Locally Reconstructable Non-malleable Secret Sharing

Bhavana Kanukurthi * Sai Lakshmi Bhavana Obbattu ** Sruthi Sekar*** Jenit Tomy†

Abstract. Non-malleable secret sharing (NMSS) schemes, introduced by Goyal and Kumar (STOC 2018), ensure that a secret m can be distributed into shares $m_1, \cdots, m_n$ (for some n), such that any t (a parameter $\leq n$) shares can be reconstructed to recover the secret m, any $t-1$ shares doesn't leak information about m and even if the shares that are used for reconstruction are tampered, it is guaranteed that the reconstruction of these tampered shares will either result in the original m or something independent of m. Since their introduction, non-malleable secret sharing schemes sparked a very impressive line of research.

In this work, we introduce a feature of local reconstructability in NMSS, which allows reconstruction of any portion of a secret by reading just a few locations of the shares. This is a useful feature, especially when the secret is long or when the shares are stored in a distributed manner on a communication network. In this work, we give a compiler that takes in any non-malleable secret sharing scheme and compiles it into a *locally reconstructable non-malleable secret sharing scheme*. To secret share a message consisting of k blocks of length ρ each, our scheme would only require reading $\rho + \log k$ bits (in addition to a few more bits, whose quantity is independent of ρ and k) from each party's share (of a reconstruction set) to locally reconstruct a single block of the message.

We show an application of our locally reconstructable non-malleable secret sharing scheme to a computational *non-malleable secure message transmission scheme in the pre-processing model*, with an improved communication complexity, when transmitting multiple messages.

1 Introduction

Secret Sharing Schemes. Secret sharing schemes [Sha79,Bla79] allow a dealer holding a secret m, to distribute the secret across a set of parties $P_1, P_2, \cdots, P_n$ as shares $m_1, m_2, \cdots, m_n$ such that subsets of parties authorised by the dealer can reconstruct the secret m and all the other subsets of parties have no information about the secret. Secret sharing schemes are fundamental building blocks in secure computation.

Non-malleability. Non-malleable secret sharing schemes(NMSS) were introduced by Goyal and Kumar [GK18a]. They ensure that if the shares of an authorised set are tampered, then reconstruction of these tampered shares is either same as the original secret or it is some independent of the original secret. Since their introduction, NMSS received wide attention with a long line of work [GK18a,GK18b,SV19,BS19,FV19,ADN+19,BFV19,KMS19,LCG+19,KMZ20], [CGGL20,CKOS20]. NMSS are built specific to the class of tampering that the shares undergo. This is because without any restriction of the tampering it is impossible to build NMSS as the tampering function can take all the shares of an authorised set and reconstruct m, compute the shares of $m+1$ with respect to this authorised set and sets them as the tampered shares. In this case, the reconstruction of the tampered shares will give $m+1$, which is not same as m but is very much related to m. Tampering families that were studied so far in the context of NMSS are a)*independent tampering*: tampering of a share depends solely on itself and is independent

* Department of Computer Science and Automation, Indian Institute of Science, Email: bhavana@iisc.ac.in. Research supported by Microsoft Research Grant.
** Microsoft Research, Bangalore, Email: oslbhavana@gmail.com
*** Department of Mathematics, Indian Institute of Science, Email: sruthi.sekar1@gmail.com. Research supported in part by TCS Research Grant.
† Department of Computer Science and Automation, Indian Institute of Science, Email: jenittomy@gmail.com

of the other shares b) *joint tampering*: tampering of a share can depend on few other shares c) *affine tampering*: All shares can be tampered together, but the tampering function is restricted to be an affine function.

Locality Reconstructability/Recoverability. Inspired from the (well studied) notion of locality in the context of codes (error correcting codes [KT00,CKO14] and Non-malleable codes [DSLSZ15,CKR16,DSKS18]), we study the notion of local reconstructability in the context of secret sharing schemes. A secret sharing scheme is locally reconstructable, if it facilitates retrieval of a portion of the underlying secret such that one does not need to read through the entire share of each party in an authorised set but instead can just read a few locations from shares of parties in the authorised set.

1.1 Our Result

- We define the notion of *locally reconstructable non-malleable secret sharing schemes*(inspired from [DSLSZ15]), which are non-malleable secret sharing schemes infused with the feature of local reconstructability. Suppose the secret to be shared is parsed as a sequence of blocks $m = (m_1, \cdots, m_k)$. Assume m is shared, the shares are (possibly) tampered and let $\tilde{m} = (\tilde{m}_1, \cdots, \tilde{m}_k)$ denote the reconstruction of the tampered shares. The non-malleability guarantee is that, either $\tilde{m}$ is independent of m or there exists an efficiently samplable set description $\mathcal{I} \subset \{1, \cdots, k\}$ (independent of m) such that for $i \in \mathcal{I}$, $\tilde{m}_i = \perp$ and for $i \notin \mathcal{I}$, $\tilde{m}_i = m_i$.
- We show how to compile any non-malleable secret sharing scheme secure against some tampering model $\mathcal{F}_{nm}$ into a locally reconstructable Non-malleable secret sharing scheme.
- ***Our tampering model:*** The above compiled scheme is non-malleable against the following tampering family. Parse each share sh_i as consisting two parts a_i and b_i, i.e $sh_i = (a_i, b_i)$. b_i's (for $i \in \{1, \cdots, n\}$) can be tampered jointly and arbitrarily but independent of any a_j. All a_i's can be tampered together as per the tampering allowed by the underlying non-malleable secret sharing i.e by any $f \in \mathcal{F}_{nm}$. In addition, we can allow the description of this tampering function to depend on the values $(b_1, \cdots, b_n)$. Below we give a pictorial representation of our tampering model. We will call this family as the *Lookahead family* as tampering of a_i's can depend on b_i's but not vice-versa[1].

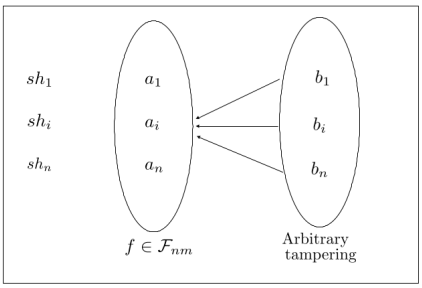

Fig. 1. Lookahead tampering. Solid arrows signify that the tampering function of a_i's (which is f) can depend on b_i's.

[1] While this may seem like an artificial model of tampering, we indeed show an application of this model to a non-malleable secure message transmission protocol.

- ***Parameters:*** Let each block of the message be ρ bits long, where ρ is some polynomial in the computational security parameter λ. Upon appropriate instantiation, we have an LRNMSS with share length of each party being $k(\rho + \log k + 2\lambda) + r(2\lambda)$, where $r(\alpha)$ is denotes the length of a share of the underlying NMSS upon sharing an α-bit secret. Asymptotically, for long messages the rate($\frac{\text{message length}}{\text{share length per party}}$) of the compiled locally reconstructable NMSS is $\frac{1}{1+o(1)}$ when ($\log k \ll \rho$). To locally reconstruct any particular block, each party of an authorised set needs to read only $\rho + \log k + r(2\lambda) + 2\lambda$ bits. Note that this quantity only depends logarithmically on k.
- ***Non-malleable secure message transmission in the pre-processing model:*** We show an application of our locally reconstructable non-malleable secret sharing scheme to a computational non-malleable message transmission protocol, in the pre-processing model (where a sender and a receiver communicate, first in a message-independent offline phase and then in a message dependent online phase). We show that a combination of our locality feature and the pre-processing, helps us improve communication, specially when the sender wants to transmit multiple messages.

1.2 Technical Overview

We parse the secret to be shared as a sequence of blocks (typically of same length) $m = (m_1, \cdots, m_k)$. Let $\mathsf{NMShare}$ denote the non-malleable secret sharing scheme to be compiled into a locally reconstructable NMSS. Let $\mathsf{Encrypt}$ be any symmetric key authenticated encryption scheme. Then our compiler proceeds as follows.

- Choose an authentication encryption key K
- Secret share K using $\mathsf{NMShare}$. Let $a_1, \cdots, a_n$ be the shares.
- Encrypt each block m_i ($i \in \{1, \cdots, k\}$) along with its location stamp, $c_i \leftarrow \mathsf{Encrypt}_K(m_i||i)$.
- For all $i \in \{1, \cdots, n\}$, set $b_i = (c_1, c_2, \cdots, c_k)$
- Output $sh_i = (a_i, b_i)$

The scheme is locally reconstructable as to recover (say) jth block, the authorised set of parties need to put together only (their respective)a_i's and c_j(which is given to them as part of b_i). Then they can check the consistency of these c_j's, reconstruct K and decrypt c_j using K to obtain m_j. If any of the above checks fail, the parties abort. This reconstruction procedure can be naturally extended to recover all the blocks. The works of [Kra93,AAG$^+$16,CFV19,FV19] use similar techniques to improve the rate, while the focus of our work is to achieve locality.

Now we provide a very brief idea of why the above scheme is non-malleable. Suppose even after tampering, if the tampered authenticated encryption key remained the same, then any tampering of the ciphertexts would be detected by the integrity of authenticated encryption. If the tampered authenticated key turns out to be independent of K, then all the information about K is lost in the shares, and the ciphertexts do not reveal anything about the messages they encrypt by the indistinguishability of encryption. We provide more details in the technical sections.

Our tampering model does not allow the tampering of ciphertexts to depend on shares of the encryption key. Allowing this kind of tampering will result in the tampered ciphertexts depending indirectly on the encryption key, which would break the encryption security. Although our scheme can be made secure against individual tampering by using secret sharing schemes with stronger security guarantees (e.g. leakage resilient schemes) as the underlying scheme, this trail would worsen the rate and deviate from our focus on building a rate-1 scheme.

While the above model for tampering our scheme seems artificial, it is indeed natural when we apply it in the context of secure message transmission. Particularly, we will send the shares

$a_1, \cdots, a_n$ of the key K in the offline phase (independent of the message to be transmitted) and then send the ciphertext c_i corresponding to message m_i in the online phase. Here, the online tampering of the ciphertext, indeed will be independent of the offline transmissions.

1.3 Organization of the paper

We provide preliminaries in Section 2. Then, we present our LRNMSS definition in Section 3.1 . We define the tampering model in Section 3.2. Our construction and security proof of the locally reconstructable non-malleable secret sharing scheme appears in Section 3.3 and Section 3.4, respectively. We also explain how to instantiate the construction in Section 3.5. In Section 4, we provide an application for our LRNMSS scheme to a non-malleable secure message transmission protocol in the pre-processing model.

2 Preliminaries

2.1 Notations

The set of all natural numbers is denoted by $\mathbb{N}$. $x \leftarrow X$ denotes sampling from a probability distribution X. All logarithms are base 2. For any two sets S and S', $S \backslash S' := \{x : x \in S, x \notin S'\}$, is the set of elements in S that are not in S'. Let $[n]$ denote the set $\{1, 2, \ldots, n\}$. Let $[n]$ represents the set of all elements. Then, the complement of the set I denoted by $\bar{I} := \{x : x \in [n], x \notin I\}$ is the set of all the elements that are not in I. For any set $T \subseteq [n]$ and a function f outputting n-tuples, $f(.)_T$ represents the output of f restricted to the set T. $negl(x)$ represents negligible function in x. For any two distributions A and B, $A \approx_c B$ means that the distributions A and B are computationally indistinguishable.

2.2 Authenticated Encryption

An encryption scheme consists of a tuple of polynomial-time algorithms $\mathcal{E} = (\mathsf{Gen}, \mathsf{Encrypt}, \mathsf{Decrypt})$ with key space $\mathcal{K}$, message space $\mathcal{M}$ and ciphertext space $\mathcal{C}$ such that:

- The randomized algorithm Gen takes as input the security parameter $\lambda \in \mathbb{N}$ and outputs a uniform key $sk \in \mathcal{K}$.
- The randomized algorithm $\mathsf{Encrypt}$ takes as input a key $sk \in \mathcal{K}$ and a message $m \in \mathcal{M}$ and outputs a ciphertext $c \in \mathcal{C}$.
- The deterministic algorithm $\mathsf{Decrypt}$ takes as input a key $sk \in \mathcal{K}$ and ciphertext $c \in \{0,1\}^*$ and outputs a value $m \in \mathcal{M} \cup \{\perp\}$, where $\perp$ denotes an invalid ciphertext.

Definition 1 ([KY00,BN00,BR00]). *An encryption scheme $\mathcal{E} = (\mathsf{Gen}, \mathsf{Encrypt}, \mathsf{Decrypt})$ is called a symmetric-key authenticated encryption scheme if it satisfies the following properties:*

1. ***Correctness.*** *For all $m \in \mathcal{M}$,*

$$\Pr[sk \leftarrow \mathsf{Gen}(1^\lambda); \mathsf{Decrypt}_{sk}(\mathsf{Encrypt}_{sk}(m)) = m] = 1$$

(where probability is taken over randomness of Gen and $\mathsf{Encrypt}$)

2. ***Semantic Security.*** *For any non-uniform PPT adversary $\mathcal{A}$, it holds that $|2 \cdot Adv_{\mathcal{E}}^{priv}(\mathcal{A}) - 1| = negl(\lambda)$, where*

$$Adv_{\mathcal{E}}^{priv} = \Pr[sk \leftarrow \mathsf{Gen}(1^\lambda); b \leftarrow \{0,1\} : \mathcal{A}^{LR_{sk,b}(\cdot,\cdot)}(1^\lambda) = b].$$

Here, the left-or-right encryption oracle $LR_{sk,b}(\cdot,\cdot)$ with $b \in \{0,1\}$ and inputs $m_0, m_1 \in \mathcal{M}$ for $|m_0| = |m_1|$, is defined as:

$$LR_{sk,b}(m_0, m_1) := \mathsf{Encrypt}_{sk}(m_b).$$

3. ***Authenticity.*** *For any non-uniform PPT adversary $\mathcal{A}$, it holds that $Adv^{auth}_{\mathcal{E}}(\mathcal{A}) = negl(\lambda)$ where*

$$Adv^{auth}_{\mathcal{E}}(\mathcal{A}) = \Pr[sk \leftarrow \mathsf{Gen}(1^\lambda), c \leftarrow \mathcal{A}^{\mathsf{Encrypt}_{sk}(\cdot)} : c \notin Q \wedge \mathsf{Decrypt}_{sk}(c) \neq \perp]$$

where Q is list of ciphertexts received by $\mathcal{A}$ through the encryption oracle.

2.3 Secret Sharing Schemes

We will be considering computational secret sharing scheme throughout this paper.

Definition 2. *Let $\mathcal{M}$ be finite set of secrets, where $|\mathcal{M}| \geq 2$. A scheme $\Sigma = (\mathsf{Share}, \mathsf{Rec})$ consists of a randomized sharing function $\mathsf{Share} : \mathcal{M} \to \mathcal{S}_1 \times \cdots \times \mathcal{S}_n$ which takes as input a secret $M \in \mathcal{M}$ and outputs n shares $(s_1, \ldots, s_n)$ where each $s_i \in \mathcal{S}_i$. The scheme Σ is called a (t, n)-threshold secret sharing scheme with message space $\mathcal{M}$ if the following properties hold:*

1. ***Correctness.*** *For any set $T \subseteq [n]$ such that $|T| \geq t$, there exists a deterministic reconstruction function $\mathsf{Rec} : \otimes_{i \in T} \mathcal{S}_i \to \mathcal{M}$ such that for every $M \in \mathcal{M}$,*

$$\Pr[\mathsf{Rec}(\mathsf{Share}(M)_T) = M] = 1$$

(over the randomness of the sharing function)

2. ***Privacy (Computational).*** *For any set $U \subseteq [n]$ such that $|U| < t$, and for every pair of secrets $M_0, M_1 \in \mathcal{M}$,*

$$\{\mathsf{Share}(M_0)_U\} \approx_c \{\mathsf{Share}(M_1)_U\}$$

2.4 Non-malleable Secret Sharing Schemes

Non-malleable secret sharing schemes were first studied in [GK18a]. We will be considering the computational variant of their definition.

Definition 3. *Let $\Sigma = (\mathsf{Share}, \mathsf{Rec})$ be a (t, n)-secret sharing scheme for message space $\mathcal{M}$. Let $\mathcal{F} \subseteq \{f : \mathcal{S}_1 \times \cdots \times \mathcal{S}_n \to \mathcal{S}_1 \times \cdots \times \mathcal{S}_n\}$ be some family of tampering functions. The scheme is said to be non-malleable w.r.t $\mathcal{F}$ if for each $f \in \mathcal{F}$ and set $T \subseteq [n]$ such that $|T| = t$, there exists a distribution $\mathsf{NMSim}^{f,T}$ such that $\forall m \in \mathcal{M}$,*

$$\mathsf{NMTamper}^{f,T}_m \approx_c NMIdeal^{\mathsf{NMSim}^{f,T}}_m$$

where $\mathsf{NMTamper}^{f,T}_m$ and $NMIdeal^{\mathsf{NMSim}^{f,T}}_m$ are distributions defined as below:

$$\mathsf{NMTamper}^{f,T}_m = \left\{ \begin{array}{l} shares \leftarrow \mathsf{Share}(m) \\ \widetilde{shares} \leftarrow f(shares) \\ \widetilde{m} \leftarrow \mathsf{Rec}(\widetilde{shares_T}) \\ Output\ \widetilde{m} \end{array} \right\}$$

$$NMIdeal^{\mathsf{NMSim}^{f,T}}_m = \left\{ \begin{array}{l} \widetilde{m} \leftarrow \mathsf{NMSim}^{f,T} \\ If\ \widetilde{m} = same^*,\ Output\ m \\ Else,\ Output\ \widetilde{m} \end{array} \right\}$$

3 Locally Reconstructable Non-malleable Secret Sharing Scheme(LRNMSSS)

In this section, we define and construct non-malleable secret sharing scheme with local reconstrcutability. Intuitively, this gives a way to secret share blocks of messages such that in order to recover a single block of message, a small number of bits from each share in a reconstruction set is needed.

3.1 LRNMSS - Definition

Definition 4. *(**LRNMSSS**) Let* $(\mathsf{Share}, \mathsf{Rec})$ *be a* (t, n)*-secret sharing scheme for message space* $\mathcal{M}$*. The scheme* $\Sigma = (\mathsf{Share}, \mathsf{Local}, \mathsf{Rec})$ *is called a* (t, n, p)*-locally reconstructable non-malleable secret sharing scheme for with message space* $\mathcal{M} = \mathcal{M}_1 \times \cdots \times \mathcal{M}_k$ *and* $\mathcal{M}_i \subseteq \{0,1\}^\rho \, \forall i \in [k]$ *if:*

1. ***Local Reconstruction.*** *For any* $M = (m_1, \ldots, m_k) \in \mathcal{M}$ *where* $m_i \in \mathcal{M}_i \, \forall i \in [k]$*, for any* $i \in [k]$ *and for any set* $T \subseteq [n]$ *such that* $|T| \geq t$*, there exists a deterministic function* Local *such that,*

$$Pr[\mathsf{Local}^{\mathsf{Share}(M)_T}(i) = m_i] = 1$$

   where Local *reads at most* p *bits from each share in* T*.*
2. ***Non-malleability.*** *Let* $\mathcal{F}$ *be some family of tampering functions. The scheme is said to be non-malleable w.r.t* $\mathcal{F}$ *if for each* $f \in \mathcal{F}$ *and set* $T \subseteq [n]$ *such that* $|T| = t$*, there exists a distribution* $\mathsf{Sim}^{f,T}$ *such that* $\forall M \in \mathcal{M}$*,*

$$\mathsf{Tamper}_M^{f,T} \approx_c Ideal_M^{\mathsf{Sim}^{f,T}}$$

   where $\mathsf{Tamper}_M^{f,T}$ *and* $Ideal_M^{\mathsf{Sim}^{f,T}}$ *are distributions defined as below:*

$$\mathsf{Tamper}_M^{f,T} = \left\{ \begin{array}{l} shares \leftarrow \mathsf{Share}(M) \\ \widetilde{shares} \leftarrow f(shares) \\ \widetilde{M} \leftarrow \mathsf{Rec}(\widetilde{shares}_T) \\ Output \ \widetilde{M} \end{array} \right\}$$

$$Ideal_M^{\mathsf{Sim}^{f,T}} = \left\{ \begin{array}{l} (\mathcal{I}^*, M^*) \leftarrow \mathsf{Sim}^{f,T} \\ If \ \mathcal{I}^* = [k], \ set \ \widetilde{M} = M^* \\ Else, \ set \ \widetilde{M}|_{\mathcal{I}^*} = \perp \ and \ \widetilde{M}|_{\overline{\mathcal{I}}^*} = M|_{\overline{\mathcal{I}}^*} \\ Output \ \widetilde{M} \end{array} \right\}$$

Now we describe the tampering model we consider in this paper.

3.2 Our Model - Lookahead Tampering

The message is partitioned into k blocks of length ρ. Let Share be a sharing function which takes as input a message $M \in \{0,1\}^{k\rho}$ and outputs n shares, namely $share_1, \ldots, share_n$ where each $share_i \in \{0,1\}^{\hat{\gamma}} \times \{0,1\}^{k\hat{\rho}}$. Each share can be viewed as $k+1$ blocks[2]. The first block is of length $\hat{\gamma}$ and next k blocks are of length $\hat{\rho}$. Let $\mathcal{F}_{nm} (\subseteq \left\{ f | f : \{0,1\}^{n\hat{\gamma}} \rightarrow \{0,1\}^{n\hat{\gamma}} \right\})$ be some set of tampering functions. We define a lookahead tampering family $\mathcal{F}$ specific to $\mathcal{F}_{nm}$. The tampering function family consists of functions of the form (f_1, f_2) where f_1 takes as input the first blocks of the shares under the constraint that $f_1 \in \mathcal{F}_{nm}$. The rest of the blocks in the shares are hardwired in function f_1. The function f_2 takes as input the last k blocks of all the shares.
Our tampering family is defined as :

$$\mathcal{F} = \left\{ f : (f_1, f_2) \middle| \begin{array}{r} \forall (x_1, \ldots, x_{nk}) \in \{0,1\}^{nk\hat{\rho}}, f_1(\cdot, x_1, \ldots, x_{nk}) \in \mathcal{F}_{nm}, \\ f_2 : \{0,1\}^{nk\hat{\rho}} \rightarrow \{0,1\}^{nk\hat{\rho}} \end{array} \right\}$$

[2] To have correspondence with the explanation in the introduction, one can consider the first block of each share to be a_i and the remaining k blocks to be b_i.

3.3 Our Construction

We use the following building blocks for the LRNMSS compiler.

1. A symmetric key authenticated encryption scheme $\mathcal{E} = (\mathsf{Gen}, \mathsf{Encrypt}, \mathsf{Decrypt})$ as in Def. 1 with key space $\mathcal{K} \subseteq \{0,1\}^{\gamma}$, message space $\mathcal{M} \subseteq \{0,1\}^{\rho+\log k}$, where k is the number of message blocks defined in Def. 4 and ciphertext space $\mathcal{C} \subseteq \{0,1\}^{\hat{\rho}}$.
2. A non-malleable secret sharing scheme $\Sigma' = (\mathsf{NMShare}^{\mathsf{t}}_{\mathsf{n}}, \mathsf{NMRec}^{\mathsf{t}}_{\mathsf{n}})$, which is non-malleable w.r.t $\mathcal{F}_{nm}$ as in Def. 3 with message space $\mathcal{M} \subseteq \{0,1\}^{\gamma}$ and share-space $\mathcal{S}_i \subseteq \{0,1\}^{\hat{\gamma}}$ for all $i \in [n]$.

Our construction combines a symmetric key authenticated encryption scheme(Def. 1) and a non-malleable secret sharing scheme (Def. 3) to obtain a locally reconstructable non-malleable secret sharing scheme(Def. 4).
Upon receiving a message having k blocks, Share function generates a key using the Gen function of the authenticated encryption and then encrypts each of the message block along with its index using the key. The key is shared using a non-malleable secret sharing scheme. A share of the key along with the ciphertexts constitutes a share of LRNMSSS.
On input an index i, Local function reads first and $(i+1)^{th}$ block of every share in a reconstruction set. Using the first blocks, Local recovers key using $\mathsf{NMRec}^{\mathsf{t}}_{\mathsf{n}}$ function of the non-malleable secret sharing. A consistency check is also made to make sure that all the $(i+1)^{th}$ blocks are the same. The $(i+1)^{th}$ block is decrypted using the recovered key. It performs a check to make sure that the index decrypted is the same as input index. Local outputs decrypted message block.
Rec function reads the shares in a reconstruction set and parses the shares as $k+1$ blocks. First block correspond to the shares of the key. Using $\mathsf{NMRec}^{\mathsf{t}}_{\mathsf{n}}$, it recovers the key . It performs consistency checks to make sure that all the ciphertext corresponding to an index are same. Rec outputs the concatenation of the decrypted messages.
The LRNMSS compiler is defined as:

$\mathsf{Share}(M)$: On input $M = (m_1, m_2, \ldots, m_k)$,

1. $sk \leftarrow \mathsf{Gen}(1^{\lambda})$.
2. $e_j \leftarrow \mathsf{Encrypt}_{sk}(m_j, j)\ \forall j \in [k]$.
3. $(sk_1, \ldots, sk_n) \leftarrow \mathsf{NMShare}^{\mathsf{t}}_{\mathsf{n}}(sk)$.
4. Output $share_i = (sk_i, e_1, \ldots, e_k)$.

$\mathsf{Local}^{share_T}(j)$: For any reconstruction set $T = \{i_1, \ldots . i_t\} \subseteq [n]$ such that $|T| \geq t$, sk_i and e^i_j represents the first and $(j+1)^{th}$ block of $share_i$ respectively. Local reads $(sk_{i_1}, e^{i_1}_j), \ldots, (sk_{i_t}, e^{i_t}_j)$ from $share_T$ on input index $j \in [k]$ and evaluates,

1. If $\exists\ i_a, i_b$ s.t. $e^{i_a}_j \neq e^{i_b}_j$, output $\perp$ and terminate.
2. Else, recover $sk \leftarrow \mathsf{NMRec}^{\mathsf{t}}_{\mathsf{n}}(sk_{i_1}, \ldots, sk_{i_t})$.
3. Recover $(\widetilde{m_j}, \tilde{j}) \leftarrow \mathsf{Decrypt}_{sk}(e^{i_1}_j)$.
   (a) If $\tilde{j} \neq j$, output $\perp$ and terminate.
4. Output $\widetilde{m_j}$.

$\mathsf{Rec}(share_T)$: For any reconstruction set $T = \{i_1, \ldots . i_t\} \subseteq [n]$ such that $|T| \geq t$ and input $share_T = share_{i_1}, \ldots, share_{i_t}$,

1. Parse $share_{i_j}$ as $(sk_{i_j}, e^{i_j}_1, \ldots, e^{i_j}_k)$.
2. $sk \leftarrow \mathsf{NMRec}^{\mathsf{t}}_{\mathsf{n}}(sk_{i_1}, \ldots, sk_{i_t})$.
3. For each $j \in [k]$,

(a) If $\exists\ i_a, i_b$ s.t. $e_j^{i_a} \neq e_j^{i_b}$, set $\widetilde{m_j} = \perp$.
(b) Else, $(\widetilde{m_j}, \tilde{j}) \leftarrow \mathsf{Decrypt}_{sk}(e_j^{i_1})$.
i. If $\tilde{j} \neq j$, set $\widetilde{m_j} = \perp$.
4. Output $\widetilde{m_1}, \ldots, \widetilde{m_k}$.

3.4 Security Analysis

Theorem 1. *Let $\mathcal{E} = (\mathsf{Gen}, \mathsf{Encrypt}, \mathsf{Decrypt})$ be a symmetric key authenticated encryption scheme with ciphertext space $\mathcal{C} \subseteq \{0,1\}^{\hat{\rho}}$, $\Sigma' = (\mathsf{NMShare}_{\mathsf{n}}^{\mathsf{t}}, \mathsf{NMRec}_{\mathsf{n}}^{\mathsf{t}})$ be a (t, n) non-malleable secret sharing scheme which is non-malleable w.r.t $\mathcal{F}_{nm}$ with share space $\mathcal{S}_i \subseteq \{0,1\}^{\hat{\gamma}}$ for all $i \in [n]$. Then, the scheme $\Sigma = (\mathsf{Share}, \mathsf{Local}, \mathsf{Rec})$ defined above is a $(t, n, \hat{\gamma} + \hat{\rho})$-locally reconstrcutable non-malleable secret sharing scheme which is non-malleable with respect to $\mathcal{F}$, the lookahead family specific to $\mathcal{F}_{nm}$.*

Proof. ***Correctness.*** The correctness of the scheme follows from the correctness of the underlying secret sharing scheme and encryption scheme.

Local Reconstruction. On input an index i, Local function reads first block and $i + 1^{th}$ block of the shares in a reconstrution set T. It recovers key from the first blocks of the shares using $\mathsf{NMRec}_{\mathsf{n}}^{\mathsf{t}}$. That would require Local to read $\hat{\gamma}$ bits from each share in T.
Local then checks if the $i + 1^{th}$ block of each share in T are same or not.This would require Local to read $\hat{\rho}$ bits from each share in T. If yes, the $i + 1^{th}$ ciphertext block is decrypted using the recovered key. It performs a check to make sure that $i + 1^{th}$ ciphertext block corresponds to the message block m_i. It checks if the decrypted index is the same as input index.
Thus, Local needs to read $p = (\hat{\gamma} + \hat{\rho})$ bits from each share in T. Total number of bits to be retrieved to reconstruct a single block m_i is $t(\hat{\gamma} + \hat{\rho})$.

Privacy. Let $T \subset [n]$ with $|T| < t$, be an arbitrary set. We wish to show that for any two messages $M^0, M^1 \in \mathcal{M}, \mathsf{Share}(M^0)_T \approx_c \mathsf{Share}(M^1)_T$.

We show this through a sequence of hybrids:

- Hybrid_0: This corresponds to the shares of $M^0 = (m_1^0, \ldots, m_k^0)$ in the set T. Generate $sk \leftarrow \mathsf{Gen}(1^\lambda)$. Further, $e_j \leftarrow \mathsf{Encrypt}_{sk}(m_j^0, j)$ for $j \in [k]$ and $(sk_1, \ldots, sk_n) \leftarrow \mathsf{NMShare}_{\mathsf{n}}^{\mathsf{t}}(sk)$. Set $share_i = (sk_i, e_1, \ldots, e_k)$ for each $i \in T$. Output $\{share_i\}_{i \in T}$.
- Hybrid_1: Generate a new key $sk' \leftarrow \mathsf{Gen}(1^\lambda)$ and replace the shares of sk in the set T with the shares of sk'.
  Generate $sk \leftarrow \mathsf{Gen}(1^\lambda)$ and $sk' \leftarrow \mathsf{Gen}(1^\lambda)$. Further, $e_j \leftarrow \mathsf{Encrypt}_{sk}(m_j^0, j)$ for $j \in [k]$ and $(sk'_1, \ldots, sk'_n) \leftarrow \mathsf{NMShare}_{\mathsf{n}}^{\mathsf{t}}(sk')$. Set $share_i = (sk'_i, e_1, \ldots, e_k)$ for each $i \in T$. Output $\{share_i\}_{i \in T}$.
- Hybrid_2: Replace the encryptions of message M^0 with encryptions of message $M^1 = (m_1^1, \ldots, m_k^1)$.
  Generate $sk \leftarrow \mathsf{Gen}(1^\lambda)$ and $sk' \leftarrow \mathsf{Gen}(1^\lambda)$. Further, $e'_j \leftarrow \mathsf{Encrypt}_{sk}(m_j^1, j)$ for $j \in [k]$ and $(sk'_1, \ldots, sk'_n) \leftarrow \mathsf{NMShare}_{\mathsf{n}}^{\mathsf{t}}(sk')$. Set $share_i = (sk'_i, e'_1, \ldots, e'_k)$ for each $i \in T$. Output $\{share_i\}_{i \in T}$.
- Hybrid_3: This corresponds to the shares of $M^1 = (m_1^1, \ldots, m_k^1)$ in the set T. Generate $sk \leftarrow \mathsf{Gen}(1^\lambda)$. Further, $e'_j \leftarrow \mathsf{Encrypt}_{sk}(m_j^1, j)$ for $j \in [k]$ and $(sk_1, \ldots, sk_n) \leftarrow \mathsf{NMShare}_{\mathsf{n}}^{\mathsf{t}}(sk)$. Set $share_i = (sk_i, e'_1, \ldots, e'_k)$ for each $i \in T$. Output $\{share_i\}_{i \in T}$.

Here, $\mathsf{Hybrid}_0 \equiv \mathsf{Share}(M^0)_T$ and $\mathsf{Hybrid}_3 \equiv \mathsf{Share}(M^1)_T$.
By the computational privacy of $\Sigma' = (\mathsf{NMShare}_{\mathsf{n}}^{\mathsf{t}}, \mathsf{NMRec}_{\mathsf{n}}^{\mathsf{t}})$, we get $\mathsf{Hybrid}_0 \approx_c \mathsf{Hybrid}_1$ and by the semantic security of $\mathcal{E} = (\mathsf{Gen}, \mathsf{Encrypt}, \mathsf{Decrypt})$, it follows $\mathsf{Hybrid}_1 \approx_c \mathsf{Hybrid}_2$. Finally, again

by the computational privacy of $\Sigma' = (\mathsf{NMShare}^{\mathsf{t}}_{\mathsf{n}}, \mathsf{NMRec}^{\mathsf{t}}_{\mathsf{n}})$, it follows $\mathsf{Hybrid}_2 \approx_c \mathsf{Hybrid}_3$. Thus, $\mathsf{Share}(M^0)_T \equiv \mathsf{Hybrid}_0 \approx_c \mathsf{Hybrid}_1 \approx_c \mathsf{Hybrid}_2 \approx_c \mathsf{Hybrid}_3 \equiv \mathsf{Share}(M^1)_T$.

Non-malleability. To show the non-malleability of our scheme, we need to show that $\forall\ f \in \mathcal{F}$, $\forall\ T \subseteq [n]$ such that $|T| = t$, $\exists\ \mathsf{Sim}^{f,T}$ such that $\forall M \in \mathcal{M}$

$$\mathsf{Tamper}^{f,T}_M \approx_c Ideal^{\mathsf{Sim}^{f,T}}_M$$

For any $f = (f_1, f_2) \in \mathcal{F}$ and any reconstruction set $T = \{i_1, \ldots, i_t\}$, we begin by describing the simulator $\mathsf{Sim}^{f,T}$.

For each $(e_1, \ldots, e_k) \in \{0,1\}^{k\hat{\rho}}$, we define a function $g : \{0,1\}^{n\hat{\gamma}} \to \{0,1\}^{n\hat{\gamma}}$, hardwired with n copies of $(e_1, \ldots, e_k)$, as $g(x) = f_1(x, (e^i_1, \ldots, e^i_k)_{i\in[n]}), \forall x \in \{0,1\}^{n\hat{\gamma}}$ where $(e^i_1, \ldots, e^i_k) = (e_1, \ldots, e_k), \forall i \in [n]$. Hence, by definition of $\mathcal{F}$, $g \in \mathcal{F}_{nm}$. Let $\mathsf{NMSim}^{g,T}$ be the simulator for the underlying NMSS (which is non-malleable w.r.t $\mathcal{F}_{nm}$) and ϕ denote the empty string.

$\mathsf{Sim}^{f,T}$:

1. $sk \leftarrow \mathsf{Gen}(1^\lambda)$.
2. $e_j \leftarrow \mathsf{Encrypt}_{sk}(0^{\rho+\log k})\ \forall j \in [k]$.
3. Set $e^i_j = e_j\ \forall j \in [k], \forall i \in [n]$ and hardwire them in g.
4. $\widetilde{sk} \leftarrow \mathsf{NMSim}^{g,T}$.
5. If $\widetilde{sk} = same^*$,
   (a) $(\tilde{e}^i_1, \ldots, \tilde{e}^i_k)_{i\in[n]} \leftarrow f_2((e^i_1, \ldots, e^i_k)_{i\in[n]})$.
   (b) $\mathcal{I} = \{j : \tilde{e}^{i_a}_j \neq \tilde{e}^{i_b}_j$ for some $i_a, i_b \in T\}$.
   (c) $\mathcal{I}_1 = \{j : \exists i_c \in T$ s.t. $\tilde{e}^{i_c}_j \neq e_j$ and $\forall i_a, i_b \in T,\ \tilde{e}^{i_a}_j = \tilde{e}^{i_b}_j\}$.
   (d) Set $(\mathcal{I}^*, M^*) = (\mathcal{I} \cup \mathcal{I}_1, \phi)$.
6. Else,
   (a) $(\tilde{e}^i_1, \ldots, \tilde{e}^i_k)_{i\in[n]} \leftarrow f_2((e^i_1, \ldots, e^i_k)_{i\in[n]})$.
   (b) $\mathcal{I} = \{j : \tilde{e}^{i_a}_j \neq \tilde{e}^{i_b}_j$ for some $i_a, i_b \in T\}$.
   (c) $\forall j \in \mathcal{I}$, set $\widetilde{m_j} = \perp$.
   (d) $\forall j \notin \mathcal{I}$, $(\widetilde{m_j}, \tilde{j}) \leftarrow \mathsf{Decrypt}_{\widetilde{sk}}(\tilde{e}^{i_1}_j)$.
      i. If $\tilde{j} \neq j$, set $\widetilde{m_j} = \perp$.
   (e) Set $(\mathcal{I}^*, M^*) = ([k], \widetilde{m_1}, \ldots, \widetilde{m_k})$.
7. Output $(\mathcal{I}^*, M^*)$.

For any $f = (f_1, f_2) \in \mathcal{F}$, any reconstruction set $T = \{i_1, \ldots, i_t\}$ and any message $M = (m_1, \ldots, m_k) \in \mathcal{M}$, we define the tamper distribution, $\mathsf{Tamper}^{f,T}_M$ as below.

$\mathsf{Tamper}^{f,T}_M$:

1. $sk \leftarrow \mathsf{Gen}(1^\lambda)$.
2. $e_j \leftarrow \mathsf{Encrypt}_{sk}(m_j, j)\ \forall j \in [k]$.
3. Set $e^i_j = e_j\ \forall j \in [k], \forall i \in [n]$.
4. $(sk_1, \ldots, sk_n) \leftarrow \mathsf{NMShare}^{\mathsf{t}}_{\mathsf{n}}(sk)$.
5. $share_i = (sk_i, e^i_1, \ldots, e^i_k)$.
6. $(\widetilde{share_1}, \ldots, \widetilde{share_n}) \leftarrow f(share_1, \ldots, share_n)$.
7. Parse $\widetilde{share_i}$ as $(\widetilde{sk_i}, \tilde{e}^i_1, \ldots, \tilde{e}^i_k)\ \forall i \in [n]$.
8. $\widetilde{sk} \leftarrow \mathsf{NMRec}^{\mathsf{t}}_{\mathsf{n}}(\widetilde{sk_{i_1}}, \ldots, \widetilde{sk_{i_t}})$.
9. For each $j \in [k]$
   (a) If $\exists\ i_a, i_b \in T$ s.t. $\tilde{e}^{i_a}_j \neq \tilde{e}^{i_b}_j$, set $\widetilde{m_j} = \perp$.

(b) Else, $(\widetilde{m_j}, \tilde{j}) \leftarrow \mathsf{Decrypt}_{\widetilde{sk}}(\tilde{e}_j^{i_1})$.
i. If $\tilde{j} \neq j$, set $\widetilde{m_j} = \perp$.
10. Output $\widetilde{m_1}, \ldots, \widetilde{m_k}$.

Now, through a sequence of hybrids, we show that the $\mathsf{Tamper}_M^{f,T}$ and $Ideal_M^{\mathsf{Sim}^{f,T}}$ are computationally indistinguishable.
We define the first hybrid, which only has some notational changes with respect to $\mathsf{Tamper}_M^{f,T}$ and is equivalent to it.
Rewriting $\mathsf{Tamper}_M^{f,T}$ as $\mathsf{Hybrid1}_M^{f,T}$: $\mathsf{Hybrid1}_M^{f,T}$ is the same as the tampering experiment with few differences. We expand the function f giving f_1 and f_2. Then, f_2 is placed after the non-malleable secret reconstruction, because $\mathsf{NMRec}_\mathsf{n}^\mathsf{t}$ doesn't depend on the output of f_2. A new variable $\mathcal{I}$ is also defined to maintain the indices having inconsistent ciphertexts.
Steps $(5) - (10)$ of $\mathsf{Tamper}_M^{f,T}$ is replaced with the following steps $(5) - (10)$ in $\mathsf{Hybrid1}_M^{f,T}$.

5. $\widetilde{sk_1}, \ldots, \widetilde{sk_n} \leftarrow f_1(sk_1, \ldots, sk_n, (e_1^i, \ldots, e_k^i)_{i\in[n]})$.
6. $\widetilde{sk} \leftarrow \mathsf{NMRec}_n^t(\widetilde{sk_{i_1}}, \ldots, \widetilde{sk_{i_t}})$.
7. $(\tilde{e}_1^i, \ldots, \tilde{e}_k^i)_{i\in[n]} \leftarrow f_2((e_1^i, \ldots, e_k^i)_{i\in[n]})$.
8. $\mathcal{I} = \{j : \tilde{e}_j^{i_a} \neq \tilde{e}_j^{i_b} \text{ for some } i_a, i_b \in T\}$.
9. (a) $\forall j \in \mathcal{I}$, set $\widetilde{m_j} = \perp$.
(b) $\forall j \notin \mathcal{I}$, $(\widetilde{m_j}, \tilde{j}) \leftarrow \mathsf{Decrypt}_{\widetilde{sk}}(\tilde{e}_j^{i_1})$.
i. If $\tilde{j} \neq j$, set $\widetilde{m_j} = \perp$.
10. Output $\widetilde{m_1}, \ldots, \widetilde{m_k}$.

Claim. For any $M \in \mathcal{M}, f \in \mathcal{F}$, and any set $T \subseteq [n]$ such that $|T| = t$,

$$\mathsf{Tamper}_M^{f,T} \equiv \mathsf{Hybrid1}_M^{f,T}$$

Proof. Clearly, only the notations were modified in $\mathsf{Hybrid1}_M^{f,T}$ and the distribution remains the same. Hence $\mathsf{Tamper}_M^{f,T}$ is identical to $\mathsf{Hybrid1}_M^{f,T}$.

In our next hybrid, we use the non-malleability of our underlying NMSS. Hence, we replace the tamper distribution of the underlying NMSS with its simulator.
Going from $\mathsf{Hybrid1}_M^{f,T}$ to $\mathsf{Hybrid2}_M^{f,T}$: $\mathsf{Hybrid2}_M^{f,T}$ is the same as $\mathsf{Hybrid1}_M^{f,T}$, except that the simulator, $\mathsf{NMSim}^{g,T}$, for the underlying NMSS, $\Sigma' = (\mathsf{NMShare}_\mathsf{n}^\mathsf{t}, \mathsf{NMRec}_\mathsf{n}^\mathsf{t})$, is used to generate the tampered key $\widetilde{sk}$.
Steps $(4) - (6)$ of $\mathsf{Hybrid1}_M^{f,T}$ is replaced with the following steps $(4) - (5)$ in $\mathsf{Hybrid2}_M^{f,T}$.

4. $\widetilde{sk} \leftarrow NMSim^{g,T}$.
5. If $\widetilde{sk} = same^*$, set $\widetilde{sk} = sk$.

Claim. If $\Sigma' = (\mathsf{NMShare}_\mathsf{n}^\mathsf{t}, \mathsf{NMRec}_\mathsf{n}^\mathsf{t})$ is a non-malleable secret sharing scheme w.r.t. $\mathcal{F}_{nm}$, then for any $M \in \mathcal{M}, f \in \mathcal{F}$, and any set $T \subseteq [n]$ such that $|T| = t$,

$$\mathsf{Hybrid1}_M^{f,T} \approx_c \mathsf{Hybrid2}_M^{f,T}$$

Proof. If the two hybrids are computationally distinguishable, we can build an adversary $\mathcal{A}$ breaking the non-malleable property of the Σ'. Let D be the distinguisher that can distinguish between $\mathsf{Hybrid1}_M^{f,T}$ and $\mathsf{Hybrid2}_M^{f,T}$.
The adversary $\mathcal{A}$ is defined as follows:

1. $\mathcal{A}$ generates $sk \leftarrow \mathsf{Gen}(1^\lambda)$.
2. $\mathcal{A}$ computes $e_j \leftarrow \mathsf{Encrypt}_{sk}(m_j, j)$ for $j \in [k]$.

3. Set $e_j^i = e_j \ \forall j \in [k], \forall i \in [n]$.
4. $\mathcal{A}$ sends $sk, g, (e_1^i, \ldots, e_k^i)_{i\in[n]}$ to the challenger.
5. $\mathcal{A}$ after receiving challenge $\widetilde{sk}$ from the challenger, does the following:
   (a) $(\tilde{e}_1^i, \ldots, \tilde{e}_k^i)_{i\in[n]} \leftarrow f_2((e_1^i, \ldots, e_k^i)_{i\in[n]})$
   (b) $\mathcal{I} = \{j : \tilde{e}_j^{i_a} \neq \tilde{e}_j^{i_b}$ for some $i_a, i_b \in T\}$
   (c) $\forall j \in \mathcal{I}$, set $\widetilde{m_j} = \perp$
   (d) $\forall j \notin \mathcal{I}$, $(\widetilde{m_j}, \tilde{j}) \leftarrow \mathsf{Decrypt}_{\widetilde{sk}}(\tilde{e}_j^{i_1})$
      i. If $\tilde{j} \neq j$, set $\widetilde{m_j} = \perp$.
   (e) Sends $D(\widetilde{m_1}, \ldots, \widetilde{m_k})$ to the challenger.

If the challenge corresponds to the output of the tampered experiment $\mathsf{NMTamper}_{sk}^{g,T}$, then D will be invoked with distribution corresponding to $\mathsf{Hybrid1}_M^{f,T}$. Otherwise, D will be invoked with distribution corresponds to the simulated experiment $NMIdeal_{sk}^{\mathsf{NMSim}^{g,T}}$. This contradicts the non-malleability property of Σ'.

In the next hybrid, we only make a few notational changes, leading to an identical distribution.
Rewriting $\mathsf{Hybrid2}_M^{f,T}$ as $\mathsf{Hybrid3}_M^{f,T}$: In $\mathsf{Hybrid3}_M^{f,T}$, $\widetilde{sk} = same^*$ and $\widetilde{sk} \neq same^*$ are considered as two different cases. When $\widetilde{sk} = same^*$, a new variable $\mathcal{I}_1$ is defined to maintain the indices having consistent, but tampered ciphertexts. For all the indices other than those in $\mathcal{I} \cup \mathcal{I}_1$, the key and the ciphertexts were not tampered. $\mathsf{Decrypt}$ outputs the original message block, m_i, at those indices.
Steps $(5) - (10)$ of $\mathsf{Hybrid2}_M^{f,T}$ is replaced with the following steps $(5) - (7)$ in $\mathsf{Hybrid3}_M^{f,T}$.

5. If $\widetilde{sk} = same^*$,
   (a) $(\tilde{e}_1^i, \ldots, \tilde{e}_k^i)_{i\in[n]} \leftarrow f_2((e_1^i, \ldots, e_k^i)_{i\in[n]})$.
   (b) $\mathcal{I} = \{j : \tilde{e}_j^{i_a} \neq \tilde{e}_j^{i_b}$ for some $i_a, i_b \in T\}$.
   (c) $\mathcal{I}_1 = \{j : \exists i_c \in T$ s.t. $\tilde{e}_j^{i_c} \neq e_j$ and $\forall i_a, i_b \in T$, $\tilde{e}_j^{i_a} = \tilde{e}_j^{i_b}\}$.
   (d) $\forall j \in \mathcal{I}$, set $\widetilde{m_j} = \perp$.
   (e) $\forall j \in \mathcal{I}_1$, $(\widetilde{m_j}, \tilde{j}) \leftarrow \mathsf{Decrypt}_{sk}(\tilde{e}_j^{i_1})$.
      i. If $\tilde{j} \neq j$, set $\widetilde{m_j} = \perp$.
   (f) $\forall j \notin \mathcal{I} \cup \mathcal{I}_1$, set $\widetilde{m_j} = m_j$.
6. Else,
   (a) $(\tilde{e}_1^i, \ldots, \tilde{e}_k^i)_{i\in[n]} \leftarrow f_2((e_1^i, \ldots, e_k^i)_{i\in[n]})$.
   (b) $\mathcal{I} = \{j : \tilde{e}_j^{i_a} \neq \tilde{e}_j^{i_b}$ for some $i_a, i_b \in T\}$.
   (c) $\forall j \in \mathcal{I}$, set $\widetilde{m_j} = \perp$.
   (d) $\forall j \notin \mathcal{I}$, $(\widetilde{m_j}, \tilde{j}) \leftarrow \mathsf{Decrypt}_{\widetilde{sk}}(\tilde{e}_j^{i_1})$.
      i. If $\tilde{j} \neq j$, set $\widetilde{m_j} = \perp$.
7. Output $\widetilde{m_1}, \ldots, \widetilde{m_k}$.

Claim. For any $M \in \mathcal{M}, f \in \mathcal{F}$, and any set $T \subseteq [n]$ such that $|T| = t$ indices,

$$\mathsf{Hybrid2}_M^{f,T} \equiv \mathsf{Hybrid3}_M^{f,T}.$$

Proof. The only difference between these hybrids are that instead of a single case in $\mathsf{Hybrid2_M}^{f,T}$, two cases were introduced in $\mathsf{Hybrid3}_M^{f,T}$ with both the cases executing the same steps. Hence, they are identical distributions.

In our next hybrid, we use the authenticity property of the underlying authenticated encryption scheme in order to completely remove the use of the original secret key sk.
Going from $\mathsf{Hybrid3}_M^{f,T}$ to $\mathsf{Hybrid4}_M^{f,T}$: In $\mathsf{Hybrid4}_M^{f,T}$, the decrypted messages corresponding to the indices in the set $\mathcal{I}_1$ are set to $\perp$.
Step $5(e)$ in $\mathsf{Hybrid3}_M^{f,T}$ is replaced with the following step in $\mathsf{Hybrid4}_M^{f,T}$.

5. (e) $\forall j \in \mathcal{I}_1$, set $\widetilde{m_j} = \perp$.

Claim. If $\mathcal{E} = (\mathsf{Gen}, \mathsf{Encrypt}, \mathsf{Decrypt})$ is an authenticated symmetric key encryption scheme, then for any $M \in \mathcal{M}, f \in \mathcal{F}$, and any set $T \subseteq [n]$ such that $|T| = t$,

$$\mathsf{Hybrid3}_M^{f,T} \approx_c \mathsf{Hybrid4}_M^{f,T}.$$

Proof. If the hybrids are computationally distinguishable, we can build an adversary which can break the authenticity property of the encryption scheme. Note that the two hybrids differ only in the case where $\tilde{sk} = same^*$ and the set $\mathcal{I}_1$ is non-empty and are identical otherwise. Pick a message $M = (m_1, \ldots, m_k) \in \mathcal{M}$ for which the two hybrids are distinguishable.
Adversary $\mathcal{A}$, which can compute a valid new ciphertext, is defined as:

1. $\mathcal{A}$ sends $(m_j, j)_{j\in[k]}$ as queries to the challenger.
2. $\mathcal{A}$ on receiving $e_1, \ldots, e_k$ from the challenger, does the following
   (a) Set $e_j^i = e_j \ \forall j \in [k], \forall i \in [n]$.
   (b) $\widetilde{sk} \leftarrow \mathsf{NMSim}^{g,T}$.
   (c) $(\tilde{e}_1^i, \ldots, \tilde{e}_k^i)_{i\in[n]} \leftarrow f_2((e_1^i, \ldots, e_k^i)_{i\in[n]})$.
   (d) $\mathcal{I}_1 = \{j : \exists i_c \in T \text{ s.t. } \tilde{e}_j^{i_c} \neq e_j \text{ and } \forall i_a, i_b \in T,\ \tilde{e}_j^{i_a} = \tilde{e}_j^{i_b}\}$.
   (e) $\mathcal{A}$ sends $(\widetilde{e}_j^{i_1})_{j\in\mathcal{I}_1}$ to the challenger.

From our assumption that the two hybrids are distinguishable, we know that there exists some $j \in \mathcal{I}_1$ such that the corresponding ciphertext $\widetilde{e}_j^{i_1}$ is valid, i.e., $\mathsf{Decrypt}_{sk}(\widetilde{e}_j^{i_1}) \neq \perp$. Since $j \in \mathcal{I}_1$, we know that $\tilde{e}_j^{i_a} = \tilde{e}_j^{i_b}$ for all $i_a, i_b \in T$ and $\tilde{e}_j^{i_1} \neq e_j$. Moreover, because of the index j being appended to the message, $\tilde{e}_j^{i_1} \neq e_q$, for each $q \in [k]$. This implies that the adversary $\mathcal{A}$ outputs a valid ciphertext, which it did not receive as a challenge, hence breaking the authenticity of the encryption.

Next, we use the semantic security of the authentication scheme to move to a hybrid where, the actual message is no longer used in the encryption.
Going from $\mathsf{Hybrid4}_M^{f,T}$ to $\mathsf{Hybrid5}_M^{f,T}$: In $\mathsf{Hybrid5}_M^{f,T}$, the ciphertexts corresponding to M are replaced with the ciphertexts corresponding to $0^{\rho+\log k}$.
Step 2 in $\mathsf{Hybrid4}_M^{f,T}$ is replaced with the following step in $\mathsf{Hybrid5}_M^{f,T}$.

2. $e_j \leftarrow \mathsf{Encrypt}_{sk}(0^{\rho+\log k})\ \forall j \in [k]$

Claim. If $\mathcal{E} = (\mathsf{Gen}, \mathsf{Encrypt}, \mathsf{Decrypt})$ is an authenticated encryption scheme, then for any $M \in \mathcal{M}, f \in \mathcal{F}$, and any set $T \subseteq [n]$ such that $|T| = t$,

$$\mathsf{Hybrid4}_M^{f,T} \approx_c \mathsf{Hybrid5}_M^{f,T}.$$

Proof. Assume to the contrary that, there exists $M \in \mathcal{M}$ and a distinguisher D that can distinguish between the hybrids $\mathsf{Hybrid4}_M^{f,T}$ and $\mathsf{Hybrid5}_M^{f,T}$. The distinguisher D can be used to construct another distinguisher D_1 which violates the semantic security of the underlying encryption scheme $\mathcal{E}$.
The distinguisher D_1 is defined as follows:

1. D_1 sets $M_0 = (m_j, j)_{j\in[k]}$, $M_1 = 0^{k(\rho+\log k)}$.
2. D_1 sends (M_0, M_1) to the challenger.
3. D_1 on receiving $(e_1, \ldots, e_k)$ from the challenger, does the following,
   (a) Set $e_j^i = e_j \ \forall j \in [k], \forall i \in [n]$.
   (b) $\widetilde{sk} \leftarrow \mathsf{NMSim}^{g,T}$.

(c) If $\widetilde{sk} = same^*$,
   i. $(\tilde{e}_1^i, \ldots, \tilde{e}_k^i)_{i\in[n]} \leftarrow f_2((e_1^i, \ldots, e_k^i)_{i\in[n]})$.
   ii. $\mathcal{I} = \{j : \tilde{e}_j^{i_a} \neq \tilde{e}_j^{i_b} \text{ for some } i_a, i_b \in T\}$.
   iii. $\mathcal{I}_1 = \{j : \exists i_c \in T \text{ s.t. } \tilde{e}_j^{i_c} \neq e_j \text{ and } \forall i_a, i_b \in T,\ \tilde{e}_j^{i_a} = \tilde{e}_j^{i_b}\}$.
   iv. $\forall j \in \mathcal{I}$, set $\widetilde{m_j} = \perp$.
   v. $\forall j \in \mathcal{I}_1$, set $\widetilde{m_j} = \perp$.
   vi. $\forall j \notin \mathcal{I} \cup \mathcal{I}_1$, set $\widetilde{m_j} = m_j$.

(d) Else,
   i. $(\tilde{e}_1^i, \ldots, \tilde{e}_k^i)_{i\in[n]} \leftarrow f_2((e_1^i, \ldots, e_k^i)_{i\in[n]})$.
   ii. $\mathcal{I} = \{j : \tilde{e}_j^{i_a} \neq \tilde{e}_j^{i_b} \text{ for some } i_a, i_b \in T\}$.
   iii. $\forall j \in \mathcal{I}$, set $\widetilde{m_j} = \perp$.
   iv. $\forall j \notin \mathcal{I}$, $(\widetilde{m_j}, \tilde{j}) \leftarrow \mathsf{Decrypt}_{\widetilde{sk}}(\tilde{e}_j^{i_1})$.
      A. If $\tilde{j} \neq j$, set $\widetilde{m_j} = \perp$.

(e) D_1 sends $D(\widetilde{m_1}, \ldots, \widetilde{m_k})$ to the challenger.

If the challenge corresponds to the ciphertext of M_0, then D will be invoked with the distribution corresponding to $\mathsf{Hybrid4}_M^{f,T}$. Otherwise, the distribution corresponds to $\mathsf{Hybrid5}_M^{f,T}$. This contradicts the semantic security of $\mathcal{E}$.

We finally make some notational changes to the above hybrid, to get an identical distribution, which would be $Ideal_M^{\mathsf{Sim}^{f,T}}$.

Rewriting $\mathsf{Hybrid5}_M^{f,T}$ as $\mathsf{Hybrid6}_M^{f,T}$: The new variable $\mathcal{I}^*$ represents the set of tampered indices. If shares are entirely tampered, the output will be independent of the original message. If the tampering function doesn't change first blocks of shares, then any modification will output $\perp$. $\mathcal{I}^*$ keeps track of these indices. The simulator for the LRNMSSS outputs tampered indices along with message vector. If the shares are entirely tampered, $\mathsf{Hybrid6}_M^{f,T}$ will output the message vector which is independent of original message. If the first block is not tampered, the $\mathsf{Hybrid6}_M^{f,T}$ outputs $\perp$ at indices in $\mathcal{I}^*$ and original messages for other indices.
Steps $(5) - (7)$ of $\mathsf{Hybrid5}_M^{f,T}$ are replaced with the following steps in $\mathsf{Hybrid6}_M^{f,T}$

5. If $\widetilde{sk} = same^*$,
   (a) $(\tilde{e}_1^i, \ldots, \tilde{e}_k^i)_{i\in[n]} \leftarrow f_2((e_1^i, \ldots, e_k^i)_{i\in[n]})$.
   (b) $\mathcal{I} = \{j : \tilde{e}_j^{i_a} \neq \tilde{e}_j^{i_b} \text{ for some } i_a, i_b \in T\}$.
   (c) $\mathcal{I}_1 = \{j : \exists i_c \in T \text{ s.t. } \tilde{e}_j^{i_c} \neq e_j \text{ and } \forall i_a, i_b \in T,\ \tilde{e}_j^{i_a} = \tilde{e}_j^{i_b}\}$.
   (d) Set $(\mathcal{I}^*, M^*) = (\mathcal{I} \cup \mathcal{I}_1, \phi)$.
6. Else,
   (a) $(\tilde{e}_1^i, \ldots, \tilde{e}_k^i)_{i\in[n]} \leftarrow f_2((e_1^i, \ldots, e_k^i)_{i\in[n]})$.
   (b) $\mathcal{I} = \{j : \tilde{e}_j^{i_a} \neq \tilde{e}_j^{i_b} \text{ for some } i_a, i_b \in T\}$.
   (c) $\forall j \in \mathcal{I}$, set $\widetilde{m_j} = \perp$.
   (d) $\forall j \notin \mathcal{I}$, $(\widetilde{m_j}, \tilde{j}) \leftarrow \mathsf{Decrypt}_{\widetilde{sk}}(\tilde{e}_j^{i_1})$.
      i. If $\tilde{j} \neq j$, set $\widetilde{m_j} = \perp$.
   (e) Set $(\mathcal{I}^*, M^*) = ([k], \widetilde{m_1}, \ldots, \widetilde{m_k})$.
7. If $\mathcal{I}^* = [k]$, set $\widetilde{M} = M^*$.
8. Else, set $\widetilde{M}|_{\mathcal{I}^*} = \perp$ and $\widetilde{M}|_{\overline{\mathcal{I}}^*} = M|_{\overline{\mathcal{I}}^*}$.
9. Output $\widetilde{M}$.

Claim. For any $M \in \mathcal{M}, f \in \mathcal{F}$, and any set $T \subseteq [n]$ such that $|T| = t$ indices,

$$\mathsf{Hybrid5}_M^{f,T} \equiv \mathsf{Hybrid6}_M^{f,T}.$$

Proof. When $\mathcal{I}^* = [k]$, key can be tampered or not. If the key was tampered, M^* stores the messages decrypted using the tampered key at all the indices. $\mathsf{Hybrid5}_M^{f,T}$ also outputs the decrypted messages at all indices when the key is tampered. If the key was not tampered, M^* stores empty vector. $\mathsf{Hybrid6}_M^{f,T}$ outputs M^* in that case. If $\mathcal{I}^* = [k]$, then $\mathcal{I} \cup \mathcal{I}_1 = [k]$. $\mathsf{Hybrid5}_M^{f,T}$ outputs $\perp$ at all the indices.
When $\mathcal{I}^* \neq [k]$, key was not tampered. Both the hybrids output $\perp$ at those indices in $\mathcal{I}^* = \mathcal{I} \cup \mathcal{I}_1$. On all the other indices, $\mathsf{Hybrid5}_M^{f,T}$ and $\mathsf{Hybrid6}_M^{f,T}$ outputs original message corresponding to their indices. Both the hybrids behave the same for all the cases. Thus, $\mathsf{Hybrid5}_M^{f,T}$ and $\mathsf{Hybrid6}_M^{f,T}$ are identical.

From our description of the simulator, $\mathsf{Sim}^{f,T}$, clearly $\mathsf{Hybrid6}_M^{f,T}$ is the same as $Ideal_M^{\mathsf{Sim}^{f,T}}$.
By the previous claims, we get
$\mathsf{Tamper}_M^{f,T} \equiv \mathsf{Hybrid1}_M^{f,T} \approx_c \mathsf{Hybrid2}_M^{f,T} \equiv \mathsf{Hybrid3}_M^{f,T} \approx_c \mathsf{Hybrid4}_M^{f,T}$
$\mathsf{Hybrid4}_M^{f,T} \approx_c \mathsf{Hybrid5}_M^{f,T} \equiv \mathsf{Hybrid6}_M^{f,T} \equiv Ideal_M^{\mathsf{Sim}^{f,T}}$

3.5 Instantiation

Let the secret to be shared in LRNMSS consists of k blocks, with size of each block ρ, where ρ is some polynomial in the computational security parameter λ. Let $(\mathsf{NMShare}_{\mathsf{n}}^{\mathsf{t}}, \mathsf{NMRec}_{\mathsf{n}}^{\mathsf{t}})$ be a NMSS with length of the each share being $r(\alpha)$ when a α-bit secret is shared. Authenticated encryption $(\mathsf{Gen}, \mathsf{Encrypt}, \mathsf{Dec})$ scheme is instantiated with *Encrypt-and-authenticate* scheme mentioned in Section 4.5 of [KL14] . We let the encryption key, randomness and tag to be of length $2\lambda, \lambda$ and λ respectively. The encryption scheme takes messages of length $\rho + \log k$, outputs a ciphertext of length $\rho + \log k + 2\lambda$.

- For messages of length $k\rho$, a single share of LRNMSS will be of length $k(\rho + \log k + 2\lambda) + r(2\lambda)$.
- Thus, the rate of LRNMSS is $\frac{k\rho}{k(\rho+\log k+2\lambda)+r(2\lambda)}$.
- For long messages, rate $= \frac{1}{1+o(1)}$ assuming $\log k \ll \rho$.
- For local reconstruction, Local is required to read $\rho + \log k + 2\lambda + r(2\lambda)$ bits from each share in a reconstruction set.

4 Computational Non-malleable Multi-message Transmission in the Pre-processing Model

Perfectly secure message transmission (SMT) was introduced in [DDWY93], where a sender S wants to transmit a message m to a receiver R, through n wires between them, ensuring that perfect secrecy is guaranteed, even in the presence of an eavesdropping adversary looking at a bounded number of wires, and perfect resiliency is guaranteed, even in the presence of an adversary controlling a bounded number of wires completely. Post their introduction, SMTs have been studied in several works [DDWY93,SNR04,WD08,KS09,KKVS18]. Non-malleable secure message transmission (NMSMT) was introduced in [GK18a], where the goal is to guarantee non-malleability in the presence of an adversary who can tamper all n wires according to some tampering model (i.e., the tampered message m' is guaranteed to be either same as the original message m, or is completely independent of it). Further, they build NMSMTs using non-malleable secret sharing schemes.

In this work, we show an application of our LRNMSS scheme to build a computational SMT protocol in the pre-processing model, that allows the sender and receiver to communicate in two phases: a message-independent *offline phase* and a message-dependent *online phase*, to non-malleably send multiple messages to the receiver, while saving on the online communication.

Formally, we allow S and R to first communicate in an offline phase, where S sends messages $x_1, \cdots, x_n$ to R (which are all independent of the messages to be transmitted in the online phase). In the online phase, S can securely send message m by sending a single message c, through one wire, to R. In both the online and offline phase, the adversary can tamper the messages being sent (with the restriction that each wire for the offline phase communication can be arbitrarily tampered independent of each other, and the single wire for the online phase communication, can be arbitrarily tampered independent of the offline communication). The guarantee is that the tampered messages are either the same or are independent of the original messages. To transmit k messages, each of size $\rho = poly(\lambda)$ (for security parameter λ), our protocol requires an offline communication of 2λ bits per wire (with n wires in total) and an online communication of $\rho + \log k + 2\lambda$ bits per message. In comparison, even if we instantiate the NMSMT protocol of [GK18a] with a rate-1 computational non-malleable secret sharing scheme [BFO+20,FV19] (as in our construction), to send k messages of length ρ each, the protocol would need to commmunicate $n\rho$ bits (in total) per message in a single online phase. Hence, by introducing an offline phase and by leveraging the locality of our LRNMSS construction, we save a factor of n in the online communication required.

We now define our model for computational non-malleable multi-message transmission formally.

Definition 5 (Computational Non-malleable Multi-message Transmission). *Let S and R denote the sender and receiver of the message transmission protocol, respectively and let $\mathcal{M}$ denote the message space from which S wants to transmit messages to R. S and R communicate in two phases: in the offline phase, they communicate through n wires connecting them, and in the online phase, they communicate through a single wire. In the offline phase, S sends messages $x_1, \cdots, x_n$ to R (each x_i is sent through the i-th wire). In the online phase, to transmit a message m to R, S sends the message c. Let $\pi(m_1, \cdots, m_k, S, R)$ denote an execution of the protocol to transmit k messages $m_1, \cdots, m_k$ (involving a single offline phase message and k online phase messages). We say that $\pi(\cdot, S, R)$ is a k-non-malleable multi-message transmission protocol with respect to a tampering family $\mathcal{F}_{split}$, if it satisfies the following properties:*

1. ***Correctness:*** *For all messages $m_1, \cdots, m_k \in \mathcal{M}$, at the end of an honest execution of the protocol $\pi(m_1, \cdots, m_k, S, R)$, the receiver receives the messages $m_1, \cdots, m_k$, with probability 1.*
2. ***Computational Privacy:*** *For every adversary $\mathcal{A}$ that can see at most $n-1$ wires in the offline phase and the single wire of the online phase, and for each pair of multi-messages $(m_1, \cdots, m_k), (m'_1, \cdots, m'_k)$,*

$$\pi^{view}_{\mathcal{A}}(m_1, \cdots, m_k, S, R) \approx_c \pi^{view}_{\mathcal{A}}(m'_1, \cdots, m'_k, S, R),$$

   where $\pi^{view}_{\mathcal{A}}(m_1, \cdots, m_k, S, R)$ denotes the distribution corresponding to the view of $\mathcal{A}$ in the protocol execution $\pi(m_1, \cdots, m_k, S, R)$, which includes the messages sent through $n-1$ wires in the offline phase and the messages sent in the online phase.
3. ***Non-malleability:***
   Tampering Family $\mathcal{F}_{split}$: *We allow each wire of the offline phase to be tampered independent of each other, and the online phase messages are tampered independent of all offline messages (but may depend on each other). Hence, each $f \in \mathcal{F}_{split}$ consists of functions $f_1, \cdots, f_n, g$, where each f_i acts on wire i of the offline phase and g acts on the online phase messages.*
   For each $f \in \mathcal{F}_{split}$, there exists a distribution Sim^f over $\mathcal{M}$, such that, for all sets of messages $(m_1, \cdots, m_k)$,

$$\mathsf{Tamper}^f_{m_1, \cdots, m_k} \approx_c Copy(m_1, \cdots, m_k, \mathsf{Sim}^f),$$

where $\mathsf{Tamper}^f_{m_1,\cdots,m_k}$ *and* $Copy(m_1,\cdots,m_k,\mathsf{Sim}^f)$ *are defined as follows:*

$$\mathsf{Tamper}^f_{m_1,\cdots,m_k} = \left\{ \begin{array}{l} (x_1,\cdots,x_n,c_1,\cdots,c_k) \leftarrow \pi(m_1,\cdots,m_k,S,R) \\ (x'_1,\cdots,x'_n,c'_1,\cdots,c'_k) = f((x_1,\cdots,x_n,c_1,\cdots,c_k)) \\ (m'_1,\cdots,m'_k) \leftarrow R(x'_1,\cdots,x'_n,c'_1,\cdots,c'_k) \end{array} \right\}$$

$$Copy(m_1,\cdots,m_k,\mathsf{Sim}^f) = \left\{ \begin{array}{l} (I,m^*_1,\cdots,m^*_k) \leftarrow \mathsf{Sim}^f \\ If\ I=[k],\ set\ m' = m^*_1,\cdots,m^*_k \\ Else,\ set\ m'|_I = \perp,\ and\ m'|_{\overline{I}} = (m_i)_{i\in\overline{I}} \\ Output: m' \end{array} \right\}$$

Construction: Consider our LRNMSS construction from 3.3, specifically for n-threshold setting. S generates $sk \leftarrow \mathsf{Gen}(1^\lambda)$ and sends the shares $(sk_1,\cdots,sk_n) \leftarrow \mathsf{NMShare}^n_n(sk)$ through the n-wires in the offline phase (sk_i is sent through wire i, for each $i \in [n]$). For each message m_j ($j \in [k]$) in the online phase, S sends the ciphertext $c_j = \mathsf{Encrypt}_{sk}(m_j, j)$ to R. Now, clearly, R can reconstruct to recover sk from the offline communication and decrypt each ciphertext from the online phase.[3]

Theorem 2. *Let the messages being transmitted be of ρ bits each. If $(\mathsf{NMShare}^n_n, \mathsf{NMRec}^n_n)$-is a n-threshold computational non-malleable secret sharing scheme against independent tampering (each share tampered independently and arbitrarily) with rate 1 and $(\mathsf{Gen}, \mathsf{Encrypt}, \mathsf{Decrypt})$ is a symmetric key authenticated encryption scheme, then the above construction describes a k-non-malleable multi-message transmission protocol with respect to the tampering family $\mathcal{F}_{split}$ with an offline communication complexity of $2n\lambda$ bits (through all wires combined) and an online communication complexity of $(\rho + \log k + 2\lambda)$ bits, per message sent.*

Proof. The correctness and computational privacy of the protocol directly follow from the correctness and privacy of our LRNMSS scheme.
For non-malleability, note that the tampering model $\mathcal{F}_{split}$, is in fact weaker than the tampering model $\mathcal{F}$ of our LRNMSS. Note that, tampering of the shares sent in the offline face indeed belong to $\mathcal{F}_{nm}$ (here, the tampering doesn't depend on the ciphertexts sent in the online phase) and the ciphertexts are all tampered independent of the offline shares. Hence, by the non-malleability of our LRNMSS scheme, the non-mallebility of our SMT protocol follows.

Communication Cost. Let each message being transmitted be of ρ bits, k be the number of messages transmitted, λ be the security parameter, n be the number of wires in the offline phase. If we instantiate our protocol with the rate 1 computational NMSS scheme of [FV19,BFO$^+$20], we get a total offline communication complexity of $2n\lambda$ bits and an online communication complexity of $(\rho + \log k + 2\lambda)$ bits, per message.

References

AAG$^+$16. Divesh Aggarwal, Shashank Agrawal, Divya Gupta, Hemanta K. Maji, Omkant Pandey, and Manoj 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.

[3] Note that, we can use a two split state non-malleable code (which is implicitly a 2-out-of-2 NMSS) in the offline phase, if non-malleability is the only concern. However, with the use of an n-out-of-n NMSS, we get stronger security guarantees, where the adversary gets more eavesdropping power ($n-1$ wires).

ADN+19. Divesh Aggarwal, Ivan Damgård, Jesper Buus Nielsen, Maciej Obremski, Erick Purwanto, João L. Ribeiro, and Mark Simkin. Stronger leakage-resilient and non-malleable secret sharing schemes for general access structures. In Alexandra Boldyreva and Daniele Micciancio, editors, *Advances in Cryptology - CRYPTO 2019 - 39th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 18-22, 2019, Proceedings, Part II*, volume 11693 of *Lecture Notes in Computer Science*, pages 510–539. Springer, 2019.

BFO+20. Gianluca Brian, Antonio Faonio, Maciej Obremski, Mark Simkin, and Daniele Venturi. Non-malleable secret sharing against bounded joint-tampering attacks in the plain model. In Daniele Micciancio and Thomas Ristenpart, editors, *Advances in Cryptology - CRYPTO 2020 - 40th Annual International Cryptology Conference, CRYPTO 2020, Santa Barbara, CA, USA, August 17-21, 2020, Proceedings, Part III*, volume 12172 of *Lecture Notes in Computer Science*, pages 127–155. Springer, 2020.

BFV19. Gianluca Brian, Antonio Faonio, and Daniele Venturi. Continuously non-malleable secret sharing for general access structures. In Dennis Hofheinz and Alon Rosen, editors, *Theory of Cryptography - 17th International Conference, TCC 2019, Nuremberg, Germany, December 1-5, 2019, Proceedings, Part II*, volume 11892 of *Lecture Notes in Computer Science*, pages 211–232. Springer, 2019.

Bla79. G.R. Blakley. Safeguarding cryptographic keys. In *Proceedings of the 1979 AFIPS National Computer Conference*, pages 313–317, Monval, NJ, USA, 1979. AFIPS Press.

BN00. Mihir Bellare and Chanathip Namprempre. Authenticated encryption: Relations among notions and analysis of the generic composition paradigm. In *International Conference on the Theory and Application of Cryptology and Information Security*, pages 531–545. Springer, 2000.

BR00. Mihir Bellare and Phillip Rogaway. Encode-then-encipher encryption: How to exploit nonces or redundancy in plaintexts for efficient cryptography. In *International Conference on the Theory and Application of Cryptology and Information Security*, pages 317–330. Springer, 2000.

BS19. Saikrishna Badrinarayanan and Akshayaram Srinivasan. Revisiting non-malleable secret sharing. In Yuval Ishai and Vincent Rijmen, editors, *Advances in Cryptology - EUROCRYPT 2019 - 38th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Darmstadt, Germany, May 19-23, 2019, Proceedings, Part I*, volume 11476 of *Lecture Notes in Computer Science*, pages 593–622. Springer, 2019.

CFV19. Sandro Coretti, Antonio Faonio, and Daniele Venturi. Rate-optimizing compilers for continuously non-malleable codes. In Robert H. Deng, Valérie Gauthier-Umaña, Martín Ochoa, and Moti Yung, editors, *Applied Cryptography and Network Security - 17th International Conference, ACNS 2019, Bogota, Colombia, June 5-7, 2019, Proceedings*, volume 11464 of *Lecture Notes in Computer Science*, pages 3–23. Springer, 2019.

CGGL20. Eshan Chattopadhyay, Jesse Goodman, Vipul Goyal, and Xin Li. Leakage-resilient extractors and secret-sharing against bounded collusion protocols. *IACR Cryptol. ePrint Arch.*, 2020:478, 2020.

CKO14. Nishanth Chandran, Bhavana Kanukurthi, and Rafail Ostrovsky. Locally updatable and locally decodable codes. In Yehuda Lindell, editor, *Theory of Cryptography*, pages 489–514, Berlin, Heidelberg, 2014. Springer Berlin Heidelberg.

CKOS20. Nishanth Chandran, Bhavana Kanukurthi, Sai Lakshmi Bhavana Obbattu, and Sruthi Sekar. Constant rate (non-malleable) secret sharing schemes tolerating joint adaptive leakage. Cryptology ePrint Archive, Report 2020/1252, 2020. `https://eprint.iacr.org/2020/1252`.

CKR16. Nishanth Chandran, Bhavana Kanukurthi, and Srinivasan Raghuraman. Information-theoretic local non-malleable codes and their applications. In Eyal Kushilevitz and Tal Malkin, editors, *Theory of Cryptography - 13th International Conference, TCC 2016-A, Tel Aviv, Israel, January 10-13, 2016, Proceedings, Part II*, volume 9563 of *Lecture Notes in Computer Science*, pages 367–392. Springer, 2016.

DDWY93. Danny Dolev, Cynthia Dwork, Orli Waarts, and Moti Yung. Perfectly secure message transmission. *J. ACM*, 40(1):17–47, 1993.

DSKS18. Dana Dachman-Soled, Mukul Kulkarni, and Aria Shahverdi. Local non-malleable codes in the bounded retrieval model. In Michel Abdalla and Ricardo Dahab, editors, *Public-Key Cryptography – PKC 2018*, pages 281–311, Cham, 2018. Springer International Publishing.

DSLSZ15. Dana Dachman-Soled, Feng-Hao Liu, Elaine Shi, and Hong-Sheng Zhou. Locally decodable and updatable non-malleable codes and their applications. In Yevgeniy Dodis and Jesper Buus Nielsen, editors, *Theory of Cryptography*, pages 427–450, Berlin, Heidelberg, 2015. Springer Berlin Heidelberg.

FV19. Antonio Faonio and Daniele Venturi. Non-malleable secret sharing in the computational setting: Adaptive tampering, noisy-leakage resilience, and improved rate. In Alexandra Boldyreva and Daniele Micciancio, editors, *Advances in Cryptology - CRYPTO 2019 - 39th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 18-22, 2019, Proceedings, Part II*, volume 11693 of *Lecture Notes in Computer Science*, pages 448–479. Springer, 2019.

GK18a. Vipul Goyal and Ashutosh Kumar. Non-malleable secret sharing. In Ilias Diakonikolas, David Kempe, and Monika Henzinger, editors, *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2018, Los Angeles, CA, USA, June 25-29, 2018*, pages 685–698. ACM, 2018.

GK18b. Vipul Goyal and Ashutosh Kumar. Non-malleable secret sharing for general access structures. In Hovav Shacham and Alexandra Boldyreva, editors, *Advances in Cryptology – CRYPTO 2018*, pages 501–530, Cham, 2018. Springer International Publishing.

KKVS18. Ravi Kishore, Ashutosh Kumar, Chiranjeevi Vanarasa, and Kannan Srinathan. On the price of proactivizing round-optimal perfectly secret message transmission. *IEEE Trans. Inf. Theory*, 64(2):1404–1422, 2018.

KL14. Jonathan Katz and Yehuda Lindell. *Introduction to modern cryptography*. CRC press, 2014.

KMS19. Ashutosh Kumar, Raghu Meka, and Amit Sahai. Leakage-resilient secret sharing against colluding parties. In David Zuckerman, editor, *60th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2019, Baltimore, Maryland, USA, November 9-12, 2019*, pages 636–660. IEEE Computer Society, 2019.

KMZ20. Ashutosh Kumar, Raghu Meka, and David Zuckerman. Bounded collusion protocols, cylinder-intersection extractors and leakage-resilient secret sharing. *Electron. Colloquium Comput. Complex.*, 27:55, 2020.

Kra93. Hugo Krawczyk. Secret sharing made short. In Douglas R. Stinson, editor, *Advances in Cryptology - CRYPTO '93, 13th Annual International Cryptology Conference, Santa Barbara, California, USA, August 22-26, 1993, Proceedings*, volume 773 of *Lecture Notes in Computer Science*, pages 136–146. Springer, 1993.

KS09. Kaoru Kurosawa and Kazuhiro Suzuki. Truly efficient 2-round perfectly secure message transmission scheme. *IEEE Trans. Inf. Theory*, 55(11):5223–5232, 2009.

KT00. Jonathan Katz and Luca Trevisan. On the efficiency of local decoding procedures for error-correcting codes. In *Proceedings of the Thirty-Second Annual ACM Symposium on Theory of Computing*, STOC '00, page 80–86, New York, NY, USA, 2000. Association for Computing Machinery.

KY00. Jonathan Katz and Moti Yung. Unforgeable encryption and chosen ciphertext secure modes of operation. In *International Workshop on Fast Software Encryption*, pages 284–299. Springer, 2000.

LCG+19. Fuchun Lin, Mahdi Cheraghchi, Venkatesan Guruswami, Reihaneh Safavi-Naini, and Huaxiong Wang. Non-malleable secret sharing against affine tampering. *CoRR*, abs/1902.06195, 2019.

Sha79. Adi Shamir. How to share a secret. *Communications of the ACM*, 22(11):612–613, 1979.

SNR04. K. Srinathan, Arvind Narayanan, and C. Pandu Rangan. Optimal perfectly secure message transmission. In Matthew K. Franklin, editor, *Advances in Cryptology - CRYPTO 2004, 24th Annual International CryptologyConference, Santa Barbara, California, USA, August 15-19, 2004, Proceedings*, volume 3152 of *Lecture Notes in Computer Science*, pages 545–561. Springer, 2004.

SV19. Akshayaram Srinivasan and Prashant Nalini Vasudevan. Leakage resilient secret sharing and applications. In Alexandra Boldyreva and Daniele Micciancio, editors, *Advances in Cryptology – CRYPTO 2019*, pages 480–509, Cham, 2019. Springer International Publishing.

WD08. Yongge Wang and Yvo Desmedt. Perfectly secure message transmission revisited. *IEEE Trans. Inf. Theory*, 54(6):2582–2595, 2008.