any constant $\zeta > 0$*, messages of length* l*, any* κ *such that* $\kappa = o\left(\frac{l}{\log l}\right)$*, the construction in figure above has block length* $(3+\zeta)l + o(l)$*, there by achieves asymptotic rate* $\frac{1}{3+\zeta}$ *and error* $2^{-\kappa}$.

Proof. We give the proof in two steps. Firstly, we prove that the proposed construction is a non-malleable coding scheme (Section 4.3). Secondly, we set the parameters to achieve the desired rate and error (Section 4.4).

4.3 Security proof

Define the 4-split-state tampering family for the above construction as

$$\mathcal{F} = \{(h_1, h_2, f, g) : h_1 : \{0,1\}^{\beta_1} \to \{0,1\}^{\beta_1}, h_2 : \{0,1\}^{\beta_2} \to \{0,1\}^{\beta_2},$$
$$f : \{0,1\}^n \to \{0,1\}^n, g : \{0,1\}^l \to \{0,1\}^l\}$$

To show that $(\mathsf{Enc}, \mathsf{Dec})$ is non-malleable we need to show that $\forall\ (h_1, h_2, f, g) \in \mathcal{F}$, $\exists\ Sim_{h_1,h_2,f,g}$ such that $\forall\ m \in \{0,1\}^l$

$$Tamper^m_{h_1,h_2,f,g} \approx_\varepsilon Copy^m_{Sim_{h_1,h_2,f,g}}$$

Let $(h_1, h_2, f, g) \in \mathcal{F}$. We define the following simulator:

$Sim_{h_1,h_2,f,g}$:

1. $k \in_R \{0,1\}^l$
2. $C = 0 \oplus k$
3. $w \in_R \{0,1\}^n$
4. $(\tilde{w}, \tilde{C}) = (f(w), g(C))$
5. $\tilde{k_{a_1}}||\tilde{k_{a_2}}||\tilde{t_1}||\tilde{t_2}||\tilde{s} \leftarrow \mathsf{NMSim}_{h_1,h_2}$
6. If $\tilde{k_{a_1}}||\tilde{k_{a_2}}||\tilde{t_1}||\tilde{t_2}||\tilde{s} = \perp$, output $\perp$
7. else if $\tilde{k_{a_1}}||\tilde{k_{a_2}}||\tilde{t_1}||\tilde{t_2}||\tilde{s} = same^*$
   a. If $\tilde{w} = w$ and $\tilde{C} = C$ output $same^*$
   b. else output $\perp$
8. else if $\mathsf{Vrfy}_{\tilde{k_{a_1}}}(\tilde{w}, \tilde{t_1}) = 1 \wedge \mathsf{Vrfy}'_{\tilde{k_{a_2}}}(\tilde{C}, \tilde{t_2}) = 1$ output $\tilde{C} \oplus \mathsf{Ext}(\tilde{w}; \tilde{s})$
9. else output $\perp$

We now prove the statistical closeness of the tampered random variable and the simulated random variable through a sequence of hybrids.

Proof Overview. At a high level, our goal is to remove the dependency of $\tilde{m}$ on m, through a series of hybrids. The codeword depends on m, directly or indirectly, through various random variables such as the seed s, w, the authentication keys as well as the tags. To begin with, we wish to remove the dependence of the tampered extracted key (used to decrypt the codeword) on the original extracted key. Through a series of hybrids, we achieve this by removing the dependence of the tampered extracted key on the seed s. Once we do this, we use the extractor property, to remove the dependency of C on w and s. Finally, we use perfect security of the one-time pad to remove dependency of $\tilde{C}$ on m.

Going from Tamper *experiment to* $\mathsf{Hybrid1}^m_{h_1,h_2,f,g}$*:* $\mathsf{Hybrid1}^m_{h_1,h_2,f,g}$ is the same as the standard tampering experiment except that we use the simulator for the underlying non-malleable code to generate the tampered random variable $\tilde{k_{a_1}}||\tilde{k_{a_2}}||\tilde{t_1}||\tilde{t_2}||\tilde{s}$.

Claim. If $(\mathsf{NMEnc}, \mathsf{NMDec})$ is an ε_1-secure non-malleable code, then $\mathsf{Tamper}^m_{h_1,h_2,f,g} \approx_{\varepsilon_1} \mathsf{Hybrid1}^m_{h_1,h_2,f,g}$

$\mathsf{Tamper}^m_{h_1,h_2,f,g}$:

1. $w \in_R \{0,1\}^n,\ s \in_R \{0,1\}^d$
2. $k_{a_1} \in_R \{0,1\}^{\tau_1}, k_{a_2} \in_R \{0,1\}^{\tau_2}$
3. $k = \mathsf{Ext}(w; s)$
4. $C = m \oplus k$
5. $t_1 = \mathsf{Tag}_{k_{a_1}}(w),\ t_2 = \mathsf{Tag}'_{k_{a_2}}(C)$
6. $(\tilde{w}, \tilde{C}) = (f(w), g(C))$
7. $\tilde{k_{a_1}}||\tilde{k_{a_2}}||\tilde{t_1}||\tilde{t_2}||\tilde{s} \leftarrow \mathsf{NMTamper}^{k_{a_1}||k_{a_2}||t_1||t_2||s}_{h_1,h_2}$
8. If $\tilde{k_{a_1}}||\tilde{k_{a_2}}||\tilde{t_1}||\tilde{t_2}||\tilde{s} = \perp$, output $\perp$
9. else if $\mathsf{Vrfy}_{\tilde{k_{a_1}}}(\tilde{w}, \tilde{t_1}) = 1 \wedge \mathsf{Vrfy}'_{\tilde{k_{a_2}}}(\tilde{C}, \tilde{t_2}) = 1$ output $\tilde{C} \oplus \mathsf{Ext}(\tilde{w}; \tilde{s})$
10. else output $\perp$

$\mathsf{Hybrid1}^m_{h_1,h_2,f,g}$:

1. $w \in_R \{0,1\}^n,\ s \in_R \{0,1\}^d$
2. $k_{a_1} \in_R \{0,1\}^{\tau_1}, k_{a_2} \in_R \{0,1\}^{\tau_2}$
3. $k = \mathsf{Ext}(w; s)$
4. $C = m \oplus k$
5. $t_1 = \mathsf{Tag}_{k_{a_1}}(w),\ t_2 = \mathsf{Tag}'_{k_{a_2}}(C)$
6. $(\tilde{w}, \tilde{C}) = (f(w), g(C))$

7a. $\tilde{k_{a_1}}||\tilde{k_{a_2}}||\tilde{t_1}||\tilde{t_2}||\tilde{s} \leftarrow \mathsf{NMSim}_{h_1,h_2}$

7b. If $\tilde{k_{a_1}}||\tilde{k_{a_2}}||\tilde{t_1}||\tilde{t_2}||\tilde{s} = same^*$, assign $\tilde{k_{a_1}}||\tilde{k_{a_2}}||\tilde{t_1}||\tilde{t_2}||\tilde{s} = k_{a_1}||k_{a_2}||t_1||t_2||s$

8. If $\tilde{k_{a_1}}||\tilde{k_{a_2}}||\tilde{t_1}||\tilde{t_2}||\tilde{s} = \perp$, output $\perp$
9. else if $\mathsf{Vrfy}_{\tilde{k_{a_1}}}(\tilde{w}, \tilde{t_1}) = 1 \wedge \mathsf{Vrfy}'_{\tilde{k_{a_2}}}(\tilde{C}, \tilde{t_2}) = 1$ output $\tilde{C} \oplus \mathsf{Ext}(\tilde{w}; \tilde{s})$
10. else output $\perp$

Proof. We wish to use the statistical closeness of the tampered and simulated random variables corresponding to $(\mathsf{NMEnc}, \mathsf{NMDec})$, to prove the claim.
Now, we apply Lemma 4, taking $B = (w, C),\ A = (k_{a_1}||k_{a_2}||t_1||t_2||s)$, and the functions as $\mathsf{NMTamper}_{h_1,h_2}, \mathsf{Copy}_{\mathsf{NMSim}_{h_1,h_2}}$ to get:

$$\text{Since, } \mathsf{NMTamper}^{k_{a_1}||k_{a_2}||t_1||t_2||s}_{h_1,h_2} \approx_{\varepsilon_1} \mathsf{Copy}^{k_{a_1}||k_{a_2}||t_1||t_2||s}_{\mathsf{NMSim}_{h_1,h_2}}$$

hence we get,

$$(w, C, k_{a_1}||k_{a_2}||t_1||t_2||s, \mathsf{NMTamper}^{k_{a_1}||k_{a_2}||t_1||t_2||s}_{h_1,h_2}) \approx_{\varepsilon_1} (w, C, k_{a_1}||k_{a_2}||t_1||t_2||s, \mathsf{Copy}^{k_{a_1}||k_{a_2}||t_1||t_2||s}_{\mathsf{NMSim}_{h_1,h_2}})$$

$$\implies \text{By Lemma 2, } (w, C, \mathsf{NMTamper}^{k_{a_1}||k_{a_2}||t_1||t_2||s}_{h_1,h_2}) \approx_{\varepsilon_1} (w, C, \mathsf{Copy}^{k_{a_1}||k_{a_2}||t_1||t_2||s}_{\mathsf{NMSim}_{h_1,h_2}})$$

$$\implies \text{By Lemma 3, } (\tilde{w}, \tilde{C}, \mathsf{NMTamper}^{k_{a_1}||k_{a_2}||t_1||t_2||s}_{h_1,h_2}) \approx_{\varepsilon_1} (\tilde{w}, \tilde{C}, \mathsf{Copy}^{k_{a_1}||k_{a_2}||t_1||t_2||s}_{\mathsf{NMSim}_{h_1,h_2}}) \tag{2}$$

Now, we express the outputs of the hybrids as a deterministic function, Q, of the above variables, to apply Lemma 3 and hence prove the claim.
$Q(\tilde{w}, \tilde{C}, \tilde{k_{a_1}}||\tilde{k_{a_2}}||\tilde{t_1}||\tilde{t_2}||\tilde{s})$:

- If $\tilde{k_{a_1}}||\tilde{k_{a_2}}||\tilde{t_1}||\tilde{t_2}||\tilde{s} = \perp$, output $\perp$
- else if $\mathsf{Vrfy}_{\tilde{k_{a_1}}}(\tilde{w}, \tilde{t_1}) = 1 \wedge \mathsf{Vrfy}'_{\tilde{k_{a_2}}}(\tilde{C}, \tilde{t_2}) = 1$ output $\tilde{C} \oplus \mathsf{Ext}(\tilde{w}; \tilde{s})$
- else output $\perp$

Then, using Eq. 2 and Lemma 3, we get

$$Q(\tilde{w}, \tilde{C}, \mathsf{NMTamper}^{k_{a_1}||k_{a_2}||t_1||t_2||s}_{h_1,h_2}) \approx_{\varepsilon_1} Q(\tilde{w}, \tilde{C}, \mathsf{Copy}^{k_{a_1}||k_{a_2}||t_1||t_2||s}_{\mathsf{NMSim}_{h_1,h_2}})$$

$$\implies \mathsf{Tamper}^m_{h_1,h_2,f,g} \approx_{\varepsilon_1} \mathsf{Hybrid1}^m_{h_1,h_2,f,g}$$

Going from $\mathsf{Hybrid1}^m_{h_1,h_2,f,g}$ *to* $\mathsf{Hybrid2}^m_{h_1,h_2,f,g}$*:* As will become evident later, $\mathsf{Hybrid1}^m_{h_1,h_2,f,g}$ is what will allow us to argue that, in the restricted case where $\tilde{s} \neq s$, the extracted key $\tilde{k}$ is independent of s. We now move to $\mathsf{Hybrid2}^m_{h_1,h_2,f,g}$ which is the same as $\mathsf{Hybrid1}^m_{h_1,h_2,f,g}$ except for the the case where s is unchanged. In this case, as we show in $\mathsf{Hybrid2}^m_{h_1,h_2,f,g}$, the output of the experiment can be computed without evaluating $\tilde{k}$. We prove that $\mathsf{Hybrid1}^m_{h_1,h_2,f,g}$ and $\mathsf{Hybrid2}^m_{h_1,h_2,f,g}$ are statistically close by using the unforgeability of the message authentication scheme.

Claim. If $(\mathsf{Tag}, \mathsf{Vrfy})$ and $(\mathsf{Tag}', \mathsf{Vrfy}')$ are ε_2-, ε_3-secure information theoretic one-time MAC (respectively), then $\mathsf{Hybrid1}^m_{h_1,h_2,f,g} \approx_{\varepsilon_2+\varepsilon_3} \mathsf{Hybrid2}^m_{h_1,h_2,f,g}$

$\mathsf{Hybrid1}^m_{h_1,h_2,f,g}$:

1. $w \in_R \{0,1\}^n,\ s \in_R \{0,1\}^d$
2. $k_{a_1} \in_R \{0,1\}^{\tau_1}, k_{a_2} \in_R \{0,1\}^{\tau_2}$
3. $k = \mathsf{Ext}(w; s)$
4. $C = m \oplus k$
5. $t_1 = \mathsf{Tag}_{k_{a_1}}(w),\ t_2 = \mathsf{Tag}'_{k_{a_2}}(C)$
6. $(\tilde{w}, \tilde{C}) = (f(w), g(C))$
7. $\tilde{k_{a_1}}||\tilde{k_{a_2}}||\tilde{t_1}||\tilde{t_2}||\tilde{s} \leftarrow \mathsf{NMSim}_{h_1,h_2}$
8. If $\tilde{k_{a_1}}||\tilde{k_{a_2}}||\tilde{t_1}||\tilde{t_2}||\tilde{s} = same^*$, assign $\tilde{k_{a_1}}||\tilde{k_{a_2}}||\tilde{t_1}||\tilde{t_2}||\tilde{s} = k_{a_1}||k_{a_2}||t_1||t_2||s$
9. If $\tilde{k_{a_1}}||\tilde{k_{a_2}}||\tilde{t_1}||\tilde{t_2}||\tilde{s} = \perp$, output $\perp$
10. else if $\mathsf{Vrfy}_{\tilde{k_{a_1}}}(\tilde{w}, \tilde{t_1}) = 1 \wedge \mathsf{Vrfy}'_{\tilde{k_{a_2}}}(\tilde{C}, \tilde{t_2}) = 1$ output $\tilde{C} \oplus \mathsf{Ext}(\tilde{w}; \tilde{s})$
11. else output $\perp$

$\mathsf{Hybrid2}^m_{h_1,h_2,f,g}$:

1. $w \in_R \{0,1\}^n,\ s \in_R \{0,1\}^d$
3. $k = \mathsf{Ext}(w; s)$
4. $C = m \oplus k$
6. $(\tilde{w}, \tilde{C}) = (f(w), g(C))$
7. $\tilde{k_{a_1}}||\tilde{k_{a_2}}||\tilde{t_1}||\tilde{t_2}||\tilde{s} \leftarrow \mathsf{NMSim}_{h_1,h_2}$
8. If $\tilde{k_{a_1}}||\tilde{k_{a_2}}||\tilde{t_1}||\tilde{t_2}||\tilde{s} = same^*$
   - If $\tilde{w} = w$ and $\tilde{C} = C$ output m
   - else output $\perp$
9. If $\tilde{k_{a_1}}||\tilde{k_{a_2}}||\tilde{t_1}||\tilde{t_2}||\tilde{s} = \perp$, output $\perp$
10. else if $\mathsf{Vrfy}_{\tilde{k_{a_1}}}(\tilde{w}, \tilde{t_1}) = 1 \wedge \mathsf{Vrfy}'_{\tilde{k_{a_2}}}(\tilde{C}, \tilde{t_2}) = 1$ output $\tilde{C} \oplus \mathsf{Ext}(\tilde{w}; \tilde{s})$
11. else output $\perp$

Proof. If $same^*$ is not the value sampled from NMSim_{h_1,h_2}, then $\mathsf{Hybrid1}^m_{h_1,h_2,f,g}$ and $\mathsf{Hybrid2}^m_{h_1,h_2,f,g}$ can be evaluated without steps (2,5,8) and (8) respectively. The output of the two hybrids are identical in this case. Therefore, the statistical distance is zero in this case. When $same^*$ is sampled, the key difference between $\mathsf{Hybrid1}^m_{h_1,h_2,f,g}$ and $\mathsf{Hybrid2}^m_{h_1,h_2,f,g}$ is that, corresponding to this case, we remove the two verify checks (of Step 10) in $\mathsf{Hybrid2}^m_{h_1,h_2,f,g}$ and simply replace it with the checks shown in Step 8. By Proposition 1 and above observation, we get:

$$\mathbf{SD}\left(\mathsf{Hybrid1}^m_{h_1,h_2,f,g}; \mathsf{Hybrid2}^m_{h_1,h_2,f,g}\right) \leq \Pr[\mathsf{NMSim}_{h_1,h_2} = same^*] \cdot$$
$$\mathbf{SD}\left(\mathsf{Hybrid1}^m_{h_1,h_2,f,g} | \mathsf{NMSim} h_1, h_2 = same^*; \mathsf{Hybrid2}^m_{h_1,h_2,f,g} | \mathsf{NMSim} h_1, h_2 = same^*\right)$$

So, now remains the case when NMSim_{h_1,h_2} outputs $same^*$. By using unforgeability of $(\mathsf{Tag}, \mathsf{Vrfy})$ and $(\mathsf{Tag}', \mathsf{Vrfy}')$ we show the that two hybrids are statistically close.

- Let E be the event that $same^*$ is sampled from NMSim_{h_1,h_2} and $\tilde{E}$ be its compliment.
- Let F be the event that $\tilde{w} = w \wedge \tilde{C} = C$ and $\tilde{F}$ its complement.
- E and F are independent because $\tilde{w}, \tilde{C}$ are deterministic functions of w and C respectively (which are independent of $\mathsf{NMSim} h_1, h_2$) and $\mathsf{NMSim} h_1, h_2$ does not take any input except for the a-priori fixed tampering functions h_1, h_2.

$$
\begin{aligned}
&2\mathbf{SD}\left(\mathsf{Hybrid1}^m_{h_1,h_2,f,g};\mathsf{Hybrid2}^m_{h_1,h_2,f,g}\right)\\
&= \sum_{\tilde{m}\in\{0,1\}^l\cup\{\perp\}} \Big|\ \Pr[\mathsf{Hybrid1}^m_{h_1,h_2,f,g}=\tilde{m}] - \Pr[\mathsf{Hybrid2}^m_{h_1,h_2,f,g}=\tilde{m}]\ \Big|\\
&= \sum_{\tilde{m}\in\{0,1\}^l\cup\{\perp\}} \Big|\ \Pr[E]\Big(\ \Pr[\mathsf{Hybrid1}^m_{h_1,h_2,f,g}=\tilde{m}|E] - \Pr[\mathsf{Hybrid2}^m_{h_1,h_2,f,g}=\tilde{m}|E]\ \Big)\\
&\qquad\qquad + \Pr[\tilde{E}]\underbrace{\Big(\ \Pr[\mathsf{Hybrid1}^m_{h_1,h_2,f,g}=\tilde{m}|\tilde{E}] - \Pr[\mathsf{Hybrid2}^m_{h_1,h_2,f,g}=\tilde{m}|\tilde{E}]\ \Big)}_{=0 \text{ as given } \tilde{E} \text{ both the hybrids are identical.}}\ \Big|\\
&= \sum_{\tilde{m}\in\{0,1\}^l\cup\{\perp\}} \Pr[E]\ \Big|\ \Pr[\mathsf{Hybrid1}^m_{h_1,h_2,f,g}=\tilde{m}|E] - \Pr[\mathsf{Hybrid2}^m_{h_1,h_2,f,g}=\tilde{m}|E]\ \Big|\\
&= \Pr[E] \sum_{\tilde{m}\in\{0,1\}^l\cup\{\perp\}} \Big|\ \Pr[F|E].\\
&\qquad \Big(\ \underbrace{\Pr[\mathsf{Hybrid1}^m_{h_1,h_2,f,g}=\tilde{m}|E,F] - \Pr[\mathsf{Hybrid2}^m_{h_1,h_2,f,g}=\tilde{m}|E,F]}_{=0 \text{ as given E and F both the hybrids output } m. \text{ So for any } \tilde{m} \text{ the difference is } 0}\ \Big) + \Pr[\tilde{F}|E].\\
&\qquad\qquad \Big(\ \Pr[\mathsf{Hybrid1}^m_{h_1,h_2,f,g}=\tilde{m}|E,\tilde{F}] - \Pr[\mathsf{Hybrid2}^m_{h_1,h_2,f,g}=\tilde{m}|E,\tilde{F}]\Big)\ \Big|\\
&= \Pr[E] \sum_{\tilde{m}\in\{0,1\}^l\cup\{\perp\}} \Big|\ \Pr[\tilde{F}]\Big(\ \Pr[\mathsf{Hybrid1}^m_{h_1,h_2,f,g}=\tilde{m}|E,\tilde{F}] -\\
&\qquad\qquad\qquad\qquad \Pr[\mathsf{Hybrid2}^m_{h_1,h_2,f,g}=\tilde{m}|E,\tilde{F}]\Big)\ \Big|\\
&= \Pr[E]\Pr[\tilde{F}] \sum_{\tilde{m}\in\{0,1\}^l} \Big|\ \Pr[\mathsf{Hybrid1}^m_{h_1,h_2,f,g}=\tilde{m}|E,\tilde{F}] - \Pr[\mathsf{Hybrid2}^m_{h_1,h_2,f,g}=\tilde{m}|E,\tilde{F}]\ \Big|\\
&\qquad + \Big|\ \Pr[\mathsf{Hybrid1}^m_{h_1,h_2,f,g}=\perp|E,\tilde{F}] - \underbrace{\Pr[\mathsf{Hybrid2}^m_{h_1,h_2,f,g}=\perp|E,\tilde{F}]}_{=1 \text{ as given E,}\tilde{F} \text{ Hybrid 2 outputs } \perp}\ \Big|\\
&= \Pr[E]\Pr[\tilde{F}] \sum_{\tilde{m}\in\{0,1\}^l} \Pr[\mathsf{Hybrid1}^m_{h_1,h_2,f,g}=\tilde{m}|E,\tilde{F}] + 1 - \Pr[\mathsf{Hybrid1}^m_{h_1,h_2,f,g}=\perp|E,\tilde{F}]\\
&= 2\Pr[E]\Pr[\tilde{F}]\Big(\ \Pr[\mathsf{Hybrid1}^m_{h_1,h_2,f,g}\neq\perp|E,\tilde{F}]\ \Big)\\
&\leq 2\Pr[E]\Pr[\tilde{F}]\Pr[\mathsf{Vrfy}_{k^{\sim}_{a_1}}(\tilde{w},\tilde{t_1})=1 \wedge \mathsf{Vrfy}'_{k^{\sim}_{a_2}}(\tilde{C},\tilde{t_2})=1 | t_1=\mathsf{Tag}_{k_{a_1}}(w), t_2=\mathsf{Tag}'_{k_{a_2}}(C), E, \tilde{F}]\\
&\leq 2\Pr[E]\Pr[\tilde{F}]\Pr[\mathsf{Vrfy}_{k_{a_1}}(\tilde{w},t_1)=1 \wedge \mathsf{Vrfy}'_{k_{a_2}}(\tilde{C},t_2)=1 | t_1=\mathsf{Tag}_{k_{a_1}}(w), t_2=\mathsf{Tag}'_{k_{a_2}}(C), \tilde{F}]\\
&\leq 2(\varepsilon_2+\varepsilon_3)
\end{aligned}
$$

$$\therefore \ \mathsf{Hybrid1}^m_{h_1,h_2,f,g} \approx_{\varepsilon_2+\varepsilon_3} \mathsf{Hybrid2}^m_{h_1,h_2,f,g}$$

Rewriting $\mathsf{Hybrid2}^m_{h_1,h_2,f,g}$ *as* $\mathsf{Hybrid3}^m_{h_1,h_2,f,g}$*:* Now we simply rewrite the $\mathsf{Hybrid2}^m_{h_1,h_2,f,g}$, starting with sampling from NMSim_{h_1,h_2}.

$\mathsf{Hybrid2}^m_{h_1,h_2,f,g}$:

1. $w \in_R \{0,1\}^n,\ s \in_R \{0,1\}^d$
2. $k = \mathsf{Ext}(w; s)$
3. $C = m \oplus k$
4. $(\tilde{w}, \tilde{C}) = (f(w), g(C))$
5. $\tilde{k_{a_1}}||\tilde{k_{a_2}}||\tilde{t_1}||\tilde{t_2}||\tilde{s} \leftarrow \mathsf{NMSim}_{h_1,h_2}$
6. If $\tilde{k_{a_1}}||\tilde{k_{a_2}}||\tilde{t_1}||\tilde{t_2}||\tilde{s} = same^*$
   - If $\tilde{w} = w$ and $\tilde{C} = C$ output m
   - else output $\perp$
7. If $\tilde{k_{a_1}}||\tilde{k_{a_2}}||\tilde{t_1}||\tilde{t_2}||\tilde{s} = \perp$, output $\perp$
8. else if $\mathsf{Vrfy}_{\tilde{k_{a_1}}}(\tilde{w}, \tilde{t_1}) = 1 \wedge \mathsf{Vrfy}'_{\tilde{k_{a_2}}}(\tilde{C}, \tilde{t_2}) = 1$ output $\tilde{C} \oplus \mathsf{Ext}(\tilde{w}; \tilde{s})$
9. else output $\perp$

$\mathsf{Hybrid3}^m_{h_1,h_2,f,g}$:

1. $\tilde{k_{a_1}}||\tilde{k_{a_2}}||\tilde{t_1}||\tilde{t_2}||\tilde{s} \leftarrow \mathsf{NMSim}_{h_1,h_2}$
2. $w \in_R \{0,1\}^n,\ s \in_R \{0,1\}^d$
3. $k = \mathsf{Ext}(w; s)$
4. $C = m \oplus k$
5. $(\tilde{w}, \tilde{C}) = (f(w), g(C))$
6. If $\tilde{k_{a_1}}||\tilde{k_{a_2}}||\tilde{t_1}||\tilde{t_2}||\tilde{s} = same^*$
   - If $\tilde{w} = w$ and $\tilde{C} = C$ output m
   - else output $\perp$
7. If $\tilde{k_{a_1}}||\tilde{k_{a_2}}||\tilde{t_1}||\tilde{t_2}||\tilde{s} = \perp$, output $\perp$
8. else if $\mathsf{Vrfy}_{\tilde{k_{a_1}}}(\tilde{w}, \tilde{t_1}) = 1 \wedge \mathsf{Vrfy}'_{\tilde{k_{a_2}}}(\tilde{C}, \tilde{t_2}) = 1$ output $\tilde{C} \oplus \mathsf{Ext}(\tilde{w}; \tilde{s})$
9. else output $\perp$

It is easy to see that we have rearranged the steps without changing the distributions of any of the random variable, $\mathsf{Hybrid2}^m_{h_1,h_2,f,g} \equiv \mathsf{Hybrid3}^m_{h_1,h_2,f,g}$.

Going from $\mathsf{Hybrid3}^m_{h_1,h_2,f,g}$ *to* $\mathsf{Hybrid4}^m_{h_1,h_2,f,g}$*:* We now wish to remove dependency of the ciphertext on the source. This removes the dependency across the two states containing w and C, which might have led to related tampering of the message. To do this we would like to use the security of our randomness extractor and replace the extracted key k by uniform. The main challenge in doing so is that, the decoded (tampered) message might itself reveal information about the key k. This is a challenge because this information is learnt after the seed s is chosen. This is the main bottleneck of our proof. The way we overcome it is by carefully arguing that the information revealed by the decoded message might be learnt from auxiliary information. Importantly, this auxiliary information is completely independent of s and therefore, we can use extractor security.

Claim. If Ext is an $(n, t, d, l, \varepsilon_4)$ average case extractor, then $\mathsf{Hybrid3}^m_{h_1,h_2,f,g} \approx_{\varepsilon_4} \mathsf{Hybrid4}^m_{h_1,h_2,f,g}$

$\mathsf{Hybrid3}^m_{h_1,h_2,f,g}$:

1. $\tilde{k_{a_1}}||\tilde{k_{a_2}}||\tilde{t_1}||\tilde{t_2}||\tilde{s} \leftarrow \mathsf{NMSim}_{h_1,h_2}$
2. $w \in_R \{0,1\}^n$,$s \in_R \{0,1\}^d$
3. $k = \mathsf{Ext}(w; s)$
4. $C = m \oplus k$
5. $(\tilde{w}, \tilde{C}) = (f(w), g(C))$
6. If $\tilde{k_{a_1}}||\tilde{k_{a_2}}||\tilde{t_1}||\tilde{t_2}||\tilde{s} = same^*$
   - If $\tilde{w} = w$ and $\tilde{C} = C$ output m
   - else output $\perp$
7. else if $\tilde{k_{a_1}}||\tilde{k_{a_2}}||\tilde{t_1}||\tilde{t_2}||\tilde{s} = \perp$, output $\perp$
8. else if $\mathsf{Vrfy}_{\tilde{k_{a_1}}}(\tilde{w}, \tilde{t_1}) = 1 \wedge \mathsf{Vrfy}'_{\tilde{k_{a_2}}}(\tilde{C}, \tilde{t_2}) = 1$ output $\tilde{C} \oplus \mathsf{Ext}(\tilde{w}; \tilde{s})$
9. else output $\perp$

$\mathsf{Hybrid4}^m_{h_1,h_2,f,g}$:

1. $\tilde{k_{a_1}}||\tilde{k_{a_2}}||\tilde{t_1}||\tilde{t_2}||\tilde{s} \leftarrow \mathsf{NMSim}_{h_1,h_2}$
2. $w \in_R \{0,1\}^n$
3. $k \in_R \{0,1\}^l$
4. $C = m \oplus k$
5. $(\tilde{w}, \tilde{C}) = (f(w), g(C))$
6. If $\tilde{k_{a_1}}||\tilde{k_{a_2}}||\tilde{t_1}||\tilde{t_2}||\tilde{s} = same^*$
   - If $\tilde{w} = w$ and $\tilde{C} = C$ output m
   - else output $\perp$
7. else if $\tilde{k_{a_1}}||\tilde{k_{a_2}}||\tilde{t_1}||\tilde{t_2}||\tilde{s} = \perp$, output $\perp$
8. else if $\mathsf{Vrfy}_{\tilde{k_{a_1}}}(\tilde{w}, \tilde{t_1}) = 1 \wedge \mathsf{Vrfy}'_{\tilde{k_{a_2}}}(\tilde{C}, \tilde{t_2}) = 1$ output $\tilde{C} \oplus \mathsf{Ext}(\tilde{w}; \tilde{s})$
9. else output $\perp$

Proof. As explained in the motivation to this claim, we wish to replace the extractor output with a uniform string. But the main challenge in this, is to capture the security, given an auxiliary information.We consider two cases and carefully analyze the auxiliary information that we use in each of them. We show that in both these cases, the auxiliary information is completely independent of s. We then use the average extractor property to argue security. We define two mutually exclusive events:

- Let $Case1$ denote the event that $\tilde{k_{a_1}}||\tilde{k_{a_2}}||\tilde{t_1}||\tilde{t_2}||\tilde{s} = same^*$.
- Let $Case2$ denote the event that $\tilde{k_{a_1}}||\tilde{k_{a_2}}||\tilde{t_1}||\tilde{t_2}||\tilde{s} \neq same^*$.

By Proposition 1, we get:
$\mathbf{SD}\left(\mathsf{Hybrid3}^m_{h_1,h_2,f,g}, \mathsf{Hybrid4}^m_{h_1,h_2,f,g}\right)$

$$\begin{aligned} &\leq Pr[Case1]\ \mathbf{SD}\left(\mathsf{Hybrid3}^m_{h_1,h_2,f,g}|Case1, \mathsf{Hybrid4}^m_{h_1,h_2,f,g}|Case1\right) \\ &+Pr[Case2]\ \mathbf{SD}\left(\mathsf{Hybrid3}^m_{h_1,h_2,f,g}|Case2, \mathsf{Hybrid4}^m_{h_1,h_2,f,g}|Case2\right) \end{aligned} \tag{3}$$

We now use the security of the average case extractor to get the desired statistical closeness in each of the two cases separately. The auxiliary information in each case is different.
Case1 : $\mathbf{\tilde{k_{a_1}}||\tilde{k_{a_2}}||\tilde{t_1}||\tilde{t_2}||\tilde{s} = same^*}$
In this case , the auxiliary information just includes a single bit, indicating whether w is modified or remains the same. So, we first define this indicator function:

$$eq(w) = \begin{cases} 0\ if\ f(w) \neq w \\ 1\ if\ f(w) = w \end{cases}$$

Let the auxiliary information be denoted by $E_1 \equiv (eq(W))$. E_1 is independent of S because E_1 is determined given W and W is independent of S. Now, E_1 and W are correlated and E_1 can take at most two possible values.
Hence, $\tilde{\mathbf{H}}_\infty(W|E_1) \geq \mathbf{H}_\infty(W) - 1 = n - 1$ by Lemma 5. As $n - 1 > t$, by security of average case extractor, we get:

$$E_1, \mathsf{Ext}(W; S) \approx_{\varepsilon_4} E_1, U_l$$

As m is independent of (W, S), we get:

$$m, E_1, \mathsf{Ext}(W; S) \approx_{\varepsilon_4} m, E_1, U_l \tag{4}$$

Now, we wish to apply Lemma 3, for which, we express the output of the hybrids in $Case1$ as a deterministic function of the variables above. Let
$Q_1(m, eq(w), k)$:

- $C = m \oplus k$
- $\tilde{C} = g(C)$
- If $eq(w) = 1$ and $\tilde{C} = C$ output m
- else output $\perp$

Then, the outputs of $\mathsf{Hybrid3}^m_{h_1,h_2,f,g}|Case1$ and $\mathsf{Hybrid4}^m_{h_1,h_2,f,g}|Case1$ are expressible by Q_1 above.

Hence, $Eq.4 \implies$ By Lemma 3, $Q_1(m, E_1, \mathsf{Ext}(W; S)) \approx_{\varepsilon_4} Q_1(m, E_1, U_l)$

$$\text{i.e., } \mathsf{Hybrid3}^m_{h_1,h_2,f,g}|Case1 \approx_{\varepsilon_4} \mathsf{Hybrid4}^m_{h_1,h_2,f,g}|Case1 \tag{5}$$

Case2 : $\tilde{\mathbf{k_{a_1}}}||\tilde{\mathbf{k_{a_2}}}||\tilde{\mathbf{t_1}}||\tilde{\mathbf{t_2}}||\tilde{\mathbf{s}} \neq \mathbf{same}^*$
This case is further divided into two mutually exclusive events of Case2.
Case2a : $\tilde{\mathbf{k_{a_1}}}||\tilde{\mathbf{k_{a_2}}}||\tilde{\mathbf{t_1}}||\tilde{\mathbf{t_2}}||\tilde{\mathbf{s}} = \perp$
Given $\tilde{k_{a_1}}||\tilde{k_{a_2}}||\tilde{t_1}||\tilde{t_2}||\tilde{s} = \perp$ both hybrids output $\perp$ with probability 1. Therefore

$$\mathbf{SD}\left(\mathsf{Hybrid3}^m_{h_1,h_2,f,g}|case2a, \mathsf{Hybrid4}^m_{h_1,h_2,f,g}|case2a\right) = 0 \tag{6}$$

Case2b : $\tilde{\mathbf{k_{a_1}}}||\tilde{\mathbf{k_{a_2}}}||\tilde{\mathbf{t_1}}||\tilde{\mathbf{t_2}}||\tilde{\mathbf{s}} \neq \perp \wedge \tilde{\mathbf{k_{a_1}}}||\tilde{\mathbf{k_{a_2}}}||\tilde{\mathbf{t_1}}||\tilde{\mathbf{t_2}}||\tilde{\mathbf{s}} \neq \mathbf{same}^*$
When $\tilde{k_{a_1}}||\tilde{k_{a_2}}||\tilde{t_1}||\tilde{t_2}||\tilde{s} \neq (same^*, \perp)$, the auxiliary information consists of an indicator of verification of $\tilde{w}$, the simulated authentication key and tag (corresponding to the ciphertext) distributions and the modified encryption key. We first define the indicator of verification bit:

$$Verify(w) = \mathsf{Vrfy}_{\tilde{k_{a_1}}}(f(w), \tilde{t_1})$$

Now, let the auxiliary information be denoted by $E_2 \equiv (Verify(W), \tilde{K_{a_2}}, \tilde{T_2}, \mathsf{Ext}(\tilde{W}; \tilde{S}))$, where $\tilde{K_{a_1}}, \tilde{K_{a_2}}, \tilde{T_1}, \tilde{T_2}, \tilde{S}$ denote the distributions on the authentication key, tag spaces and the seed, when sampled from the simulator conditioned on the event Case2b. E_2 is clearly a deterministic

function of $\tilde{K_{a_1}}, \tilde{K_{a_2}}, \tilde{W}, \tilde{T_1}, \tilde{T_2}, \tilde{S}$, all of which are independent of S (as we use the simulator). Hence, E_2 is independent of S. Now, E_2 and W are correlated. E_2 can take at most $2^{1+\tau_2+\delta_2+l}$ possible values.
Hence, $\tilde{\mathbf{H}}_\infty(W|E_2) \geq \mathbf{H}_\infty(W) - (1+\tau_2+\delta_2+l) = n - (1+\tau_2+\delta_2+l)$, by Lemma 5. As $n-(1+\tau_2+\delta_2+l) > t$ (due to the way we set parameters in section 4.5), by security of average case extractor, we get:

$$E_2, \mathsf{Ext}(W;S) \approx_{\varepsilon_4} E_2, U_l$$

As m is independent of (W,S), we get:

$$m, E_2, \mathsf{Ext}(W;S) \approx_{\varepsilon_4} m, E_2, U_l \tag{7}$$

Now, we wish to apply Lemma 3 and again, we express the outputs of $\mathsf{Hybrid3}^m_{h_1,h_2,f,g}|Case2b$ and $\mathsf{Hybrid4}^m_{h_1,h_2,f,g}|Case2b$ as a deterministic function of the variables above. Define function Q_2 as follows.
$Q_2(m, Verify(w), \tilde{k_{a_2}}, \tilde{t_2}, \mathsf{Ext}(\tilde{w};\tilde{s}), k)$:

- $C = m \oplus k$
- $\tilde{C} = g(C)$
- If $Verify(w) = 1$ and $\mathsf{Vrfy}'_{\tilde{k_{a_2}}}(\tilde{C}, \tilde{t_2}) = 1$ output $\tilde{C} \oplus \mathsf{Ext}(\tilde{w};\tilde{s})$
- else output $\perp$

Then, the outputs of $\mathsf{Hybrid3}^m_{h_1,h_2,f,g}|Case2b$ and $\mathsf{Hybrid4}^m_{h_1,h_2,f,g}|Case2b$ are expressible by Q_2 above.

$$\begin{aligned} \text{Hence, } Eq.7 \implies & \text{By Lemma 3, } Q_2(m, E_2, \mathsf{Ext}(W;S)) \approx_{\varepsilon_4} Q_2(m, E_2, U_l) \\ & \text{i.e., } \mathsf{Hybrid3}^m_{h_1,h_2,f,g}|Case2b \approx_{\varepsilon_4} \mathsf{Hybrid4}^m_{h_1,h_2,f,g}|Case2b \end{aligned} \tag{8}$$

Hence, by Proposition 1, Equations 3, 5, 6 and 8 above, we get:

$$\mathsf{Hybrid3}^m_{h_1,h_2,f,g} \approx_{\varepsilon_4} \mathsf{Hybrid4}^m_{h_1,h_2,f,g}$$

Remark on Auxiliary Information. We first observe that the auxiliary information in both the cases contains the additional information required to get the outputs of the hybrids, which are independent of the seed. In $Case1$, we just have a single bit of auxiliary information, which is independent of s. In $Case2$ however, as we add the verification bit to E_2, it is important that this verification check is independent of s. If we authenticate w and C together, under a single MAC, then the verify check would be dependent on C as well, which in turn depends on s in the third hybrid. So, it is important that we authenticate w and C using separate one time MAC. Then, we give the modified authentication key and tag corresponding to C in E_2, which are independent of s.

Rewriting $\mathsf{Hybrid4}^m_{h_1,h_2,f,g}$ *as* $\mathsf{Hybrid5}^m_{h_1,h_2,f,g}$*:* We again rewrite $\mathsf{Hybrid4}^m_{h_1,h_2,f,g}$ such that we first choose the encryption key k uniformly at random, to get $\mathsf{Hybrid5}^m_{h_1,h_2,f,g}$. This reordering of steps is to stress on the fact that, we have now removed the dependency of the encryption key on w and s and we sample it uniformly at random.

$\mathsf{Hybrid4}^m_{h_1,h_2,f,g}$:

1. $\tilde{k_{a_1}}||\tilde{k_{a_2}}||\tilde{t_1}||\tilde{t_2}||\tilde{s} \leftarrow \mathsf{NMSim}_{h_1,h_2}$
2. $w \in_R \{0,1\}^n$
3. $k \in_R \{0,1\}^l$
4. $C = m \oplus k$
5. $(\tilde{w}, \tilde{C}) = (f(w), g(C))$
6. If $\tilde{k_{a_1}}||\tilde{k_{a_2}}||\tilde{t_1}||\tilde{t_2}||\tilde{s} = same^*$
   - If $\tilde{w} = w$ and $\tilde{C} = C$ output m
   - else output $\perp$
7. else if $\tilde{k_{a_1}}||\tilde{k_{a_2}}||\tilde{t_1}||\tilde{t_2}||\tilde{s} = \perp$, output $\perp$
8. else if $\mathsf{Vrfy}_{\tilde{k_{a_1}}}(\tilde{w}, \tilde{t_1}) = 1 \wedge \mathsf{Vrfy}'_{\tilde{k_{a_2}}}(\tilde{C}, \tilde{t_2}) = 1$ output $\tilde{C} \oplus \mathsf{Ext}(\tilde{w}; \tilde{s})$
9. else output $\perp$

$\mathsf{Hybrid5}^m_{h_1,h_2,f,g}$:

1. $k \in_R \{0,1\}^l$
2. $C = m \oplus k$
3. $w \in_R \{0,1\}^n$
4. $(\tilde{w}, \tilde{C}) = (f(w), g(C))$
5. $\tilde{k_{a_1}}||\tilde{k_{a_2}}||\tilde{t_1}||\tilde{t_2}||\tilde{s} \leftarrow \mathsf{NMSim}_{h_1,h_2}$
6. If $\tilde{k_{a_1}}||\tilde{k_{a_2}}||\tilde{t_1}||\tilde{t_2}||\tilde{s} = same^*$
   - If $\tilde{w} = w$ and $\tilde{C} = C$ output m
   - else output $\perp$
7. else if $\tilde{k_{a_1}}||\tilde{k_{a_2}}||\tilde{t_1}||\tilde{t_2}||\tilde{s} = \perp$, output $\perp$
8. else if $\mathsf{Vrfy}_{\tilde{k_{a_1}}}(\tilde{w}, \tilde{t_1}) = 1 \wedge \mathsf{Vrfy}'_{\tilde{k_{a_2}}}(\tilde{C}, \tilde{t_2}) = 1$ output $\tilde{C} \oplus \mathsf{Ext}(\tilde{w}; \tilde{s})$
9. else output $\perp$

As we have only reordered the steps without changing any of the distributions, clearly $\mathsf{Hybrid4}^m_{h_1,h_2,f,g} \equiv \mathsf{Hybrid5}^m_{h_1,h_2,f,g}$.

Going from $\mathsf{Hybrid5}^m_{h_1,h_2,f,g}$ *to* $\mathsf{Hybrid6}^m_{h_1,h_2,f,g}$*:* In the final hybrid, $\mathsf{Hybrid6}^m_{h_1,h_2,f,g}$, we use the perfect security of the one time pad to remove the dependency of C (and hence, of $\tilde{C}$) on m. This gives us the simulated view, independent of m.

Claim. $\mathsf{Hybrid5}^m_{h_1,h_2,f,g} \equiv \mathsf{Hybrid6}^m_{h_1,h_2,f,g}$ by perfect security of One Time Pad encryption.

Proof. We begin by expressing the hybrid outputs as a deterministic function of the message and the ciphertext. Define function Q_3 as follows:
$Q_3(m, C)$:

- $\tilde{k_{a_1}}||\tilde{k_{a_2}}||\tilde{t_1}||\tilde{t_2}||\tilde{s} \leftarrow \mathsf{NMSim}_{h_1,h_2}$
- $w \in_R \{0,1\}^n$
- $(\tilde{w}, \tilde{C}) = (f(w), g(C))$
- If $\tilde{k_{a_1}}||\tilde{k_{a_2}}||\tilde{t_1}||\tilde{t_2}||\tilde{s} = same^*$
  - If $\tilde{w} = w$ and $\tilde{C} = C$ output m
  - else output $\perp$

$\mathsf{Hybrid5}^m_{h_1,h_2,f,g}$:	$\mathsf{Hybrid6}^m_{h_1,h_2,f,g}$:
1. $k \in_R \{0,1\}^l$ 2. $C = m \oplus k$ 3. $w \in_R \{0,1\}^n$ 4. $(\tilde{w}, \tilde{C}) = (f(w), g(C))$ 5. $\tilde{k_{a_1}}\|\tilde{k_{a_2}}\|\tilde{t_1}\|\tilde{t_2}\|\tilde{s} \leftarrow \mathsf{NMSim}_{h_1,h_2}$ 6. If $\tilde{k_{a_1}}\|\tilde{k_{a_2}}\|\tilde{t_1}\|\tilde{t_2}\|\tilde{s} = same^*$ • If $\tilde{w} = w$ and $\tilde{C} = C$ output m • else output $\perp$ 7. else if $\tilde{k_{a_1}}\|\tilde{k_{a_2}}\|\tilde{t_1}\|\tilde{t_2}\|\tilde{s} = \perp$, output $\perp$ 8. else if $\mathsf{Vrfy}_{\tilde{k_{a_1}}}(\tilde{w}, \tilde{t_1}) = 1 \wedge \mathsf{Vrfy'}_{\tilde{k_{a_2}}}(\tilde{C}, \tilde{t_2}) = 1$ output $\tilde{C} \oplus \mathsf{Ext}(\tilde{w}; \tilde{s})$ 9. else output $\perp$	1. $k \in_R \{0,1\}^l$ 2. $C = 0 \oplus k$ 3. $w \in_R \{0,1\}^n$ 4. $(\tilde{w}, \tilde{C}) = (f(w), g(C))$ 5. $\tilde{k_{a_1}}\|\tilde{k_{a_2}}\|\tilde{t_1}\|\tilde{t_2}\|\tilde{s} \leftarrow \mathsf{NMSim}_{h_1,h_2}$ 6. If $\tilde{k_{a_1}}\|\tilde{k_{a_2}}\|\tilde{t_1}\|\tilde{t_2}\|\tilde{s} = same^*$ • If $\tilde{w} = w$ and $\tilde{C} = C$ output m • else output $\perp$ 7. else if $\tilde{k_{a_1}}\|\tilde{k_{a_2}}\|\tilde{t_1}\|\tilde{t_2}\|\tilde{s} = \perp$, output $\perp$ 8. else if $\mathsf{Vrfy}_{\tilde{k_{a_1}}}(\tilde{w}, \tilde{t_1}) = 1 \wedge \mathsf{Vrfy'}_{\tilde{k_{a_2}}}(\tilde{C}, \tilde{t_2}) = 1$ output $\tilde{C} \oplus \mathsf{Ext}(\tilde{w}; \tilde{s})$ 9. else output $\perp$

– else if $\tilde{k_{a_1}}\|\tilde{k_{a_2}}\|\tilde{t_1}\|\tilde{t_2}\|\tilde{s} = \perp$, output $\perp$
– else If $\mathsf{Vrfy}_{\tilde{k_{a_1}}}(\tilde{w}, \tilde{t_1}) = 1$ and $\mathsf{Vrfy'}_{\tilde{k_{a_2}}}(\tilde{C}, \tilde{t_2}) = 1$ Output $\tilde{C} \oplus \mathsf{Ext}(\tilde{w}, \tilde{s})$
– else Output $\perp$

Replace sequence of steps 3-9 with an output of $Q_3(m, C)$ in both the hybrids. By perfect security of OTP encryption, for any message m and a uniformly random key k

$$(m, (m \oplus k)) \equiv (m, (0 \oplus k))$$

$$Q_3(m, (m \oplus k)) \equiv Q_3(m, (0 \oplus k))$$

$$\mathsf{Hybrid5}^m_{h_1,h_2,f,g} \equiv \mathsf{Hybrid6}^m_{h_1,h_2,f,g}$$

Combining results of above claims 4.3, 4.3, 4.3 and 4.3, we get

$$\begin{aligned} Tamper^m_{h_1,h_2,f,g} &\approx_{\varepsilon_1} \mathsf{Hybrid1}^m_{h_1,h_2,f,g} \approx_{\varepsilon_2+\varepsilon_3} \mathsf{Hybrid2}^m_{h_1,h_2,f,g} \equiv \mathsf{Hybrid3}^m_{h_1,h_2,f,g} \\ \mathsf{Hybrid3}^m_{h_1,h_2,f,g} &\approx_{\varepsilon_4} \mathsf{Hybrid4}^m_{h_1,h_2,f,g} \equiv \mathsf{Hybrid5}^m_{h_1,h_2,f,g} \\ \mathsf{Hybrid5}^m_{h_1,h_2,f,g} &\equiv \mathsf{Hybrid6}^m_{h_1,h_2,f,g} \equiv Copy^m_{Sim_{h_1,h_2,f,g}} \\ \implies Tamper^m_{h_1,h_2,f,g} &\approx_{\varepsilon_1+\varepsilon_2+\varepsilon_3+\varepsilon_4} Copy^m_{Sim_{h_1,h_2,f,g}} \end{aligned}$$

4.4 Rate and Error analysis

We now present the details of the rate of the code as well as the error it achieves. As we are encoding the seed of the extractor using the underlying non-malleable code, it is important that the strong extractor we use has short seed length. This is guaranteed by the following lemma.

Lemma 7. *[19] For every constant $\nu > 0$ all integers $n \geq t$ and all $\epsilon \geq 0$, there is an explicit (efficient) $(n, t, d, l, \epsilon)-$strong extractor with $l = (1-\nu)t - \mathcal{O}(\log(n) + \log(\frac{1}{\epsilon}))$ and $d = \mathcal{O}(\log(n) + \log(\frac{1}{\epsilon}))$.*

Now, as we give some auxiliary information about the source, we require the security of the extractor to hold, even given this information. Hence, we use average case extractors, given in the following lemma.

Lemma 8. *[13] For any $\mu > 0$, if Ext is a (worst case)$(n, t, d, l, \epsilon)-$strong extractor, then Ext is also an average-case $(n, t+\log(\frac{1}{\mu}), d, l, \epsilon+\mu)$ strong extractor.*

We now combine the Lemmata 7 and 8 to get an average case extractor with optimal seed length.

Corollary 1. *For any $\mu > 0$ and every constant $\nu > 0$ all integers $n \geq t$ and all $\epsilon \geq 0$, there is an explicit (efficient) $(n, t+\log(\frac{1}{\mu}), d, l, \epsilon+\mu)-$ average case strong extractor with $l = (1-\nu)t - \mathcal{O}(\log(n) + \log(\frac{1}{\epsilon}))$ and $d = \mathcal{O}(\log(n) + \log(\frac{1}{\epsilon}))$.*

Now, we also encode the authentication keys and tags using the underlying non-malleable code. Hence, we require them to have short lengths. This is guaranteed by the following lemma.

Lemma 9. *[[14], Lemma 26] For any $n', \varepsilon_2 > 0$ there is an efficient ε_2-secure one time MAC with $\delta \leq (\log(n') + \log(\frac{1}{\varepsilon_2}))$, $\tau \leq 2\delta$, where τ, n', δ are key, message, tag length respectively.*

We refer the reader to [12] for a construction satisfying these parameters.

4.5 Setting parameters

- Set all the error parameters $\epsilon, \mu, \epsilon_1\epsilon_2, \epsilon_3 = 2^{-\lambda}$ and $\epsilon_4 = \epsilon + \mu$
- The codeword of the construction in Figure 1 has four states: a two-split state NMC codeword(L, R), source(w), ciphertext(C). In order to estimate the rate we need to estimate the length of each of these states.
- Let l be the length of the message for construction in Figure 1. As we are using the one-time pad encryption scheme, $|C| = l$.
- We now estimate the length of the source - n. Although the source has full entropy (i.e., uniformly random), as we can see from the proof of Claim 4.3, we do reveal some auxiliary information which needs to be taken into account. The amount of auxiliary information that is revealed in Claim 4.3 can be upper bounded by the amount of auxiliary information that is revealed in Case 2b. So we need to upper bound this entropy loss. This auxiliary information consists of an indicator bit, the key, tag of cipher text, and an extractor output. Of these, we know that the extractor output is of length l bits and the indicator is just one bit. So we need to estimate the length of the authentication key, tag pair of the cipher text.

– τ_2, δ_2- length of key, tag to authenticate cipher text(C) of length l.
  - Applying Lemma 9 with $n' = l, \epsilon_3 = 2^{-\lambda}$ gives $\delta_2 \leq (\log(l) + \lambda)$ and $\tau_2 \leq 2\delta_2$. Therefore
$$\tau_2 + \delta_2 \leq 3(\log(l) + \lambda) \tag{9}$$
– By Lemma 5, the average entropy of the source given auxiliary information is at least $n - \underbrace{(1 + \tau_2 + \delta_2 + l)}_{length\ of\ aux\ info}$ which is at least $n - (1 + 3\log(l) + 3\lambda + l)$.
– Also we need to make sure that the average entropy we are left with is at least the entropy threshold $(t + \log(\frac{1}{\mu}))$. So we need to estimate t
  - By Corollary 1, we have $t = (l + \mathcal{O}(\log(n) + \log(\frac{1}{\epsilon})))\frac{1}{1-\nu}$
  - It is necessary and sufficient if $n - (1 + 3\log(l) + 3\lambda + l) > t + \log(\frac{1}{\mu})$

$$\Rightarrow n \geq (1 + \frac{1}{1-\nu})l + 3\log(l) + 4(\lambda) + \mathcal{O}(\log(n) + \lambda)$$

    As ν can be very small constant thats close to 0, fixing $n = (2 + \zeta)l + \mathcal{O}(\log(l) + \lambda)$ for some constant ζ close to 0, would satisfy the above equation.

  We now estimate the length of the codeword of the underlying NMC. We encode an authentication key, tag pair of ciphertext , authentication key, tag pair of the source, extractor seed. The length of the authentication key, tag pair of ciphertext is given in Equation 9. We estimate the lengths of the remaining variables below.
– d- seed length of the extractor.
  - From Corollary 1. we have $d = \mathcal{O}(\log(n) + \log(\frac{1}{\epsilon}))$.
  - Substituting $n = (2 + \zeta)l + \mathcal{O}(\log(l) + \lambda)$ and $\epsilon = 2^{-\lambda}$ gives

$$d = \mathcal{O}(\log((2+\zeta)l + \mathcal{O}(\log(l) + \lambda)) + \lambda) = \mathcal{O}(\log(l + \lambda) + \lambda) = \mathcal{O}(\log(l) + \lambda) \tag{10}$$
– τ_1, δ_1- length of key, tag to authenticate source(W) of length n.
  - Applying Lemma 9 with $n' = n, \epsilon_2 = 2^{-\lambda}$ gives $\delta_1 \leq (\log(n) + \lambda)$ and $\tau_1 \leq 2\delta_1$. Therefore
$$\tau_1 + \delta_1 \leq 3(\log(n) + \lambda) \tag{11}$$
– $\alpha = \tau_1 + \tau_2 + \delta_1 + \delta_2 + d$ - length of message that we are encoding using NMC in [22].
  - By Equations 11, 9, 10

$$\alpha \leq (c + 1)(\log(l) + \lambda) + 3(\log(l) + \lambda) + 3(\log(n) + \lambda)$$

    By the same argument as in equation 10

$$\alpha = \mathcal{O}(\log(l) + \lambda) \tag{12}$$

– 2β- codeword length of NMC in [22].
  - By Equation 1, we have $\beta = \mathcal{O}(\alpha \log(\alpha))$
    By Equation 12, we have

$$\alpha \log(\alpha) = \mathcal{O}((\log(l)+\lambda).\log(\log(l)+\lambda)) = \mathcal{O}((\log(l))^2+\lambda.\log(\lambda)+2.\lambda.\log(l))) \tag{13}$$

    Therefore,

$$\beta = \mathcal{O}((\log(l))^2 + \lambda \log(\lambda) + 2\lambda \log(l))) \tag{14}$$

– Now we have upper bound on the length of all states of the codeword in terms of l and λ.

4.6 Rate

Let R denote the rate of proposed construction.

$$R = \frac{l}{2\beta + n + l}$$

Substituting n and β (by Eq. 14)

$$= \frac{l}{\mathcal{O}((\log(l))^2 + \lambda.\log(\lambda) + 2.\lambda.\log(l)) + (2+\zeta)l + \mathcal{O}(\log(l) + \lambda) + l}$$

For some constant c

$$\geq \frac{l}{c((\log(l))^2 + \lambda \log(\lambda) + 2\lambda \log(l)) + (2+\zeta)l + \mathcal{O}(\log(l) + \lambda) + l}$$

$$= \frac{1}{\frac{c((\log(l))^2 + \lambda \log(\lambda) + 2\lambda \log(l)) + (2+\zeta)l + \mathcal{O}(\log(l) + \lambda) + l}{l}}$$

For large l

$$= \frac{1}{\frac{c(\lambda \log(\lambda) + 2\lambda \log(l)) + \mathcal{O}(\lambda)}{l} + 3 + \zeta}$$

For $\lambda = o\left(\frac{l}{\log l}\right)$

$$= \frac{1}{3+\zeta}$$

Construction in Figure 1 achieves rate that is at least $\frac{1}{3+\zeta}$, for some ζ very close to 0.

4.7 Error

Error of the proposed construction is $\varepsilon_1 + \varepsilon_2 + \varepsilon_3 + \varepsilon_4 = 5(2^{-\lambda})$. Because $\lambda = o\left(\frac{l}{\log l}\right)$ the error will be at least $2^{-\frac{l}{\log l}}$. For any $\rho > 0$, fixing $\lambda = \frac{l}{\log^{\rho+1} l}$, the error would be at most $5.2^{-\frac{l}{\log^{\rho+1} l}}$. Setting $\kappa = \lambda - \log 5$ the error would be $2^{-\kappa} = 2^{-\Omega(l/\log^{\rho+1} l)}$.

Conclusion

In this work, we constructed an efficient non-malleable code in the t-split-state model, for $t = 4$, that achieves a constant rate of $\frac{1}{3+\zeta}$, for any constant $\zeta > 0$ and error $2^{-\Omega(\ell/\log^{c+1} \ell)}$, where ℓ is the length of the message and $c > 0$ is a constant. This improves the constant-rate constructions of Cheraghchi and Guruswami [8] (by bringing down the number of states from n to 4) and Chattopadhyay and Zuckerman [6](by making the "constant" in the rate explicit and by bringing down the number of states from 10 to 4). We stress that as is the case with all information-theoretic primitives, optimizing constant factors in achieving key parameters, such as, in this case, the rate/number of states etc., is both crucial and challenging.

While we obtain our specific parameters by using the 2 state non-malleable code construction due to [22], our techniques are general and uses the underlying NMC in a black-box. Hence, our construction can be generalized to obtain a $(t+2)$-state NMC from any t-state NMC, leading to interesting trade-offs between the rate vs the number of states depending on the parameters of underlying NMC (Appendix A).

An interesting open problem would be to see if our techniques can be used to improve the rate of non-malleable codes with special features such as "locality" [11,5,10], security against continuous tampering [17,21], and leakage-resilience [2].

Acknowledgment

We thank Yevgeniy Dodis for insightful comments related to the generalization in Appendix A We also thank the anonymous referees for several helpful comments.

A Appendix I

A.1 Building constant rate $(t+2)$ - state NMC from any t-state NMC with inverse polynomial rate

Theorem 2. *Let* $\mathsf{NMEnc}, \mathsf{NMDec}$ *be an* ε_1*-secure* $t-$ *split state non-malleable code with rate* $\omega\left(\frac{1}{\alpha^a}\right)$*, for some constant* a *and message length* α*. The algorithms* $(\mathsf{Tag}, \mathsf{Vrfy}), (\mathsf{Tag}', \mathsf{Vrfy}'), \mathsf{Ext}$ *be as a specified in Section 4.1.*
For any constant $\zeta > 0$*, messages of length* l*, any* κ *such that* $\kappa = o\left(\frac{l}{l^{a+1}}\right)$*, the* $(t+2)-$ *split state construction in figure below has block length* $(3+\zeta)l + o(l)$*, there by achieves asymptotic rate* $\frac{1}{3+\zeta}$ *and error* $2^{-\kappa}$*.*

$\mathsf{Enc}(m)$:	$\mathsf{Dec}(L_1, L_2, \cdots, L_t, w, C)$:
– $w \in_R \{0,1\}^n,\ s \in_R \{0,1\}^d$ – $k_{a_1} \in_R \{0,1\}^{\tau_1}, k_{a_2} \in_R \{0,1\}^{\tau_2}$ – $k = \mathsf{Ext}(w, s)$ – $C = m \oplus k$ – $t_1 = \mathsf{Tag}_{k_{a_1}}(w)$, $t_2 = \mathsf{Tag}'_{k_{a_2}}(C)$ – $(L_1, L_2, \cdots, L_t) =$ $\mathsf{NMEnc}(k_{a_1}\|k_{a_2}\|t_1\|t_2\|s)$ – Output :$(L_1, L_2, \cdots, L_t, w, C)$	– $k_{a_1}\|k_{a_2}\|t_1\|t_2\|s =$ $\mathsf{NMDec}(L_1, L_2, \cdots, L_t)$ – If $k_{a_1}\|k_{a_2}\|t_1\|t_2\|s = \perp$ output $\perp$ – else if $\mathsf{Vrfy}_{k_{a_1}}(w, t_1) = 1$ $\wedge\ \mathsf{Vrfy}'_{k_{a_2}}(C, t_2) = 1$ output $C \oplus \mathsf{Ext}(w, s)$ – else output $\perp$

Proof. The construction in figure above is a secure $(t+2)$- state NMC. The security proof is similar to the proof in Section 4.3.
Set parameters $\kappa, \epsilon, \mu, \epsilon_1\epsilon_2, \epsilon_3, \epsilon_4, n, \alpha$ in terms of l, λ as in Section 4.5.

– Let β be length of t-state codeword of $(\mathsf{NMEnc}, \mathsf{NMDec})$ for messages of length α.
– $\beta = O(\alpha^{a+1})$
– The rate r of the $(t+2)-$state NMC (Enc, Dec) is

$$r = \frac{l}{\beta + n + l}$$

Substituting n and β

$$= \frac{l}{\mathcal{O}(\alpha^{a+1}) + (2+\zeta)l + \mathcal{O}(\log(l) + \lambda) + l}$$

Substituting $\alpha = \mathcal{O}(\log(l) + \lambda)$

$$= \frac{l}{\mathcal{O}((\log(l) + \lambda)^{a+1}) + (2+\zeta)l + \mathcal{O}(\log(l) + \lambda) + l}$$

For some constant c

$$\geq \frac{1}{\frac{c((\log(l)+\lambda)^{a+1})+(2+\zeta)l+\mathcal{O}(\log(l)+\lambda)+l}{l}}$$

For large l

$$= \frac{1}{\frac{c((\log(l)+\lambda)^{a+1})+\mathcal{O}(\lambda)}{l}+3+\zeta}$$

For $\lambda = o\left(\frac{l}{l^{a+1}}\right)$ $\qquad r = \frac{1}{3+\zeta}$

Error analysis is similar to analysis in Section 4.7.

B Appendix II

Lemma 10. *If* $\alpha = \Omega(\frac{\beta}{\log(\beta)})$*, then* $\beta = \mathcal{O}(\alpha.\log(\alpha))$

Proof. By the definition of Ω, $\exists$ a constant $c > 0$ such that for large α, β

$$0 \leq c.\frac{\beta}{\log(\beta)} \leq \alpha \tag{15}$$

$$c\beta \leq \alpha.\log(\beta)$$

$$c\beta \leq \alpha\sqrt{\beta}$$

If $c \geq 1$

$$\sqrt{\beta} \leq \alpha$$

$$\log(\beta) \leq 2.\log(\alpha)$$

Multiplying with Eq 15, we get

$$0 \leq \frac{c}{2}.\beta \leq \alpha\log(\alpha) \tag{16}$$

If $c < 1$, let $c' = \frac{1}{c}$

$$\sqrt{\beta} \leq c'.\alpha$$

$$\log(\beta) \leq 2(\log(c') + \log(\alpha))$$

$$\log(\beta) \leq 4.\log(\alpha)$$

Multiplying with Eq 15

$$0 \leq \frac{c}{4}.\beta \leq \alpha\log(\alpha) \tag{17}$$

In either case, for large α, β, for a constant $\frac{c}{4} > 0$

$$0 \leq \frac{c}{4}.\beta \leq \alpha\log(\alpha)$$

$$\implies \alpha\log(\alpha) = \Omega(\beta)$$

$$\implies \beta = \mathcal{O}(\alpha\log(\alpha))$$

References

1. D. Aggarwal, S. Agrawal, D. Gupta, H. K. Maji, O. Pandey, and M. Prabhakaran. Optimal computational split-state non-malleable codes. In *Theory of Cryptography - 13th International Conference, TCC 2016-A, Tel Aviv, Israel, January 10-13, 2016, Proceedings, Part II*, pages 393–417, 2016.
2. D. Aggarwal, Y. Dodis, T. Kazana, and M. Obremski. Non-malleable reductions and applications. In *Proceedings of the Forty-Seventh Annual ACM on Symposium on Theory of Computing, STOC 2015, Portland, OR, USA, June 14-17, 2015*, pages 459–468, 2015.
3. D. Aggarwal, Y. Dodis, and S. Lovett. Non-malleable codes from additive combinatorics. In *Symposium on Theory of Computing, STOC 2014, New York, NY, USA, May 31 - June 03, 2014*, pages 774–783, 2014.
4. J. Carter and M. Wegman. Universal classes of hash functions. *Journal of Computer and System Sciences*, 18:143–154, 1979.
5. N. Chandran, B. Kanukurthi, and S. Raghuraman. Information-theoretic local non-malleable codes and their applications. In *Theory of Cryptography - 13th International Conference, TCC 2016-A, Tel Aviv, Israel, January 10-13, 2016, Proceedings, Part II*, pages 367–392, 2016.
6. E. Chattopadhyay and D. Zuckerman. Non-malleable codes against constant split-state tampering. In *55th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2014, Philadelphia, PA, USA, October 18-21, 2014*, pages 306–315, 2014.
7. M. Cheraghchi and V. Guruswami. Capacity of non-malleable codes. In *Innovations in Theoretical Computer Science, ITCS'14, Princeton, NJ, USA, January 12-14, 2014*, pages 155–168, 2014.
8. M. Cheraghchi and V. Guruswami. Non-malleable coding against bit-wise and split-state tampering. In *Theory of Cryptography - 11th Theory of Cryptography Conference, TCC 2014, San Diego, CA, USA, February 24-26, 2014. Proceedings*, pages 440–464, 2014.
9. S. Coretti, U. Maurer, B. Tackmann, and D. Venturi. From single-bit to multi-bit public-key encryption via non-malleable codes. *IACR Cryptology ePrint Archive*, 2014:324, 2014.
10. D. Dachman-Soled, M. Kulkarni, and A. Shahverdi. Tight upper and lower bounds for leakage-resilient, locally decodable and updatable non-malleable codes. *IACR Cryptology ePrint Archive*, 2017:15, 2017.
11. D. Dachman-Soled, F. Liu, E. Shi, and H. Zhou. Locally decodable and updatable non-malleable codes and their applications. *IACR Cryptology ePrint Archive*, 2014:663, 2014.
12. Y. Dodis, B. Kanukurthi, J. Katz, L. Reyzin, and A. Smith. Robust fuzzy extractors and authenticated key agreement from close secrets. *IEEE Transactions on Information Theory*, 2012.
13. Y. Dodis, R. Ostrovsky, L. Reyzin, and A. Smith. Fuzzy extractors: How to generate strong keys from biometrics and other noisy data. *SIAM Journal on Computing*, 38(1):97–139, 2008. arXiv:cs/0602007.
14. Y. Dodis and D. Wichs. Non-malleable extractors and symmetric key cryptography from weak secrets. In *Proceedings of the Forty-First Annual ACM Symposium on Theory of Computing*, pages 601–610, Bethesda, Maryland, 31 May–2 June 2009.
15. S. Dziembowski, T. Kazana, and M. Obremski. Non-malleable codes from two-source extractors. In *Advances in Cryptology - CRYPTO 2013 - 33rd Annual*

Cryptology Conference, Santa Barbara, CA, USA, August 18-22, 2013. Proceedings, Part II, pages 239–257, 2013.
16. S. Dziembowski, K. Pietrzak, and D. Wichs. Non-malleable codes. In *Innovations in Computer Science - ICS 2010, Tsinghua University, Beijing, China, January 5-7, 2010. Proceedings*, pages 434–452, 2010.
17. S. Faust, P. Mukherjee, J. B. Nielsen, and D. Venturi. Continuous non-malleable codes. In *Theory of Cryptography - 11th Theory of Cryptography Conference, TCC 2014, San Diego, CA, USA, February 24-26, 2014. Proceedings*, pages 465–488, 2014.
18. V. Goyal, O. Pandey, and S. Richelson. Textbook non-malleable commitments. In *Proceedings of the 48th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2016, Cambridge, MA, USA, June 18-21, 2016*, pages 1128–1141, 2016.
19. V. Guruswami, C. Umans, and S. P. Vadhan. Unbalanced expanders and randomness extractors from Parvaresh-Vardy codes. In *IEEE Conference on Computational Complexity*, pages 96–108, 2007.
20. J. Håstad, R. Impagliazzo, L. Levin, and M. Luby. Construction of pseudorandom generator from any one-way function. *SIAM Journal on Computing*, 28(4):1364–1396, 1999.
21. Z. Jafargholi and D. Wichs. Tamper detection and continuous non-malleable codes. In *Theory of Cryptography - 12th Theory of Cryptography Conference, TCC 2015, Warsaw, Poland, March 23-25, 2015, Proceedings, Part I*, pages 451–480, 2015.
22. X. Li. Improved non-malleable extractors, non-malleable codes and independent source extractors. In *Symposium on Theory of Computing, STOC 2017, Montreal, Canada, June 19-23, 2017*, 2017.
23. F. Liu and A. Lysyanskaya. Tamper and leakage resilience in the split-state model. *IACR Cryptology ePrint Archive*, 2012:297, 2012.
24. N. Nisan and D. Zuckerman. Randomness is linear in space. *Journal of Computer and System Sciences*, 52(1):43–53, 1996.
25. D. R. Stinson. Universal hash families and the leftover hash lemma, and applications to cryptography and computing. *Journal of Combinatorial Mathematics and Combinatorial Computing*, 42:3–31, 2002. Available at `http://www.cacr.math.uwaterloo.ca/~dstinson/publist.html`.
26. S. Vadhan. *Pseudorandomness*. Foundations and Trends in Theoretical Computer Science. Now Publishers, 2012. Available at `http://people.seas.harvard.edu/~salil/pseudorandomness/`.