Revisiting Non-Malleable Secret Sharing

Saikrishna Badrinarayanan[1(✉)] and Akshayaram Srinivasan[2]

[1] UCLA, Los Angeles, USA
saikrishna@cs.ucla.edu
[2] UC Berkeley, Berkeley, USA
akshayaram@berkeley.edu

Abstract. A threshold secret sharing scheme (with threshold t) allows a dealer to share a secret among a set of parties such that any group of t or more parties can recover the secret and no group of at most $t-1$ parties learn any information about the secret. A non-malleable threshold secret sharing scheme, introduced in the recent work of Goyal and Kumar (STOC'18), additionally protects a threshold secret sharing scheme when its shares are subject to tampering attacks. Specifically, it guarantees that the reconstructed secret from the tampered shares is either the original secret or something that is unrelated to the original secret.

In this work, we continue the study of threshold non-malleable secret sharing against the class of tampering functions that tamper each share independently. We focus on achieving greater *efficiency* and guaranteeing a *stronger* security property. We obtain the following results:

- **Rate Improvement.** We give the first construction of a threshold non-malleable secret sharing scheme that has rate > 0. Specifically, for every $n, t \geq 4$, we give a construction of a t-out-of-n non-malleable secret sharing scheme with rate $\Theta(\frac{1}{t \log^2 n})$. In the prior constructions, the rate was $\Theta(\frac{1}{n \log m})$ where m is the length of the secret and thus, the rate tends to 0 as $m \to \infty$. Furthermore, we also optimize the parameters of our construction and give a concretely efficient scheme.
- **Multiple Tampering.** We give the first construction of a threshold non-malleable secret sharing scheme secure in the stronger setting of bounded tampering wherein the shares are tampered by multiple (but bounded in number) possibly different tampering functions. The rate of such a scheme is $\Theta(\frac{1}{k^3 t \log^2 n})$ where k is an apriori bound on the number of tamperings. We complement this positive result by proving that it is impossible to have a threshold non-malleable secret sharing scheme that is secure in the presence of an apriori unbounded number of tamperings.
- **General Access Structures.** We extend our results beyond threshold secret sharing and give constructions of rate-efficient, non-malleable secret sharing schemes for more general monotone access structures that are secure against multiple (bounded) tampering attacks.

Y. Ishai and V. Rijmen (Eds.): EUROCRYPT 2019, LNCS 11476, pp. 593–622, 2019.
https://doi.org/10.1007/978-3-030-17653-2_20

1 Introduction

A t-out-of-n threshold secret sharing scheme [Sha79,Bla79] allows a dealer to share a secret among n parties such that any subset of t or more parties can recover the secret but any subset of $t-1$ parties learn no information about the secret. Threshold secret sharing schemes are central tools in cryptography and have several applications such as constructing secure multiparty computation protocols [GMW87,BGW88,CCD88], threshold cryptographic systems [DF90, Fra90,DDFY94] and leakage resilient circuit compilers [ISW03,FRR+10,Rot12] to name a few.

Most of the threshold secret sharing schemes in literature are *linear*. This means that if we multiply each share by a constant c, we get a set of shares that correspond to a new secret that is c times the original secret. This property has in fact, been crucially leveraged in most of the applications including designing secure multiparty computation protocols and constructing threshold cryptosystems. However, this highly desirable feature becomes undesirable if our primary goal is to protect the shares against tampering attacks. More specifically, this linearity property allows an adversary to tamper (or maul) each share independently and output a new set of shares that reconstruct to a related secret (for example, two times the original secret). Indeed, if the shares of the secret are stored on a device such as a smart card, an adversary could potentially tamper with the smart card and change the value of the share that is being stored by overwriting it with a new value or maybe flipping a few bits. Notice that in the above tampering attack, the adversary need not learn the actual secret. However, the adversary is guaranteed to produce a set of shares that reconstruct to a related secret. Such an attack could be devastating when the shares, for example, correspond to a cryptographic secret key (such as a signing key) as it allows an adversary to mount related-key attacks (see [BDL01]). In fact, most of the known constructions of threshold signatures use Shamir's secret sharing to distribute the signing key among the parties and hence they are all susceptible to such attacks.

Non-Malleable Secret Sharing. To protect a secret sharing scheme against such share tampering attacks, Goyal and Kumar [GK18a,GK18b] introduced the notion of Non-Malleable Secret Sharing. Roughly, a secret sharing scheme $(\mathsf{Share}, \mathsf{Rec})$ is non-malleable against a tampering function class $\mathcal{F}$ if for every $f \in \mathcal{F}$ and every secret s, $\mathsf{Rec}(f(\mathsf{shares}))$ where $\mathsf{shares} \leftarrow \mathsf{Share}(s)$ is either s or something that is unrelated to s.[1] Of course, we cannot hope to protect against all possible tampering functions as a function can first reconstruct the secret from the shares, multiply it by 2 and then share this value to obtain a valid sharing of a related secret. Thus, the prior works placed restrictions on the set of functions that can tamper the shares. A natural restricted family of tampering functions that we will consider in this work is $\mathcal{F}_{ind}$ which consists of the set of all functions that tamper each share independently.

[1] See Sect. 3 for a precise definition.

Connection to Non-Malleable Codes. Non-malleable secret sharing is related to another cryptographic primitive called as Non-Malleable Codes which was introduced in an influential work by Dziembowski, Pietrzak and Wichs [DPW10].[2] A non-malleable code relaxes the usual notion of error correction by requiring that the decoding procedure outputs either the original message or something that is independent of the message when given a tampered codeword as input. A beautiful line of work, starting from [DPW10], has given several constructions of non-malleable codes with security against various tampering function classes [LL12, DKO13, FMNV14, FMVW14, ADL14, AGM+15, FMNV15, JW15, CKR16, CGM+16, AAG+16, CGL16, BDKM16, Li17, KOS17, CL17, KOS18, BDKM18, GMW17, OPVV18, KLT18, BDG+18].

We now elaborate on the connection between non-malleable codes and non-malleable secret sharing. A tampering function family in the literature of non-malleable codes that is somewhat similar to $\mathcal{F}_{ind}$ is the k-split-state function family. A k-split-state function compartmentalizes a codeword into k-parts and applies a tampering function to each part, independent of the other parts. Seeing the similarity between $\mathcal{F}_{ind}$ and k-split-state functions, it might be tempting to conclude that a non-malleable code against a k-split-state function family is in fact a k-out-of-k non-malleable secret sharing. However, as demonstrated in [GK18a], this might not be true in general. In particular, [GK18a] showed that even a 3-split-state non-malleable code need not be a 3-out-of-3 non-malleable secret sharing as non-malleable codes may not always protect the secrecy of the message. In particular, the first few bits of the codeword could reveal some bits of the message and still, this coding scheme could be non-malleable. Nevertheless, for the special case of 2, Aggarwal et al. [ADKO15] showed that any 2-split-state non-malleable code is indeed a 2-out-of-2 non-malleable secret sharing scheme. In the other direction, we note that any k-out-of-k non-malleable secret sharing scheme against $\mathcal{F}_{ind}$ is in fact a k-split-state non-malleable code.

Rate of Non-Malleable Secret Sharing. One of the main efficiency parameters in any secret sharing scheme is its *rate* which is defined as the ratio between the length of the secret and the maximum size of a share. In the prior work, Goyal and Kumar [GK18a] gave an elegant construction of t-out-of-n non-malleable secret sharing from any 2-split-state non-malleable code. However, the rate of this scheme is equal to $O(\frac{1}{n \log m})$ where m is the length of the secret. The rate tends to 0 as the length of the secret m tends to ∞ and hence, a natural question to ask is:

Can we obtain a construction of threshold non-malleable secret sharing with rate > 0?

The problem of improving the rate was mentioned as an explicit open question in [GK18a].

[2] We refer the reader to [GK18a, GK18b] for a thorough discussion on the connection between non-malleable secret sharing and related notions such as verifiable secret sharing [CGMA85] and AMD codes [CDF+08].

Multiple Tamperings. In the real world, a tampering adversary could potentially mount more than one tampering attack. In particular, if each share of a cryptographic secret key is stored on a small device (such as smart cards), the adversary could potentially clone these devices to obtain multiple copies of the shares. The adversary could then apply a different tampering function on each copy and obtain information about related secrets. Thus, a more realistic security definition would be to consider multiple tampering functions $f_1, \ldots, f_k \in \mathcal{F}$, and require that for every secret s, the joint distribution $(\mathsf{Rec}(f_1(\mathsf{shares})), \ldots, \mathsf{Rec}(f_k(\mathsf{shares})))$ where $\mathsf{shares} \leftarrow \mathsf{Share}(s)$ is independent of s.[3] For the case of non-malleable codes, security against multiple tamperings has already been considered in [FMNV14, JW15, CGL16, OPVV18]. However, for the case of non-malleable secret sharing, the prior work [GK18a] only considered a single tampering function and a natural question would be:

Can we obtain a construction of threshold non-malleable secret sharing against multiple tamperings?

1.1 Our Results

In this work, we obtain the following results.

Rate Improvement. We give the first construction of a threshold non-malleable secret sharing scheme that has rate > 0. Specifically, the rate of our construction is $\Theta(\frac{1}{t \log^2 n})$ where t is the threshold and n is the number of parties. More formally,

Theorem 1. *For any $n, t \geq 4$ and any $\rho > 0$, there exists a construction of t-out-of-n non-malleable secret sharing scheme against $\mathcal{F}_{ind}$ for sharing m-bit secrets for any $m > \log n$ with rate $\Theta(\frac{1}{t \log^2 n})$ and simulation error $2^{-\Omega(\frac{m}{\log^{1+\rho} m})}$. The running times of the sharing and reconstruction algorithms are polynomial in n and m.*

Local Leakage Resilient Secret Sharing. One of the main tools used in proving Theorem 1 (which may be of independent interest) is an efficient construction of *local leakage-resilient* threshold secret sharing scheme [GK18a, BDIR18]. A t-out-of-n secret sharing scheme is said to be local leakage-resilient (parameterized by a leakage bound μ and set size s), if the secrecy holds against any adversary who might obtain at most $t - 1$ shares in the clear and additionally, for any set $S \subseteq [n]$ of size at most s, the adversary obtains μ bits from each share belonging to a party in the set S. Goyal and Kumar [GK18a] gave a construction of a 2-out-of-n local leakage resilient secret sharing scheme. In this work, we give an efficient construction of t-out-of-n local leakage resilient secret sharing

[3] As in the case of single tampering, a tampering function could just output the same shares and in which the reconstructed secret will be s. Our definition also captures this property and we refer to Sect. 3 for a precise definition.

scheme when t is a constant. This result must be contrasted with a recent result by Benhamouda et al. [BDIR18] who showed that the Shamir's secret sharing scheme is local leakage resilient when the field size is sufficiently large and the threshold $t = n - o(\log n)$. A more precise statement of our construction of local leakage resilient secret sharing scheme appears below.

Theorem 2. *For any $\epsilon > 0$, $t, n \in \mathbb{N}$, and parameters $\mu \in \mathbb{N}, s \leq n$, there exists an efficient construction of t-out-of-n secret sharing scheme for sharing m-bit secrets that is (μ, s)-local leakage resilient with privacy error ϵ. The size of each share when t is a constant is $O\left((m + s\mu + \log(\log n/\epsilon)) \log n\right)$.*

Concrete Efficiency. A major advantage of our result is its *concrete efficiency*. In the prior work, the constant hidden inside the big-O notation was large and was not explicitly estimated. We have optimized the parameters of our construction and we illustrate the size of shares for various values of (n, t) in Table 1.[4]

Table 1. Share sizes for simulation error of at most 2^{-80}.

(# of Parties, Threshold)	Secret length (in bits)	Share size (in KB)
$(7, 4)$	812	273.73
$(9, 5)$	812	399.85
$(25, 13)$	812	1757.53
$(100, 51)$	812	12.34×10^3
$(7, 4)$	1024	345.19
$(9, 5)$	1024	504.24
$(25, 13)$	1024	2216.40
$(100, 51)$	1024	15.56×10^3

Comparison with [GK18a]. When compared to the result of [GK18a] which could support thresholds $t \geq 2$, our construction can only support threshold $t \geq 4$. However, getting a rate > 0 non-malleable secret sharing scheme for threshold $t = 2$ would imply a 2-split-state non-malleable code with rate > 0 which is a major open problem. For the case of $t = 3$, though we know constructions of 3-split-state non-malleable codes with rate > 0 [KOS18, GMW17], they do not satisfy the privacy property of a 3-out-of-3 secret sharing scheme. In particular, given two states of the codeword, some information about the message is leaked. Thus, getting a 3-out-of-n non-malleable secret sharing scheme with rate > 0 seems out of reach of the current techniques and we leave this as an open problem.

[4] 812 bits is the minimal message length that gives 80 bits of security.

Multiple Tampering. We initiate the study of non-malleable secret sharing under multiple tampering. Here, the shares can be subject to multiple (possibly different) tampering functions and we require that the joint distribution of the reconstructed secrets to be independent of s. For this stronger security notion, we first prove a negative result that states that a non-malleable secret sharing cannot exist when the number of tamperings (also called as the tampering degree) is apriori unbounded. This result generalizes a similar result for the case of a split-state non-malleable codes. Formally,

Theorem 3. *For any $n, t \in \mathbb{N}$, there does not exist a t-out-of-n non-malleable secret sharing scheme against $\mathcal{F}_{ind}$ that can support an apriori unbounded tampering degree.*

When the tampering degree is apriori bounded, we get constructions of threshold non-malleable secret sharing scheme. Formally,

Theorem 4. *For any $n, t \geq 4$, and $\mathsf{K} \in \mathbb{N}$, there exists a t-out-of-n non-malleable secret sharing scheme with tampering degree K for sharing m-bit secrets for a large enough[5] m against $\mathcal{F}_{ind}$ with rate $= \Theta(\frac{1}{\mathsf{K}^3 t \log^2 n})$ and simulation error $2^{-m^{\Omega(1)}}$. The running time of the sharing and reconstruction algorithms are polynomial in n and m.*

General Access Structures. We extend our techniques used in the proof of Theorems 1, 4 to give constructions of non-malleable secret sharing scheme for more general monotone access structures rather than just threshold structures. Before we state our result, we give some definitions.

Definition 1. *An access structure $\mathcal{A}$ is said to be monotone if for any set $S \in A$, any superset of S is also in $\mathcal{A}$. A monotone access structure $\mathcal{A}$ is said to be 4-monotone if for any set $S \in \mathcal{A}$, $|S| \geq 4$.*

We also give the definition of a minimal authorized set.

Definition 2. *For a monotone access structure $\mathcal{A}$, a set $S \in \mathcal{A}$ is a minimal authorized set if any strict subset of S is not in $\mathcal{A}$. We denote t_{max} to be $\max |S|$ where S is a minimal authorized set of $\mathcal{A}$.*

We now state our extension to general access structures.

Theorem 5. *For any $n, \mathsf{K} \in \mathbb{N}$ and 4-monotone access structure $\mathcal{A}$, if there exists a statistically private (with privacy error ϵ) secret sharing scheme for $\mathcal{A}$ that can share m-bit secrets for a large enough m with rate R, there exists a non-malleable secret sharing scheme for sharing m-bit secrets for the same access structure $\mathcal{A}$ with tampering degree K against $\mathcal{F}_{ind}$ with rate $\Theta(\frac{R}{\mathsf{K}^3 t_{\max} \log^2 n})$ and simulation error $\epsilon + 2^{-m^{\Omega(1)}}$.*

Thus, starting with a secret sharing scheme for monotone span programs [KW93] or for more general access structures [LV18], we get non-malleable secret sharing schemes for the same access structures with comparable rate.

[5] See the main body for the precise statement.

Comparison with [GK18b]. In the prior work [GK18b], the rate of the non-malleable secret sharing for general access structures also depended on the length of the message and thus, even when R is constant, their construction could only achieve a rate of 0. However, unlike our construction, they could support all monotone access structures (and not just 4-monotone) and they could even start with a computational secret sharing scheme for an access structure $\mathcal{A}$ and convert it to a non-malleable secret sharing scheme for $\mathcal{A}$.

Concurrent Work. In a concurrent and independent work, Aggarwal et al. [ADN+18] consider the multiple tampering model and give constructions of non-malleable secret sharing for general access structures in this model. There are three main differences between our work and their work. Firstly, the rate of their construction asymptotically tends to 0 even for the threshold case. However, the rate of our construction is greater than 0 when we instantiate the compiler with a rate > 0 secret sharing scheme. Secondly, their work considers a stronger model wherein each tampering function can choose a different reconstruction set. We prove the security of our construction in a weaker model wherein the reconstruction set is the same for each tampering function. We note that the impossibility result for unbounded tampering holds even if the reconstruction set is the same. Thirdly, their construction can give non-malleable secret sharing scheme for any 3-monotone access structure whereas our construction can only work for 4-monotone access structure. In another concurrent and independent work, Kumar et al. [KMS18] gave a construction of non-malleable secret sharing in a stronger model where the tampering functions might obtain bounded leakage from the other shares.

2 Our Techniques

In this section, we give a high level overview of the techniques used to obtain our results.

2.1 Rate Improvement

Goyal and Kumar [GK18a] *Approach.* We first give a brief overview of the construction of threshold non-malleable secret sharing of Goyal and Kumar [GK18a] and then explain why it could achieve only a rate of 0. At a high level, Goyal and Kumar start with any 2-split-state non-malleable code and convert it into a t-out-of-n non-malleable secret sharing scheme. We only explain their construction for the case when $t \geq 3$, and for the case of $t = 2$, they gave a slightly different construction. For the case when $t \geq 3$, the sharing procedure does the following. The secret is first encoded using a 2-split-state non-malleable code to obtain the two states L and R. L is now shared using any t-out-of-n secret sharing scheme, say Shamir's secret sharing to get the shares $\mathsf{SL}_1, \ldots, \mathsf{SL}_n$ and R is shared using a 2-out-of-n local leakage resilient secret sharing scheme to get the shares $\mathsf{SR}_1, \ldots, \mathsf{SR}_n$. The share corresponding to party i includes $(\mathsf{SL}_i, \mathsf{SR}_i)$.

To recover the secret given at least t shares, the parties first use the recovery procedures of the threshold secret sharing scheme and local leakage resilient secret sharing scheme to recover L and R respectively. Later, the secret is obtained by decoding L and R using the decoding procedure of the non-malleable code. The correctness of the construction is straightforward and to argue secrecy, it can been seen that given any set of $t-1$ shares, L is perfectly hidden and this follows from the security of Shamir's secret sharing. Now, using the fact that any 2-split-state non-malleable code is a 2-out-of-2 secret sharing scheme, it can be shown that the right state R statistically hides the secret.

To argue the non-malleability of this construction, Goyal and Kumar showed that any tampering attack on the secret sharing scheme can be reduced to a tampering attack on the underlying 2-split-state non-malleable code. The main challenge in designing such a reduction is that the tampering functions against the underlying non-malleable code must be split-state, meaning that the tampering function against L (denoted by f) must be independent of R and the tampering function against R (denoted by g) must be independent of L. To make the tampering function g to be independent of L, [GK18a] made use of the fact that there is an inherent difference in the parameters used for secret sharing L and R. Specifically, since R is shared using a 2-out-of-n secret sharing scheme, the tampered right state $\widetilde{\mathsf{R}}$ can be recovered from any two tampered shares, say $\widetilde{\mathsf{SR}}_1, \widetilde{\mathsf{SR}}_2$. Now, since L is shared using a t-out-of-n secret sharing scheme and $t \geq 3$, the shares SL_1 and SL_2 information theoretically provides no information about L. This, in particular means that we can fix the shares SL_1 and SL_2 independent of L and the tampering function g could use these fixed shares to output the tampered right state $\widetilde{\mathsf{R}}$. Now, when f is given the actual L, it can sample $\mathsf{SL}_3, \ldots, \mathsf{SL}_n$ as a valid secret sharing of L that is consistent with the fixed $\mathsf{SL}_1, \mathsf{SL}_2$. This allowed them to argue one-sided independence i.e., g is independent of L. On the other hand, making the tampering function f to be independent of R is a lot trickier. This is because any two shares information theoretically fixes R and in order to recover $\widetilde{\mathsf{L}}$, we need at least t ($\geq$3) shares. Hence, we may not be able to argue that f is independent of R. To argue this independence, Goyal and Kumar used the fact that R is shared using a *local leakage resilient* secret sharing scheme. In particular, they made the size of SR_i to be much larger than the size of SL_i and showed that even when we leak $|\mathsf{SL}_i|$ bits from each share SR_i, R is still statistically hidden. This allowed them to define leakage functions $\mathsf{leak}_1, \ldots, \mathsf{leak}_n$ where leak_i had SL_i hardwired in its description, it applies the tampering function on $(\mathsf{SL}_i, \mathsf{SR}_i)$ and outputs the tampered $\widetilde{\mathsf{SL}}_i$. Now, from the secrecy of the local leakage resilient secret sharing scheme, the distribution $\widetilde{\mathsf{SL}}_1, \ldots, \widetilde{\mathsf{SL}}_n$ (which completely determines $\widetilde{\mathsf{L}}$) is independent of R and thus $\widetilde{\mathsf{L}}$ is independent of R. This allowed them to obtain two-sided independence.

A drawback of this approach is that the rate of this scheme is at least as bad as that of the underlying 2-split-state non-malleable code. As mentioned before, obtaining a 2-split-state non-malleable code with rate > 0 is a major open problem. Thus, this construction could only achieve a rate of 0.

Our Approach. While constructing 2-split-state non-malleable code with rate >0 has been notoriously hard, significant progress has been made for the case of 3-split-state non-malleable codes. Very recently, independent works of Gupta et al. [GMW17] and Kanukurthi et al. [KOS18] gave constructions of 3-split-state non-malleable codes with an explicit constant rate. The main idea behind our rate-improved construction is to use a constant rate, 3-split-state non-malleable code instead of a rate 0, 2-split-state non-malleable code. To be more precise, we first encode the secret using a 3-split-state non-malleable code to get the three states $(\mathsf{L}, \mathsf{C}, \mathsf{R})$. We then share the first state L using a t-out-of-n secret sharing scheme to get $(\mathsf{SL}_1, \ldots, \mathsf{SL}_n)$ as before. Then, we share C using a t_1-out-of-n secret sharing scheme to get $(\mathsf{SC}_1, \ldots, \mathsf{SC}_n)$ and R using a t_2-out-of-n secret sharing scheme to get $(\mathsf{SR}_1, \ldots, \mathsf{SR}_n)$. Here, t_1, t_2 are some parameters that we will fix later. The share corresponding to party i includes $(\mathsf{SL}_i, \mathsf{SC}_i, \mathsf{SR}_i)$. While the underlying intuition behind this idea is natural, proving that this construction is a non-malleable secret sharing scheme faces several barriers which we elaborate below.

First Challenge. The first barrier that we encounter is, unlike a 2-split-state non-malleable code which is always a 2-out-of-2 secret sharing scheme, a 3-split-state non-malleable code may not be a 3-out-of-3 secret sharing scheme. In particular, we will not be able use the [GK18a] trick of sharing the 3-states using secret sharing schemes with different thresholds to gain one-sided independence. This is because given $t-1$ shares, complete information about two states will be revealed, and we could use these two states to gain some information about the underlying message. Thus, the privacy of the scheme breaks down. Indeed, as mentioned in the introduction, the constructions of Kanukurthi et al. [KOS18] and Gupta et al. [GMW17] are not 3-out-of-3 secret sharing schemes.

The main trick that we use to solve this challenge is that, while these constructions [KOS18, GMW17] are not 3-out-of-3 secret sharing schemes, we observe that there exist two states (let us call them C and R) such that these two states statistically hide the message. This means that we can potentially share these two states using secret sharing schemes with smaller thresholds and may use it to argue one-sided independence.

Second Challenge. The second main challenge is in ensuring that the tampering functions we design for the underlying 3-split-state non-malleable code are indeed split-state. Let us call the tampering functions that tamper L, C, and R as f, g, and h respectively. To argue that f, g and h are split-state, we must ensure f is independent of C and R and similarly, g is independent of L and R and h is independent of L and C. For the case of 2-split-state used in the prior work, this independence was achieved by using secret sharing with different thresholds and relying on the leakage resilience property. For the case of 3-split-state, we need a more sophisticated approach of *stratifying* the three secret sharing schemes so that we avoid circular dependence in the parameters. We now elaborate more on this solution.

To make g and h to be independent of L, we choose the thresholds t_1 and t_2 to be less than t. This allows us to fix a certain number of shares independent of L and use these shares to extract $\widetilde{\mathsf{C}}$ and $\widetilde{\mathsf{R}}$. Similarly, to make h to be independent of C, we choose the threshold $t_2 < t_1$. This again allows us to fix certain shares C and use them to extract $\widetilde{\mathsf{R}}$. Thus, by choosing $t > t_1 > t_2$, we could achieve something analogous to one-sided independence. Specifically, we achieved independence of g from L and independence of h from (L, C). For complete split-state property, we still need to make sure that f is independent of (C, R) and g is independent of R. To make the tampering function f to be independent of C, we rely on the local leakage resilience property of the t_1-out-of-n secret sharing scheme. That is, we make the size of the shares SC_i to be much larger than SL_i such that, in spite of leaking $|\mathsf{SL}_i|$ bits from each share SC_i, the secrecy of C is maintained. We can use this to show that the joint distribution $(\widetilde{\mathsf{SL}}_1, \ldots, \widetilde{\mathsf{SL}}_n)$ (which completely determines $\widetilde{\mathsf{L}}$) is independent of C. Now, to argue that both f and g are independent of R, we rely on the local leakage resilience property of the t_2-out-of-n secret sharing scheme. That is, we make the shares of SR_i to be much larger than $(\mathsf{SL}_i, \mathsf{SC}_i)$ so that, in spite of leaking $|\mathsf{SL}_i| + |\mathsf{SC}_i|$ bits from each share SR_i, the secrecy of R is maintained. We then use this property to argue that the joint distribution $(\widetilde{\mathsf{SL}}_1, \widetilde{\mathsf{SC}}_1), \ldots, (\widetilde{\mathsf{SL}}_n, \widetilde{\mathsf{SC}}_n)$ is independent of R. Thus, the idea of stratifying the three threshold secret sharing schemes with different parameters as described above allows to argue that f, g and h are split-state. As we will later see, this technique of stratification is very powerful and it allows us to easily extend this construction to more general monotone access structures.

Third Challenge. The third and the more subtle challenge is the following. To reduce the tampering attack on the secret sharing scheme to a tampering attack on the underlying non-malleable code, we must additionally ensure *consistency* i.e., the tampered message output by the split-state functions must be statistically close to the message output by the tampering experiment of the underlying secret sharing scheme. To illustrate this issue in some more detail, let us consider the tampering functions f and g in the construction of Goyal and Kumar [GK18a] for the simple case when $n = t = 3$. Recall that the tampering function g samples $\mathsf{SR}_1, \mathsf{SR}_2$ such that it is a valid 2-out-of-n secret sharing of R and uses the fixed $\mathsf{SL}_1, \mathsf{SL}_2$ (independent of L) to extract the tampered $\widetilde{\mathsf{R}}$ from $(\widetilde{\mathsf{SR}}_1, \widetilde{\mathsf{SR}}_2)$. However, note that g cannot use any valid secret sharing of $\mathsf{SR}_1, \mathsf{SR}_2$ of R. In particular, it must also satisfy the property that the tampering function applied on $\mathsf{SL}_1, \mathsf{SR}_1$ gives the exact same $\widetilde{\mathsf{SL}}_1$ that f uses in the reconstruction (a similar condition for position 2 must be satisfied). This is crucial, as otherwise there might be a difference in the distributions of the tampered message output by the split-state functions and the message output in the tampering experiment of the secret sharing scheme. In case there is a difference, we cannot hope to use the adversary against the non-malleable secret sharing to break the underlying non-malleable code. This example illustrates this issue for a simple case when $t = n = 3$. To ensure consistency for larger values of n and t, Goyal and Kumar

fixed $(\mathsf{SL}_1, \ldots, \mathsf{SL}_{t-1})$ (instead of just fixing $\mathsf{SL}_1, \mathsf{SL}_2$) and the function g ensures consistency of each of the tampered shares $\widetilde{\mathsf{SL}}_1, \ldots, \widetilde{\mathsf{SL}}_{t-1}$. However, this approach completely fails when we move to 3 states. For the case of 3-states, the tampering function, say h, must sample $\mathsf{SR}_1, \ldots, \mathsf{SR}_n$ such that it is consistent with $\widetilde{\mathsf{SL}}_1, \ldots, \widetilde{\mathsf{SL}}_{t-1}$ used by f. However, even to check this consistency, h would need the shares $\mathsf{SC}_1, \ldots, \mathsf{SC}_{t-1}$ which completely determines C. In this case, we cannot argue that h is independent of C.

To tackle this challenge, we deviate from the approach of Goyal and Kumar [GK18a] and have a new proof strategy that ensures consistency and at the same time maintains the split-state property. In this strategy, we only fix the values $(\mathsf{SL}_1, \mathsf{SL}_2, \mathsf{SL}_3)$ for the first secret sharing scheme, $(\mathsf{SC}_1, \mathsf{SC}_2)$ for the second secret sharing scheme and fix SR_3 for the third secret sharing scheme. Note that we consider $t \geq 4$, $t_1 \geq 3$ and $t_2 \geq 2$ and thus, the fixed shares are independent of L, C, and R respectively.[6] We design our split-state functions in such a way that the tampering function f need not do any consistency checks, the tampering function g has to do the consistency check only on $\widetilde{\mathsf{SL}}_3$ (which it can do since SL_3 and SR_3 are fixed) and the function h needs to do a consistency check only on $\{\widetilde{\mathsf{SL}}_i, \widetilde{\mathsf{SC}}_i\}_{i\in[1,2]}$ (which it can do since $\mathsf{SL}_1, \mathsf{SC}_1, \mathsf{SL}_2, \mathsf{SC}_2$ are fixed). This approach of reducing the number of checks to maintain consistency helps us in arguing independence between the tampering functions. However, this approach creates additional problems in extracting $\widetilde{\mathsf{L}}$ as the tampering function f needs to use the shares $(\mathsf{SR}_4, \ldots, \mathsf{SR}_n)$ and $(\mathsf{SC}_4, \ldots, \mathsf{SC}_n)$ (which completely determines C and R respectively). We solve this by letting f extract $\widetilde{\mathsf{L}}$ using shares of some arbitrary values of C and R and we then use the leakage resilience property to ensure that the outputs in the split-state tampering experiment and the secret sharing tampering experiment are statistically close.

Completing the Proof. This proof strategy helps us in getting a rate > 0 construction of a t-out-of-n non-malleable secret sharing scheme for $t \geq 4$. However, there is one crucial block that is still missing. Goyal and Kumar [GK18a] only gave a construction of 2-out-of-n local leakage resilient secret sharing scheme. And, for this strategy to work we also need a construction of t_1-out-of-n local leakage resilient secret sharing scheme for some $t_1 > 2$. As mentioned in the introduction, the recent work by Benhamouda et al. [BDIR18] only gives a construction of local leakage resilient secret sharing when the threshold value is large (in particular, $n - o(\log n)$). To solve this, we give an efficient construction of a t-out-of-n local leakage resilient secret sharing scheme when t is a constant. This is in fact sufficient to get a rate > 0 construction of non-malleable secret sharing scheme. We now give details on the techniques used in this construction.

Local Leakage Resilient Secret Sharing Scheme. The starting point of our construction is the 2-out-of-2 local leakage resilient secret sharing from the work of Goyal and Kumar [GK18a] based on the inner product two-source extractor [CG88]. We first extend it to a k-out-of-k local leakage resilient secret sharing

[6] This is the reason why we could only achieve thresholds $t \geq 4$.

scheme for any arbitrary k. Let us now illustrate this for the case when k is even i.e., $k = 2p$. To share a secret s, we first additively secret share s into $s_1, \ldots, s_p$ and we encode each s_i using the 2-out-of-2 leakage resilient secret sharing scheme to obtain the shares $(\mathsf{share}_{2i-1}, \mathsf{share}_{2i})$. We then give share_i to party i for each $i \in [k]$. Note that given $t-1$ shares, at most $p-1$ additive secret shares can be revealed. We now rely on the local leakage resilience property of the 2-out-of-2 secret sharing to argue that the final additive share is hidden even when given bounded leakage from the last share. This helps us in arguing the k-out-k local leakage resilience property. The next goal is to extend this to a k-out-of-n secret sharing scheme. Since we are interested in getting good rate, we should not increase the size of the shares substantially. A naïve way of doing this would be to share the secret $\binom{n}{k}$ times (one for each possible set of k-parties) using the k-out-of-k secret sharing scheme and give the respective shares to the parties. The size of each share in this construction would blow up by a factor $\binom{n}{k-1}$ when compared to the k-out-of-k secret sharing scheme. Though, this is polynomial in n when k is a constant, this is clearly sub-optimal when n is large and would result in bad concrete parameters. We note that Goyal and Kumar [GK18a] used a similar approach to obtain a 2-out-of-n local leakage resilient secret sharing.

In this work, we use a very different approach to construct a k-out-of-n local leakage resilient secret sharing from a k-out-of-k local leakage resilient secret sharing. The main advantage of this transformation is that it is substantially more rate efficient than the naïve solution. Our transformation makes use of combinatorial objects called as perfect hash functions [FK84].[7] A family of functions mapping $\{1, \ldots, n\}$ to $\{1, \ldots, k\}$ is said to be a perfect hash function family if for every set $S \subseteq [n]$ of size at most k, there exists at least one function in the family that is injective on S. Let us now illustrate how this primitive is helpful in extending a k-out-of-k secret sharing scheme to a k-out-of-n secret sharing scheme. Given a perfect hash function family $\{h_i\}_{i \in [\ell]}$ of size ℓ, we share the secret s independently ℓ times using the k-out-of-k secret sharing scheme to obtain $(\mathsf{share}^i_1, \ldots, \mathsf{share}^i_k)$ for each $i \in [\ell]$. We now set the shares corresponding to party i as $(\mathsf{share}^1_{h_1(i)}, \ldots, \mathsf{share}^\ell_{h_\ell(i)})$. To recover the secret from some set of k shares given by $S = \{s_1, \ldots, s_k\}$, we use the following strategy. Given any subset S of size k, perfect hash function family guarantees that there is at least one index $i \in [\ell]$ such that h_i is injective on S. We can now use $\{\mathsf{share}^i_{h_i(s_1)}, \ldots, \mathsf{share}^i_{h_i(s_k)}\} = \{\mathsf{share}^i_1, \ldots, \mathsf{share}^i_k\}$ to recover the secret using the reconstruction procedure of the k-out-of-k secret sharing.

We show that this transformation additionally preserves local leakage resilience. In particular, if we start with a k-out-of-k local leakage resilient secret sharing scheme then we obtain a k-out-of-n local leakage resilient secret sharing. The size of each share in our k-out-of-n leakage resilient secret sharing scheme is ℓ times the share size of k-out-of-k secret sharing scheme. Thus, to minimize

[7] We note that using perfect hash function families for constructing threshold secret sharing scheme is not new (see [Bla99,SNW01] for a comprehensive discussion). However, to the best of our knowledge, this is the first application of this technique to construct local leakage resilient secret sharing scheme.

rate we must minimize the size of the perfect hash function family. Constructing perfect hash function family of minimal size for all $k \in \mathbb{N}$ is an interesting and a well-known open problem in combinatorics. In this work, we give an efficient randomized construction (with good concrete parameters) of a perfect hash function family for a constant k with size $O(\log n + \log(1/\epsilon))$ where ϵ is the error probability. Alternatively, we can also use the explicit construction (which is slightly less efficient when compared to the randomized construction) of size $O(\log n)$ (when k is a constant) given by Alon et al. [AYZ95]. Combining either the randomized/explicit construction of perfect hash function family with a construction of k-out-of-k local leakage resilient secret sharing scheme, we get an efficient construction of k-out-of-n local leakage resilient secret sharing scheme when k is a constant.

2.2 Multiple Tampering

We also initiate the study of non-malleable secret sharing under multiple tamperings. As discussed in the introduction, this is a much stronger model when compared to that of a single tampering.

Negative Result. We first show that when the number of tampering functions that can maul the secret sharing scheme is apriori unbounded, there does not exist any threshold non-malleable secret sharing scheme. This generalizes a similar result for the case of split-state non-malleable code (see [GLM+04,FMNV14] for details) and the main idea is inspired by these works. The underlying intuition behind the negative result is simple: we come up with a set of tampering functions such that each tampering experiment leaks one bit of a share. Now, given the outcomes of $t \cdot s$ such tampering experiments where s is the size of the share, the distinguisher can clearly learn every bit of t shares and thus, learn full information about the underlying secret and break non-malleability.

For the tampering experiment to leak one bit of the share of party i, we use the following simple strategy. Let us fix an authorized set of size t say, $\{1, \ldots, t\}$. We choose two sets of shares: $\{\mathsf{share}_1, \ldots, \mathsf{share}_i, \ldots, \mathsf{share}_t\}$ and $\{\mathsf{share}_1, \ldots, \mathsf{share}'_i, \ldots, \mathsf{share}_t\}$ such that they reconstruct to two different secrets. Note that the privacy of a secret sharing scheme guarantees that such shares must exist. Whenever the particular bit of the share of party i is 1, the tampering function f_i outputs share'_i whereas the other tampering functions, say f_j will output share_j. On the other hand, if the particular bit is 0 then the tampering function f_i outputs share_i and the other tampering functions still output share_j. Observe that the reconstructed secret in the two cases reveals the particular bit of the share of party i. We can use a similar strategy to leak every bit of all the t shares which completely determine the secret.

Positive Result. We complement the negative result by showing that when the number of tamperings is apriori bounded, we can obtain an efficient construction of a threshold non-malleable secret sharing scheme. A natural approach would be to start with a split-state non-malleable code that is secure against bounded

tamperings and convert it into a non-malleable secret sharing scheme. To the best of our knowledge, the only known construction of split-state non-malleable code that is secure in the presence of bounded tampering is that of Chattopadhyay et al. [CGL16]. However, the rate of this code is 0 even when we restrict ourselves to just two tamperings. In order to achieve a better rate, we modify the constructions of Kanukurthi et al. [KOS18] and Gupta et al. [GMW17] such that we obtain a 3-split-state non-malleable code that secure in the setting of bounded tampering. The rate of this construction is $O(\frac{1}{k})$ where k is the apriori bound on the number of tamperings. Fortunately, even in this construction, we still maintain the property that there exists two states that statistically hide the message. We then prove that the same construction described earlier is a secure non-malleable secret sharing under bounded tampering when we instantiate the underlying code with a bounded tampering secure 3-split-state non-malleable codes.

2.3 General Access Structures

To obtain a secret sharing scheme for more general access structures, we start with any statistically secure secret sharing scheme for that access structure, and use it to share L instead of using a threshold secret sharing scheme. We require that the underlying access structure to be 4-monotone so that we can argue the privacy of our scheme. Recall that a 4-monotone access structure is one in which the size of every set in the access structure is at least 4. Even in this more general case, the technique of stratifying the secret sharing schemes allows us to prove non-malleability in almost an identical fashion to the case of threshold secret sharing. We remark that the work of [GK18b] which gave constructions of non-malleable secret sharing scheme for general monotone access structures additionally required their local leakage resilient secret sharing scheme to satisfy a security property called as strong local leakage resilience. Our construction does not require this property and we show that "plain" local leakage resilience is sufficient for extending to more general monotone access structures.

Organization. We give the definitions of non-malleable secret sharing and non-malleable codes in Sect. 3. In Sect. 4, we present the construction of the k-out-of-n leakage resilient secret sharing scheme. In Sect. 5, we describe our rate-efficient threshold non-malleable secret sharing scheme for the single tampering. We give the impossibility result for unbounded many tamperings in the full version. Finally, in Sect. 6, we describe our result on non-malleable secret sharing for general access structures against multiple bounded tampering. Note that the result in Sect. 6 implicitly captures the result for threshold non-malleable secret sharing against bounded tampering. We present this more general result for ease of exposition.

3 Preliminaries

Notation. We use capital letters to denote distributions and their support, and corresponding lowercase letters to denote a sample from the same. Let $[n]$ denote the set $\{1, 2, \ldots, n\}$, and U_r denote the uniform distribution over $\{0,1\}^r$. For any $i \in [n]$, let x_i denote the symbol at the i-th co-ordinate of x, and for any $T \subseteq [n]$, let $x_T \in \{0,1\}^{|T|}$ denote the projection of x to the co-ordinates indexed by T. We write $\circ$ to denote concatenation. We give the standard definitions of min-entropy, statistical distance and seeded extractors in the full version.

3.1 Threshold Non-Malleable Secret Sharing Scheme

We first give the definition of a sharing function, then define a threshold secret sharing scheme and finally give the definition of a threshold non-malleable secret sharing. These three definitions are taken verbatim from [GK18a]. We define non-malleable secret sharing for more general access structures in the full version.

Definition 3 (Sharing Function). *Let $[n] = \{1, 2, \ldots, n\}$ be a set of identities of n parties. Let $\mathcal{M}$ be the domain of secrets. A sharing function* Share *is a randomized mapping from $\mathcal{M}$ to $\mathcal{S}_1 \times \mathcal{S}_2 \times \ldots \times \mathcal{S}_n$, where $\mathcal{S}_i$ is called the domain of shares of party with identity i. A dealer distributes a secret $m \in \mathcal{M}$ by computing the vector $\mathsf{Share}(m) = (\mathsf{S}_1, \ldots, \mathsf{S}_n)$, and privately communicating each share S_i to the party i. For a set $T \subseteq [n]$, we denote $\mathsf{Share}(m)_T$ to be a restriction of $\mathsf{Share}(m)$ to its T entries.*

Definition 4 ($(t, n, \epsilon_c, \epsilon_s)$-Secret Sharing Scheme). *Let $\mathcal{M}$ be a finite set of secrets, where $|\mathcal{M}| \geq 2$. Let $[n] = \{1, 2, \ldots, n\}$ be a set of identities (indices) of n parties. A sharing function* Share *with domain of secrets $\mathcal{M}$ is a $(t, n, \epsilon_c, \epsilon_s)$-secret sharing scheme if the following two properties hold:*

- ***Correctness:*** *The secret can be reconstructed by any t-out-of-n parties. That is, 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 $m \in \mathcal{M}$,*

$$\Pr[\mathsf{Rec}(\mathsf{Share}(m)_T) = m] = 1 - \epsilon_c$$

  where the probability is over the randomness of the Share *function. We will slightly abuse the notation and denote* Rec *as the reconstruction procedure that takes in T and $\mathsf{Share}(m)_T$ where T is of size at least t and outputs the secret.*
- ***Statistical Privacy:*** *Any collusion of less than t parties should have "almost" no information about the underlying secret. More formally, for any unauthorized set $U \subseteq [n]$ such that $|U| < t$, and for every pair of secrets $m_0, m_1 \in M$, for any distinguisher D with output in $\{0,1\}$, the following holds:*

$$|\Pr[D(\mathsf{Share}(m_0)_U) = 1] - \Pr[D(\mathsf{Share}(m_1)_U) = 1]| \leq \epsilon_s$$

We define the rate of the secret sharing scheme as

$$\lim_{|m|\to\infty} \frac{|m|}{\max_{i\in[n]} |\mathsf{Share}(m)_i|}$$

Definition 5 (Threshold Non-Malleable Secret Sharing [GK18a]). *Let* $(\mathsf{Share}, \mathsf{Rec})$ *be a* $(t, n, \epsilon_c, \epsilon_s)$*-secret sharing scheme for message space* $\mathcal{M}$*. Let* $\mathcal{F}$ *be some family of tampering functions. For each* $f \in \mathcal{F}$*,* $m \in \mathcal{M}$ *and authorized set* $T \subseteq [n]$ *containing* t *indices, define the tampered distribution* $\mathsf{Tamper}_m^{f,T}$ *as* $\mathsf{Rec}(f(\mathsf{Share}(m))_T)$ *where the randomness is over the sharing function* Share*. We say that the* $(t, n, \epsilon_c, \epsilon_s)$*-secret sharing scheme,* $(\mathsf{Share}, \mathsf{Rec})$ *is* ϵ'*-non-malleable w.r.t.* $\mathcal{F}$ *if for each* $f \in \mathcal{F}$ *and any authorized set* T *consisting of* t *indices, there exists a distribution* $D^{f,T}$ *over* $\mathcal{M} \cup \{\mathsf{same}^\star\}$ *such that:*

$$|\mathsf{Tamper}_m^{f,T} - \mathrm{copy}(D^{f,T}, m)| \leq \epsilon'$$

where copy *is defined by* $\mathrm{copy}(x, y) = \begin{cases} x & \text{if } x \neq \mathsf{same}^\star \\ y & \text{if } x = \mathsf{same}^\star \end{cases}$.

Many Tampering Extension. We now extend the above definition to capture multiple tampering attacks. Informally, we say that a secret sharing scheme is non-malleable w.r.t. family $\mathcal{F}$ with tampering degree K if for any set of K functions $f_1, \ldots, f_{\mathsf{K}} \in \mathcal{F}$, the output of the following tampering experiment is independent of the shared message m: (i) we first share a secret m to obtain the corresponding shares, (ii) we tamper the shares using $f_1, \ldots, f_{\mathsf{K}}$, (iii) we finally, output the K-reconstructed tampered secrets. Note that in the above experiment the message m is secret shared only once but is subjected to K (possibly different) tamperings. We refer to the full version for the formal definition.

3.2 Non-Malleable Codes

Dziembowski, Pietrzak and Wichs [DPW10] introduced the notion of non-malleable codes which generalizes the usual notion of error correction. In particular, it guarantees that when a codeword is subject to tampering attack, the reconstructed message is either the original one or something that is independent of the original message.

Definition 6 (Non-Malleable Codes [DPW10]). *Let* $\mathsf{Enc} : \{0,1\}^m \to \{0,1\}^n$ *and* $\mathsf{Dec} : \{0,1\}^n \to \{0,1\}^m \cup \{\perp\}$ *be (possibly randomized) functions, such that* $\mathsf{Dec}\big(\mathsf{Enc}(s)\big) = s$ *with probability 1 for all* $s \in \{0,1\}^m$*. Let* $\mathcal{F}$ *be a family of tampering functions and fix* $\varepsilon > 0$*. We say that* $(\mathsf{Enc}, \mathsf{Dec})$ *is* $\varepsilon-$non-malleable *w.r.t.* $\mathcal{F}$ *if for every* $f \in \mathcal{F}$*, there exists a random variable* D_f *on* $\{0,1\}^m \cup \{\mathsf{same}^\star\}$*, such that for all* $s \in \{0,1\}^m$*,*

$$|\mathsf{Dec}(f(X_s)) - \mathrm{copy}(D_f, s)| \leq \epsilon$$

where $X_s \leftarrow \mathsf{Enc}(s)$ *and* copy *is defined by* $\mathrm{copy}(x, y) = \begin{cases} x & \text{if } x \neq \mathsf{same}^\star \\ y & \text{if } x = \mathsf{same}^\star \end{cases}$*. We call* n *the* length *of the code and* m/n *the* rate.

Chattopadhyay, Goyal and Li [CGL16] defined a stronger notion of non-malleability against multiple tampering. We recall this definition in the full version of the paper.

Split-state Tampering Functions. We focus on the *split-state* tampering model where the encoding scheme splits s into c states: $\mathsf{Enc}(s) = (\mathsf{S}_1, \ldots, \mathsf{S}_c) \in \mathcal{S}_1 \times \mathcal{S}_2 \ldots \times \mathcal{S}_c$ and the tampering family is $\mathcal{F}_{split} = \{(f_1, \ldots, f_c) | f_i : \mathcal{S}_i \to \mathcal{S}_i\}$. We will call such a code as c-split-state non-malleable code.

Augmented Non-Malleable Codes. We recall the definition of augmented, 2-split-state non-malleable codes [AAG+16].

Definition 7 (Augmented Non-Malleable Codes [AAG+16]). *A coding scheme* $(\mathsf{Enc}, \mathsf{Dec})$ *with code length* $2n$ *and message length* m *is an augmented 2-split-state non-malleable code with error* ϵ *if for every function* $f, g : \{0,1\}^n \to \{0,1\}^n$*, there exists a random variable* $D_{(f,g)}$ *on* $\{0,1\}^n \times (\{0,1\}^m \cup \{\mathsf{same}^\star\})$ *such that for all messages* $s \in \{0,1\}^m$*, it holds that*

$$|(\mathsf{L}, \mathsf{Dec}(f(\mathsf{L}), g(\mathsf{R}))) - \mathcal{S}(D_{(f,g)}, s)| \leq \epsilon$$

where $(\mathsf{L}, \mathsf{R}) = \mathsf{Enc}(s)$*,* $(\mathsf{L}, \widetilde{m}) \leftarrow D_{f,g}$ *and* $\mathcal{S}((\mathsf{L}, \widetilde{m}), s)$ *outputs* (L, s) *if* $\widetilde{m} = \mathsf{same}^\star$ *and otherwise outputs* $(\mathsf{L}, \widetilde{m})$*.*

Explicit Constructions. We now recall the constructions of split-state non-malleable codes.

Theorem 6 ([Li17]). *For any* $n \in \mathbb{N}$*, there exists an explicit construction of 2-split-state non-malleable code with efficient encoder/decoder, code length* $2n$*, rate* $O(\frac{1}{\log n})$ *and error* $2^{-\Omega(\frac{n}{\log n})}$*.*

Theorem 7 ([KOS18, GMW17]). *For every* $n \in \mathbb{N}$ *and* $\rho > 0$*, there exists an explicit construction of 3-split-state non-malleable code with efficient encoder/decoder, code length* $(3 + o(1))n$*, rate* $\frac{1}{3+o(1)}$ *and error* $2^{-\Omega(n/\log^{1+\rho}(n))}$*.*

Theorem 8 ([CGL16]). *There exists a constant* $\gamma > 0$ *such that for every* $n \in \mathbb{N}$ *and* $t \leq n^\gamma$*, there exists an explicit construction of 2-split-state non-malleable code with an efficient encoder/decoder, tampering degree* t*, code length* $2n$*, rate* $\frac{1}{n^{\Omega(1)}}$ *and error* $2^{-n^{\Omega(1)}}$*.*

Theorem 9 ([GKP+18]). *There exists a constant* $\gamma > 0$ *such that for every* $n \in \mathbb{N}$ *and* $t \leq n^\gamma$*, there exists an explicit construction of an augmented, split-state non-malleable code with an efficient encoder/decoder, tampering degree* t*, code length* $2n$*, rate* $\frac{1}{n^{\Omega(1)}}$ *and error* $2^{-n^{\Omega(1)}}$*.*

Theorem 10. *There exists a constant* $\gamma > 0$ *such that for every* $n \in \mathbb{N}$ *and* $t \leq n^\gamma$*, there exists an explicit construction of 3-split-state non-malleable code with an efficient encoder/decoder, tampering degree* t*, code length* $3n$*, rate* $\Theta(\frac{1}{t})$ *and error* $2^{-n^{\Omega(1)}}$*.*

We give the proof of this theorem in the full version.

Additional Property. We show in the full version that the construction given in [KOS18,GMW17] satisfies the property that given two particular states of the codeword, the message remains statistically hidden.

4 *k*-out-of-n Leakage Resilient Secret Sharing Scheme

In this section, we give a new, rate-efficient construction of k-out-of-n leakage resilient secret sharing scheme for a constant k. Later, in Sect. 5, we will use this primitive along with a 3-split-state non-malleable code with explicit constant rate (see Theorem 7) from the works of Kanukurthi et al. [KOS18] and Gupta et al. [GMW17] to construct a t-out-of-n non-malleable secret sharing scheme with the above mentioned rate.

We first recall the definition of a leakage resilient secret sharing scheme from [GK18a].

Definition 8 (Leakage Resilient Secret Sharing [GK18a]). *A $(t, n, \epsilon_c, \epsilon_s)$ (for $t \geq 2$) secret sharing scheme* $(\mathsf{Share}, \mathsf{Rec})$ *for message space $\mathcal{M}$ is said to be ϵ-leakage resilient against a leakage family $\mathcal{F}$ if for all functions $f \in \mathcal{F}$ and for any two messages $m_0, m_1 \in \mathcal{M}$:*

$$|f(\mathsf{Share}(m_0)) - f(\mathsf{Share}(m_1))| \leq \epsilon$$

Leakage Function Family. We are interested in constructing leakage resilient secret sharing schemes against the specific function family $\mathcal{F}_{k,\overline{k},\overrightarrow{\mu}} = \{f_{K,\overline{K},\overrightarrow{\mu}} : K \subseteq [n], |K| = k, \overline{K} \subseteq K, |\overline{K}| \leq \overline{k}\}$ where $f_{K,\overline{K},\overrightarrow{\mu}}$ on input $(\mathsf{share}_1, \ldots, \mathsf{share}_n)$ outputs share_i for each $i \in \overline{K}$ in the clear and outputs $f_i(\mathsf{share}_i)$ for every $i \in K \setminus \overline{K}$ such that f_i is an arbitrary function outputting μ_i bits. When we just write μ (without the vector sign), we mean that every function f_i outputs at most μ bits.

Organization. The rest of this section is organized as follows: we first construct a k-out-of-k leakage resilient secret sharing scheme against $\mathcal{F}_{k,k-1,\mu}$ (in other words, $k-1$ shares are output in the clear and μ bits are leaked from the k-th share) in Sect. 4.1. In Sect. 4.2, we recall the definition of a combinatorial object called as *perfect hash function family* and give a randomized construction of such a family. Next, in Sect. 4.3, we combine the construction of k-out-of-k leakage resilient secret sharing scheme and a perfect hash function family to give a construction of k-out-of-n leakage resilient secret sharing scheme (for a constant k).

4.1 *k*-out-of-*k* Leakage Resilient Secret Sharing

In this subsection, we will construct a k-out-k leakage resilient secret sharing scheme against $\mathcal{F}_{k,k-1,\mu}$ for an arbitrary $k \geq 2$ (and not just for a constant k). As a building block, we will use a 2-out-of-2 leakage resilient secret sharing which was constructed in [GK18a]. We first recall the lemma regarding this construction.

Lemma 1 ([GK18a]). *For any $\epsilon > 0$ and $\mu, m \in \mathbb{N}$, there exists a construction of $(2, 2, 0, 0)$ secret sharing scheme for sharing m-bit secrets that is ϵ-leakage resilient against $\mathcal{F}_{2,1,\mu}$ such that the size of each share is $O(m + \mu + \log \frac{1}{\epsilon})$. The running time of the sharing and reconstruction procedures are $\mathrm{poly}(m, \mu, \log(1/\epsilon))$.*

Let us denote the secret sharing scheme guaranteed by Lemma 1 as $(\mathsf{LRShare}_{(2,2)}, \mathsf{LRRec}_{(2,2)})$. We will use this to construct a k-out-of-k leakage resilient secret sharing scheme for $k > 2$.

Lemma 2. *For any $\epsilon > 0$, $k \geq 2$ and $\mu, m \in \mathbb{N}$, there exists a construction of $(k, k, 0, 0)$ secret sharing scheme for sharing m-bit secrets that is ϵ-leakage resilient against $\mathcal{F}_{k,k-1,\mu}$ such that the size of each share is $O(m + \mu + \log \frac{1}{\epsilon})$. The running time of the sharing and the reconstruction procedures are $\mathrm{poly}(m, \mu, k, \log(1/\epsilon))$.*

We give the proof of this Lemma in the full version.

4.2 Perfect Hash Function Family

In this subsection, we recall the definition of the combinatorial objects called as *perfect hash function family* and give an efficient randomized construction for constant k.

Definition 9 (Perfect Hash Function Family [FK84]). *For every $n, k \in \mathbb{N}$, a set of hash functions $\{h_i\}_{i \in [\ell]}$ where $h_i : [n] \to [k]$ is said to be (n, k)-perfect hash function family if for each subset $S \subseteq [n]$ of size k there exists an $i \in [\ell]$ such that h_i is injective on S.*

Before we give the randomized construction, we will state and prove the following useful lemma.

Lemma 3. *For every $\epsilon > 0$, $n, k \in \mathbb{N}$, the set of functions $\{h_i\}_{i \in [\ell]}$ where each h_i is chosen randomly from the set of all functions mapping $[n] \to [k]$ is a perfectly hash function family with probability $1 - \epsilon$ when $\ell = \frac{\log \binom{n}{k} + \log \frac{1}{\epsilon}}{\log \frac{1}{1 - \frac{k!}{k^k}}}$. Specifically, when k is constant, we can set $\ell = O(\log n + \log \frac{1}{\epsilon})$.*

Proof. Let us first fix a subset $S \subseteq [n]$ of size k. Let us choose a function h uniformly at random from the set of all functions mapping $[n] \to [k]$.

$$\Pr[h \text{ is not injective over } S] = 1 - \frac{k!}{k^k}$$

Let us now choose $h_1, \ldots, h_\ell$ uniformly at random from the set of all functions mapping $[n] \to [k]$.

$$\Pr[\forall\ i \in [\ell],\ h_i \text{ is not injective over } S] = (1 - \frac{k!}{k^k})^\ell$$

By union bound,

$$\Pr[\exists\ S \text{ s.t.},\forall\ i \in [\ell],\ h_i \text{ is not injective over } S] = \binom{n}{k}(1 - \frac{k!}{k^k})^\ell$$

We want $\binom{n}{k}(1 - \frac{k!}{k^k})^\ell = \epsilon$. We get the bound for ℓ by rearranging this equation.

Randomized Construction for Constant k. For any k, n and some error parameter ϵ, set ℓ as in Lemma 3. Choose a function $h_i : [n] \to [k]$ uniformly at random for each $i \in [\ell]$. From Lemma 3, we infer that $\{h_i\}_{i\in[\ell]}$ is a perfect hash function family except with probability ϵ. The construction is efficient since the number of random bits needed for choosing each h_i is $n \log k$ which is polynomial in n when k is a constant.

Explicit Construction. Building on the work of Schmidt and Siegal [SS90], Alon et al. [AYZ95] gave an explicit construction of (n, k)-perfect hash function family of size $2^{O(k)} \log n$. We now recall the lemma from [AYZ95].

Lemma 4 ([AYZ95,SS90]). *For every $n, k \in \mathbb{N}$, there exists an explicit and efficiently computable construction of (n, k)-perfect hash function family $\{h_i\}_{i\in[\ell]}$ where $\ell = 2^{O(k)} \log n$.*

The explicit construction is obtained by brute forcing over a small bias probability space [NN93] and finding such a family is not as efficient as our randomized construction. On the positive side, the explicit construction is error-free unlike our randomized construction.

4.3 Construction of k-out-n Leakage Resilient Secret Sharing

In this subsection, we will use a k-out-of-k leakage resilient secret sharing scheme from Sect. 4.1 and a perfect hash function family from Sect. 4.2 to construct a k-out-of-n leakage resilient secret sharing scheme against $\mathcal{F}_{t,k-1,\overrightarrow{\mu}}$ for an arbitrary $t \leq n$ (recall the definition of $\mathcal{F}_{k,\overline{k},\overrightarrow{\mu}}$ from Definition 8). We give the description in Fig. 1.

Theorem 11. *For every $\epsilon_c, \epsilon_s > 0$, $n, k, m \in \mathbb{N}$ and $\overrightarrow{\mu} \in \mathbb{N}^n$, the construction given in Fig. 1 is a $(k, n, \epsilon_c, 0)$ secret sharing scheme for sharing m-bit secrets that is ϵ_s-leakage resilient against leakage functions $\mathcal{F}_{t,k-1,\overrightarrow{\mu}}$ for any $t \leq n$. The running times of the sharing and reconstruction algorithms are $\mathrm{poly}(n, m, \sum_i \mu_i, \log(1/\epsilon_c\epsilon_s))$ when k is a constant. In particular, when $\epsilon_s = \epsilon_c = 2^{-m}$, the running times are $\mathrm{poly}(n, m, \sum_i \mu_i)$. The size of each share when k is a constant is $O((m + \max_T \sum_{i\in T, T\subseteq[n], |T|=t} \mu_i + \log(\log n/\epsilon_s)) \log n)$.*

We give the proof of this theorem in the full version.

Remark 1. In Fig. 1, we cannot directly set the size $\ell = O(\log n + \log \frac{1}{\epsilon_c})$ and perform a single sampling to find a perfect hash function family. This is because when we want $\epsilon_c = 2^{-m}$, the size of the function family grows with m and this affects the rate significantly. That is why, it is important to set $\epsilon = 1/2$ and do $\log \frac{1}{\epsilon_c}$ independent repetitions in the $\mathsf{LRShare}_{(k,n)}$ function to reduce the error to ϵ_c.

Let $(\mathsf{LRShare}_{(k,k)}, \mathsf{LRRec}_{(k,k)})$ be a k-out-of-k leakage resilient secret sharing scheme.

$\mathsf{LRShare}_{(k,n)}$: To share a secret s:

1. For each $\mathsf{trial} \in [1, \log(1/\epsilon_c)]$ do:
   (a) Set $\epsilon = 1/2$ and $\ell = O(\log n)$. Sample a (candidate) (n,k)-perfect hash function family $\{h_i\}_{i\in[\ell]}$ as described in Section 4.2
   (b) Check if $\{h_i\}_{i\in[\ell]}$ is a family of (n,k)-perfect hash functions. That is, for each set $S \subset [n]$ and $|S| = k$, check if there exists an $i \in [\ell]$ such that h_i is injective on S.
   (c) If yes, exit the loop. Otherwise, go to the beginning.
2. If the above loop fails to find a perfect hash function family then abort.
3. For each $i \in [\ell]$, sample $\overline{\mathsf{share}}_{i,1}, \ldots, \overline{\mathsf{share}}_{i,k} \leftarrow \mathsf{LRShare}_{(k,k)}(s)$.
4. For each $j \in [n]$, set $\mathsf{share}_j = (h_1(j), \overline{\mathsf{share}}_{1,h_1(j)}) \circ (h_2(j), \overline{\mathsf{share}}_{2,h_2(j)}) \circ \ldots \circ (h_\ell(j), \overline{\mathsf{share}}_{\ell,h_\ell(j)})$.

$\mathsf{LRRec}_{(k,n)}$: Given the shares $\mathsf{share}_{j_1}, \mathsf{share}_{j_2}, \ldots, \mathsf{share}_{j_k}$ do:

1. Choose an $i \in [\ell]$, such that $\{h_i(j_1), h_i(j_2), \ldots, h_i(j_k)\} = \{1, \ldots, k\}$.
2. Recover s as $\mathsf{LRRec}_{(k,k)}(\overline{\mathsf{share}}_{i,1}, \ldots, \overline{\mathsf{share}}_{i,k})$.

Fig. 1. $(k, n, \epsilon_c, 0)$ Leakage resilient secret sharing scheme

5 Non-Malleable Secret Sharing for Threshold Access Structures

In this section, we give a construction of t-out-of-n (for any $t \geq 4$) Non-Malleable Secret Sharing scheme with rate $\Theta(\frac{1}{t \log^2 n})$ against tampering function family $\mathcal{F}_{\mathsf{ind}}$ that tampers each share independently. We first give the formal description of the tampering function family.

Individual Tampering Family $\mathcal{F}_{\mathsf{ind}}$. Let Share be the sharing function of the secret sharing scheme that outputs n-shares in $\mathcal{S}_1 \times \mathcal{S}_2 \ldots \times \mathcal{S}_n$. The function family $\mathcal{F}_{\mathsf{ind}}$ is composed of functions $(f_1, \ldots, f_n)$ where each $f_i : \mathcal{S}_i \to \mathcal{S}_i$.

5.1 Construction

Building Blocks. The construction uses the following building blocks. We instantiate them with concrete schemes later:

- A 3-split-state non-malleable code $(\mathsf{Enc}, \mathsf{Dec})$ where $\mathsf{Enc} : \mathcal{M} \to \mathcal{L} \times \mathcal{C} \times \mathcal{R}$ and the simulation error of the scheme is ϵ_1. Furthermore, we assume that for any two messages $m, m' \in \mathcal{M}$, $(\mathsf{C}, \mathsf{R}) \approx_{\epsilon_2} (\mathsf{C}', \mathsf{R}')$ where $(\mathsf{L}, \mathsf{C}, \mathsf{R}) \leftarrow \mathsf{Enc}(m)$ and $(\mathsf{L}', \mathsf{C}', \mathsf{R}') \leftarrow \mathsf{Enc}(m')$.
- A $(t, n, 0, 0)$ secret sharing scheme $(\mathsf{SecShare}_{(t,n)}, \mathsf{SecRec}_{(t,n)})$ with perfect privacy for message space $\mathcal{L}$. We will assume that the size of each share is m_1.

- A $(3, n, \epsilon_3', 0)$ secret sharing scheme $(\mathsf{LRShare}_{(3,n)}, \mathsf{LRRec}_{(3,n)})$ that is ϵ_3-leakage resilient against leakage functions $\mathcal{F}_{t,2,m_1}$[8] for message space $\mathcal{C}$. We assume that the size of each share is m_2.
- A $(2, n, \epsilon_4', 0)$ secret sharing scheme $(\mathsf{LRShare}_{(2,n)}, \mathsf{LRRec}_{(2,n)})$ for message space $\mathcal{R}$ that is ϵ_4-leakage resilient against leakage functions $\mathcal{F}_{t,1,\overrightarrow{\mu}}$ where $\max_T \sum_{i \in T, T \subseteq [n], |T|=t} \mu_i = O(m_2 + tm_1)$. We assume that the size of each share is m_3.

Construction. We give the formal description of the construction in Fig. 2 and give an informal overview below. To share a secret s, we first encode s to $(\mathsf{L}, \mathsf{C}, \mathsf{R})$ using the 3-split-state non-malleable code. We first encode L to $(\mathsf{SL}_1, \ldots, \mathsf{SL}_n)$ using the t-out-of-n threshold secret sharing scheme. We then encode C into $(\mathsf{SC}_1, \ldots, \mathsf{SC}_n)$ using the 3-out-of-n leakage resilience secret sharing scheme $\mathsf{LRShare}_{(3,n)}$. We finally encode R into $(\mathsf{SR}_1, \ldots, \mathsf{SR}_n)$ using the 2-out-of-n leakage resilient secret sharing scheme $\mathsf{LRShare}_{(2,n)}$. We set the i-th share share_i to be the concatenation of $\mathsf{SL}_i, \mathsf{SC}_i$ and SR_i. In order to reconstruct, we using the corresponding reconstruction procedures $\mathsf{SecRec}, \mathsf{LRRec}_{(3,n)}$ and $\mathsf{LRRec}_{(2,n)}$ to compute L, C and R respectively. We finally use the decoding procedure of 3-split-state non-malleable code to reconstruct the secret s from L, C and R.

Theorem 12. *For any arbitrary $n \in \mathbb{N}$ and threshold $t \geq 4$, the construction given in Fig. 2 is a $(t, n, \epsilon_3' + \epsilon_4', \epsilon_2)$ secret sharing scheme. Furthermore, it is $(\epsilon_1 + \epsilon_3 + \epsilon_4)$-non-malleable against $\mathcal{F}_{\mathsf{inf}}$.*

We give the proof of this theorem in the full version.

5.2 Rate Analysis

We now instantiate the primitives and provide the rate analysis.

1. We instantiate the three split state non-malleable code from the works of [KOS18, GMW17] (see Theorem 7). Using their construction, the $|\mathsf{L}| = |\mathsf{C}| = |\mathsf{R}| = O(m)$ bits and the error $\epsilon_1 = 2^{-\Omega(m/\log^{1+\rho}(m))}$ for any $\rho > 0$.
2. We use Shamir's secret sharing [Sha79] as the t-out-of-n secret sharing scheme. We get $m_1 = O(m)$ whenever $m > \log n$.
3. We instantiate $(\mathsf{LRShare}_{(3,n)}, \mathsf{LRRec}_{(3,n)})$ and $(\mathsf{LRShare}_{(2,n)}, \mathsf{LRRec}_{(2,n)})$ from Theorem 11. We get $m_2 = O(mt \log n)$ and $m_3 = O(mt \log^2 n)$ by setting ϵ_3 and ϵ_4 to be $2^{-\Omega(m/\log m)}$.

Thus the rate of our construction is $\Theta(\frac{1}{t \log^2 n})$ and the error is $2^{-\Omega(m/\log^{1+\rho}(m))}$.

We defer the concrete optimization of the rate of our construction to the full version of the paper.

[8] Recall that this denotes that the function can choose to leak at most m_1 bits from each share in a set of size $t-2$ apart from the two that are completely leaked.

$\mathsf{Share}(m)$: To share a secret $s \in \mathcal{M}$ do:

1. Encode the secret s as $(\mathsf{L}, \mathsf{C}, \mathsf{R}) \leftarrow \mathsf{Enc}(s)$.
2. Compute the shares

$$(\mathsf{SL}_1, \ldots, \mathsf{SL}_n) \leftarrow \mathsf{SecShare}_{(t,n)}(\mathsf{L})$$

$$(\mathsf{SC}_1, \ldots, \mathsf{SC}_n) \leftarrow \mathsf{LRShare}_{(3,n)}(\mathsf{C})$$

$$(\mathsf{SR}_1, \ldots, \mathsf{SR}_n) \leftarrow \mathsf{LRShare}_{(2,n)}(\mathsf{R})$$

3. For each $i \in [n]$, set share_i as $(\mathsf{SL}_i, \mathsf{SC}_i, \mathsf{SR}_i)$ and output $(\mathsf{share}_1, \ldots, \mathsf{share}_n)$ as the shares.

$\mathsf{Rec}(\mathsf{Share}(m)_T)$: To reconstruct the secret from the shares in an authorized set T of size t do:

1. Let the shares corresponding to the set T be $(\mathsf{share}_{i_1}, \ldots, \mathsf{share}_{i_t})$.
2. For each $j \in \{i_1, \ldots, i_t\}$, parse share_j as $(\mathsf{SL}_j, \mathsf{SC}_j, \mathsf{SR}_j)$.
3. Reconstruct

$$\mathsf{L} := \mathsf{SecRec}_{(t,n)}(\mathsf{SL}_{i_1}, \ldots, \mathsf{SL}_{i_t})$$

$$\mathsf{C} := \mathsf{LRRec}_{(3,n)}(\mathsf{SC}_{i_1}, \mathsf{SC}_{i_2}, \mathsf{SC}_{i_3})$$

$$\mathsf{R} := \mathsf{LRRec}_{(2,n)}(\mathsf{SR}_{i_1}, \mathsf{SR}_{i_2})$$

4. Output the secret s as $\mathsf{Dec}(\mathsf{L}, \mathsf{C}, \mathsf{R})$.

Fig. 2. Construction of t-out-of-n non-malleable secret sharing scheme

6 NMSS for General Access Structures with Multiple Tampering

We first define non-malleable secret sharing for general access structures in the next subsection and then give the construction in the subsequent subsection.

6.1 Definitions

First, we recall the definition of a secret sharing scheme for a general monotone access structure $\mathcal{A}$ - a generalization of the one defined for threshold access structures in Definition 4.

Definition 10 ($(\mathcal{A}, n, \epsilon_c, \epsilon_s)$-Secret Sharing Scheme). *Let $\mathcal{M}$ be a finite set of secrets, where $|\mathcal{M}| \geq 2$. Let $[n] = \{1, 2, \ldots, n\}$ be a set of identities (indices) of n parties. A sharing function* Share *with domain of secrets $\mathcal{M}$ is a $(\mathcal{A}, n, \epsilon_c, \epsilon_s)$-secret sharing scheme with respect to monotone access structure $\mathcal{A}$ if the following two properties hold:*

- ***Correctness:*** *The secret can be reconstructed by any set of parties that are part of the access structure $\mathcal{A}$. That is, for any set $T \in \mathcal{A}$, there exists a deterministic reconstruction function $\mathsf{Rec} : \otimes_{i \in T} \mathcal{S}_i \rightarrow \mathcal{M}$ such that for every $m \in \mathcal{M}$,*

$$\Pr[\mathsf{Rec}(\mathsf{Share}(m)_T) = m] = 1 - \epsilon_c$$

where the probability is over the randomness of the Share *function. We will slightly abuse the notation and denote* Rec *as the reconstruction procedure that takes in* $T \in \mathcal{A}$ *and* $\mathsf{Share}(m)_T$ *as input and outputs the secret.*

– ***Statistical Privacy:*** *Any collusion of parties not part of the access structure should have "almost" no information about the underlying secret. More formally, for any unauthorized set* $U \subseteq [n]$ *such that* $U \notin \mathcal{A}$*, and for every pair of secrets* $m_0, m_1 \in M$*, for any distinguisher* D *with output in* $\{0,1\}$*, the following holds:*

$$|\Pr[D(\mathsf{Share}(m_0)_U) = 1] - \Pr[D(\mathsf{Share}(m_1)_U) = 1]| \leq \epsilon_s$$

We define the rate of the secret sharing scheme as $\frac{|m|}{\max_{i\in[n]} |\mathsf{Share}(m)_i|}$

We now define the notion of a non-malleable secret sharing scheme for general access structures which is a generalization of the definition for threshold access structures given in Definition 5.

Definition 11 (Non-Malleable Secret Sharing for General Access Structures [GK18b]). *Let* $(\mathsf{Share}, \mathsf{Rec})$ *be a* $(\mathcal{A}, n, \epsilon_c, \epsilon_s)$*-secret sharing scheme for message space* $\mathcal{M}$ *and access structure* $\mathcal{A}$*. Let* $\mathcal{F}$ *be a family of tampering functions. For each* $f \in \mathcal{F}$*,* $m \in \mathcal{M}$ *and authorized set* $T \in \mathcal{A}$*, define the tampered distribution* $\mathsf{Tamper}_m^{f,T}$ *as* $\mathsf{Rec}(f(\mathsf{Share}(m))_T)$ *where the randomness is over the sharing function* Share*. We say that the* $(\mathcal{A}, n, \epsilon_c, \epsilon_s)$*-secret sharing scheme,* $(\mathsf{Share}, \mathsf{Rec})$ *is* ϵ'*-non-malleable w.r.t.* $\mathcal{F}$ *if for each* $f \in \mathcal{F}$ *and any authorized set* $T \in \mathcal{A}$*, there exists a distribution* $D^{f,T}$ *over* $\mathcal{M} \cup \{\mathsf{same}^\star\}$ *such that:*

$$|\mathsf{Tamper}_m^{f,T} - \mathrm{copy}(D^{f,T}, m)| \leq \epsilon'$$

where copy *is defined by* $\mathrm{copy}(x, y) = \begin{cases} x & \text{if } x \neq \mathsf{same}^\star \\ y & \text{if } x = \mathsf{same}^\star \end{cases}$.

Many Tampering Extension. Similar to the threshold case, in the full version, we extend the above definition to capture multiple tampering attacks.

6.2 Construction

In this section, we show how to build a one-many non-malleable secret sharing scheme for general access structures.

First, let $(\mathsf{SecShare}_{(\mathcal{A},n)}, \mathsf{SecRec}_{(\mathcal{A},n)})$ be any statistically private secret sharing scheme with rate R for a 4-monotone access structure $\mathcal{A}$ over n parties. We refer the reader to [KW93, LV18] for explicit constructions.

Let $\mathsf{t}_{\mathsf{max}}$ denote the maximum size of a minimal authorized set of $\mathcal{A}$.[9] We give a construction of a Non-Malleable Secret Sharing scheme with tampering degree K for a 4-monotone access structure $\mathcal{A}$ with rate $O(\frac{\mathsf{R}}{\mathsf{K}^3 \mathsf{t}_{\mathsf{max}} \log^2 n})$ with respect to a individual tampering function family $\mathcal{F}_{ind}$.

[9] We refer the reader to Definition 1, Definition 2 for definitions of 4-monotone access structures and minimal authorized set.

Building Blocks. The construction uses the following building blocks. We instantiate them with concrete schemes later:

- A one-many 3-split-state non-malleable code $(\mathsf{Enc}, \mathsf{Dec})$ where $\mathsf{Enc} : \mathcal{M} \to \mathcal{L} \times \mathcal{C} \times \mathcal{R}$, the simulation error of the scheme is ϵ_1 and the scheme is secure against K tamperings. Furthermore, we assume that for any two messages $m, m' \in \mathcal{M}$, $(\mathsf{C}, \mathsf{R}) \approx_{\epsilon_2} (\mathsf{C}', \mathsf{R}')$ where $(\mathsf{L}, \mathsf{C}, \mathsf{R}) \leftarrow \mathsf{Enc}(m)$ and $(\mathsf{L}', \mathsf{C}', \mathsf{R}') \leftarrow \mathsf{Enc}(m')$.
- A $(\mathcal{A}, n, 0, 0)$ (where $\mathcal{A}$ is 4-monotone) secret sharing scheme $(\mathsf{SecShare}_{(\mathcal{A},n)}, \mathsf{SecRec}_{(\mathcal{A},n)})$ with perfect privacy for message space $\mathcal{L}$.[10] We will assume that the size of each share is m_1.
- A $(3, n, \epsilon_3', 0)$ secret sharing scheme $(\mathsf{LRShare}_{(3,n)}, \mathsf{LRRec}_{(3,n)})$ that is ϵ_3-leakage resilient against leakage functions $\mathcal{F}_{\mathsf{t_{max}},2,\mathsf{K}m_1}$ for message space $\mathcal{C}$. We assume that the size of each share is m_2.
- A $(2, n, \epsilon_4', 0)$ secret sharing scheme $(\mathsf{LRShare}_{(2,n)}, \mathsf{LRRec}_{(2,n)})$ for message space $\mathcal{R}$ that is ϵ_4-leakage resilient against leakage functions $\mathcal{F}_{\mathsf{t_{max}},1,\overrightarrow{\mu}}$ where $\max_T \sum_{i \in T, T \in \mathcal{A}, |T| = \mathsf{t_{max}}} \mu_i = O(\mathsf{K}m_2 + \mathsf{Kt_{max}}m_1)$. We assume that the size of each share is m_3.

$\mathsf{Share}(m)$: To share a secret $s \in \mathcal{M}$ do:

1. Encode the secret s as $(\mathsf{L}, \mathsf{C}, \mathsf{R}) \leftarrow \mathsf{Enc}(s)$.
2. Compute the shares

$$(\mathsf{SL}_1, \ldots, \mathsf{SL}_n) \leftarrow \mathsf{SecShare}_{(\mathcal{A},n)}(\mathsf{L})$$

$$(\mathsf{SC}_1, \ldots, \mathsf{SC}_n) \leftarrow \mathsf{LRShare}_{(3,n)}(\mathsf{C})$$

$$(\mathsf{SR}_1, \ldots, \mathsf{SR}_n) \leftarrow \mathsf{LRShare}_{(2,n)}(\mathsf{R})$$

3. For each $i \in [n]$, set share_i as $(\mathsf{SL}_i, \mathsf{SC}_i, \mathsf{SR}_i)$ and output $(\mathsf{share}_1, \ldots, \mathsf{share}_n)$ as the set of shares.

$\mathsf{Rec}(\mathsf{Share}(m)_T)$: Given a set of shares in an authorized set $T' \in \mathcal{A}$, let $T \subseteq T'$ denote a minimal authorized set. To reconstruct the secret from the shares in set T, (of size at most $\mathsf{t_{max}}$) do:

1. Let the shares corresponding to the set T be $(\mathsf{share}_{i_1}, \ldots, \mathsf{share}_{i_{\mathsf{t_{max}}}})$.
2. For each $j \in \{i_1, \ldots, i_{\mathsf{t_{max}}}\}$, parse share_j as $(\mathsf{SL}_j, \mathsf{SC}_j, \mathsf{SR}_j)$.
3. Reconstruct

$$\mathsf{L} := \mathsf{SecRec}_{(\mathcal{A},n)}(\mathsf{SL}_{i_1}, \ldots, \mathsf{SL}_{i_{\mathsf{t_{max}}}})$$

$$\mathsf{C} := \mathsf{LRRec}_{(3,n)}(\mathsf{SC}_{i_1}, \mathsf{SC}_{i_2}, \mathsf{SC}_{i_3})$$

$$\mathsf{R} := \mathsf{LRRec}_{(2,n)}(\mathsf{SR}_{i_1}, \mathsf{SR}_{i_2})$$

4. Output the secret s as $\mathsf{Dec}(\mathsf{L}, \mathsf{C}, \mathsf{R})$.

Fig. 3. Construction of non-malleable secret sharing scheme for general access structures against multiple tampering

[10] We note that our proof of security goes through even if this secret sharing scheme only has statistical privacy.

Construction. The construction is very similar to the construction of non-malleable secret sharing for threshold access structures given in Sect. 5 with the only difference being that we now use the $(\mathcal{A}, n, 0, 0)$ secret sharing scheme. Note that in the construction we additionally need a procedure to find a minimal authorized set from any authorized set. This procedure is efficient if we can efficiently test the membership in $\mathcal{A}$. We point the reader to [GK18b] for details of this procedure. We give the formal description of the construction in Fig. 3 for completeness.

Theorem 13. *There exists a constant $\gamma > 0$ such that, for any arbitrary $n, \mathsf{K} \in \mathbb{N}$ and 4-monotone access structure $\mathcal{A}$, the construction given in Fig. 3 is a $(\mathcal{A}, n, \epsilon_3' + \epsilon_4', \epsilon_2)$ secret sharing scheme for messages of length m where $m \geq \mathsf{K}^{\gamma}$. Furthermore, it is $(\epsilon_1 + \epsilon_3 + \epsilon_4)$ one-many non-malleable with tampering degree K with respect to tampering function family $\mathcal{F}_{ind}$.*

We give the proof of this theorem and the rate analysis in the full version.

Acknowledgements. The first author's research supported in part by the IBM PhD Fellowship. The first author's research also supported in part from a DARPA /ARL SAFEWARE award, NSF Frontier Award 1413955, and NSF grant1619348, BSF grant 2012378, a Xerox Faculty Research Award, a Google Faculty Research Award, an equipment grant from Intel, an Okawa Foundation Research Grant, NSF-BSF grant 1619348, DARPA SafeWare subcontract to Galois Inc., DARPA SPAWAR contract N66001-15-1C-4065, US-Israel BSF grant 2012366, OKAWA Foundation Research Award, IBM Faculty Research Award, Xerox Faculty Research Award, B. John Garrick Foundation Award, Teradata Research Award, and Lockheed-Martin Corporation Research Award. This material is based upon work supported by the Defense Advanced Research Projects Agency through the ARL under Contract W911NF-15-C- 0205. The second author's research supported in part from DARPA/ARL SAFEWARE Award W911NF15C0210, AFOSR Award FA9550-15-1-0274, AFOSR YIP Award, DARPA and SPAWAR under contract N66001-15-C-4065, a Hellman Award and research grants by the Okawa Foundation, Visa Inc., and Center for Long-Term Cybersecurity (CLTC, UC Berkeley) of Sanjam Garg. The views expressed are those of the authors and do not reflect the official policy or position of the funding agencies.

The authors thank Pasin Manurangsi for pointing to the work of Alon et al. [AYZ95] for the explicit construction of perfect hash function family. The authors also thank Sanjam Garg, Peihan Miao and Prashant Vasudevan for useful comments on the write-up.

References

[AAG+16] Aggarwal, D., Agrawal, S., Gupta, D., Maji, H.K., Pandey, O., Prabhakaran, M.: Optimal computational split-state non-malleable codes. In: Kushilevitz, E., Malkin, T. (eds.) TCC 2016. LNCS, vol. 9563, pp. 393–417. Springer, Heidelberg (2016). https://doi.org/10.1007/978-3-662-49099-0_15

[ADKO15] Aggarwal, D., Dodis, Y., Kazana, T., Obremski, M.: Non-malleable reductions and applications. In: STOC, pp. 459–468 (2015)

[ADL14] Aggarwal, D., Dodis, Y., Lovett, S.: Non-malleable codes from additive combinatorics. In: STOC, pp. 774–783 (2014)

[ADN+18] Aggarwal, D., et al.: Stronger leakage-resilient and non-malleable secret-sharing schemes for general access structures. Cryptology ePrint Archive, Report 2018/1147 (2018). https://eprint.iacr.org/2018/1147

[AGM+15] Agrawal, S., Gupta, D., Maji, H.K., Pandey, O., Prabhakaran, M.: Explicit non-malleable codes against bit-wise tampering and permutations. In: Gennaro, R., Robshaw, M. (eds.) CRYPTO 2015. LNCS, vol. 9215, pp. 538–557. Springer, Heidelberg (2015). https://doi.org/10.1007/978-3-662-47989-6_26

[AYZ95] Alon, N., Yuster, R., Zwick, U.: Color-coding. J. ACM **42**(4), 844–856 (1995)

[BDG+18] Ball, M., Dachman-Soled, D., Guo, S., Malkin, T., Tan, L.-Y.: Non-malleable codes for small-depth circuits. In: FOCS (2018, to appear)

[BDIR18] Benhamouda, F., Degwekar, A., Ishai, Y., Rabin, T.: On the local leakage resilience of linear secret sharing schemes. In: Shacham, H., Boldyreva, A. (eds.) CRYPTO 2018, Part I. LNCS, vol. 10991, pp. 531–561. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-96884-1_18

[BDKM16] Ball, M., Dachman-Soled, D., Kulkarni, M., Malkin, T.: Non-malleable codes for bounded depth, bounded fan-in circuits. In: Fischlin, M., Coron, J.-S. (eds.) EUROCRYPT 2016. LNCS, vol. 9666, pp. 881–908. Springer, Heidelberg (2016). https://doi.org/10.1007/978-3-662-49896-5_31

[BDKM18] Ball, M., Dachman-Soled, D., Kulkarni, M., Malkin, T.: Non-malleable codes from average-case hardness: AC^0, decision trees, and streaming space-bounded tampering. In: Nielsen, J.B., Rijmen, V. (eds.) EUROCRYPT 2018. LNCS, vol. 10822, pp. 618–650. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-78372-7_20

[BDL01] Boneh, D., DeMillo, R.A., Lipton, R.J.: On the importance of eliminating errors in cryptographic computations. J. Cryptol. **14**(2), 101–119 (2001)

[BGW88] Ben-Or, M., Goldwasser, S., Wigderson, A.: Completeness theorems for non-cryptographic fault-tolerant distributed computation (extended abstract). In: STOC, pp. 1–10 (1988)

[Bla79] Blakley, G.R.: Safeguarding cryptographic keys. In: Proceedings of AFIPS 1979 National Computer Conference, vol. 48, pp. 313–317 (1979)

[Bla99] Blackburn, S.R.: Combinatorics and Threshold Cryptography. Chapman and Hall CRC Research Notes in Mathematics, pp. 49–70 (1999)

[CCD88] Chaum, D., Crepeau, C., Damgaard, I.: Multiparty unconditionally secure protocols (extended abstract). In: STOC, pp. 11–19. ACM (1988)

[CDF+08] Cramer, R., Dodis, Y., Fehr, S., Padró, C., Wichs, D.: Detection of algebraic manipulation with applications to robust secret sharing and fuzzy extractors. In: Smart, N. (ed.) EUROCRYPT 2008. LNCS, vol. 4965, pp. 471–488. Springer, Heidelberg (2008). https://doi.org/10.1007/978-3-540-78967-3_27

[CG88] Chor, B., Goldreich, O.: Unbiased bits from sources of weak randomness and probabilistic communication complexity. SIAM J. Comput. **17**(2), 230–261 (1988)

[CGL16] Chattopadhyay, E., Goyal, V., Li, X.: Non-malleable extractors and codes, with their many tampered extensions. In: Proceedings of the 48th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2016, Cambridge, MA, USA, 18–21 June 2016, pp. 285–298 (2016)

[CGM+16] Chandran, N., Goyal, V., Mukherjee, P., Pandey, O., Upadhyay, J.: Block-wise non-malleable codes. In: ICALP (2016)

[CGMA85] Chor, B., Goldwasser, S., Micali, S., Awerbuch, B.: Verifiable secret sharing and achieving simultaneity in the presence of faults (extended abstract). In: 26th Annual Symposium on Foundations of Computer Science, pp. 383–395. IEEE Computer Society Press, October 1985

[CKR16] Chandran, N., Kanukurthi, B., Raghuraman, S.: Information-theoretic local non-malleable codes and their applications. In: Kushilevitz, E., Malkin, T. (eds.) TCC 2016, Part II. LNCS, vol. 9563, pp. 367–392. Springer, Heidelberg (2016). https://doi.org/10.1007/978-3-662-49099-0_14

[CL17] Chattopadhyay, E., Li, X.: Non-malleable codes and extractors for small-depth circuits, and affine functions. In: Hatami, H., McKenzie, P., King, V. (eds.) 49th Annual ACM Symposium on Theory of Computing, pp. 1171–1184. ACM Press, June 2017

[DDFY94] De Santis, A., Desmedt, Y., Frankel, Y., Yung, M.: How to share a function securely. In: 26th Annual ACM Symposium on Theory of Computing, pp. 522–533. ACM Press, May 1994

[DF90] Desmedt, Y., Frankel, Y.: Threshold cryptosystems. In: Brassard, G. (ed.) CRYPTO 1989. LNCS, vol. 435, pp. 307–315. Springer, New York (1990). https://doi.org/10.1007/0-387-34805-0_28

[DKO13] Dziembowski, S., Kazana, T., Obremski, M.: Non-malleable codes from two-source extractors. In: Canetti, R., Garay, J.A. (eds.) CRYPTO 2013, Part II. LNCS, vol. 8043, pp. 239–257. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-40084-1_14

[DPW10] Dziembowski, S., Pietrzak, K., Wichs, D.: Non-malleable codes. In: Proceedings of Innovations in Computer Science - ICS 2010, Tsinghua University, Beijing, China, 5–7 January 2010, pp. 434–452 (2010)

[FK84] Fredman, M.L., Komlós, J.: On the size of separating systems and families of perfect hash functions. SIAM J. Algebraic Discrete Methods **5**(1), 61–68 (1984)

[FMNV14] Faust, S., Mukherjee, P., Nielsen, J.B., Venturi, D.: Continuous non-malleable codes. In: Lindell, Y. (ed.) TCC 2014. LNCS, vol. 8349, pp. 465–488. Springer, Heidelberg (2014). https://doi.org/10.1007/978-3-642-54242-8_20

[FMNV15] Faust, S., Mukherjee, P., Nielsen, J.B., Venturi, D.: A tamper and leakage resilient von neumann architecture. In: Katz, J. (ed.) PKC 2015. LNCS, vol. 9020, pp. 579–603. Springer, Heidelberg (2015). https://doi.org/10.1007/978-3-662-46447-2_26

[FMVW14] Faust, S., Mukherjee, P., Venturi, D., Wichs, D.: Efficient non-malleable codes and key-derivation for poly-size tampering circuits. In: Nguyen, P.Q., Oswald, E. (eds.) EUROCRYPT 2014. LNCS, vol. 8441, pp. 111–128. Springer, Heidelberg (2014). https://doi.org/10.1007/978-3-642-55220-5_7

[Fra90] Frankel, Y.: A practical protocol for large group oriented networks. In: Quisquater, J.-J., Vandewalle, J. (eds.) EUROCRYPT 1989. LNCS, vol. 434, pp. 56–61. Springer, Heidelberg (1990). https://doi.org/10.1007/3-540-46885-4_8

[FRR+10] Faust, S., Rabin, T., Reyzin, L., Tromer, E., Vaikuntanathan, V.: Protecting circuits from leakage: the computationally-bounded and noisy cases. In: Gilbert, H. (ed.) EUROCRYPT 2010. LNCS, vol. 6110, pp. 135–156. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-13190-5_7

[GK18a] Goyal, V., Kumar, A.: Non-malleable secret sharing. In: STOC, pp. 685–698 (2018)

[GK18b] Goyal, V., Kumar, A.: Non-malleable secret sharing for general access structures. In: Shacham, H., Boldyreva, A. (eds.) CRYPTO 2018, Part I. LNCS, vol. 10991, pp. 501–530. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-96884-1_17

[GKP+18] Goyal, V., Kumar, A., Park, S., Richelson, S., Srinivasan, A.: Non-malleable commitments from non-malleable extractors. Manuscript, accessed via personal communication (2018)

[GLM+04] Gennaro, R., Lysyanskaya, A., Malkin, T., Micali, S., Rabin, T.: Algorithmic tamper-proof (ATP) security: theoretical foundations for security against hardware tampering. In: Naor, M. (ed.) TCC 2004. LNCS, vol. 2951, pp. 258–277. Springer, Heidelberg (2004). https://doi.org/10.1007/978-3-540-24638-1_15

[GMW87] Goldreich, O., Micali, S., Wigderson, A.: How to play any mental game or A completeness theorem for protocols with honest majority. In: Aho, A. (ed.) 19th Annual ACM Symposium on Theory of Computing, pp. 218–229. ACM Press, May 1987

[GMW17] Gupta, D., Maji, H.K., Wang, M.: Constant-rate non-malleable codes in the split-state model. Cryptology ePrint Archive, Report 2017/1048 (2017). https://eprint.iacr.org/2017/1048

[ISW03] Ishai, Y., Sahai, A., Wagner, D.: Private circuits: securing hardware against probing attacks. In: Boneh, D. (ed.) CRYPTO 2003. LNCS, vol. 2729, pp. 463–481. Springer, Heidelberg (2003). https://doi.org/10.1007/978-3-540-45146-4_27

[JW15] Jafargholi, Z., Wichs, D.: Tamper detection and continuous non-malleable codes. In: Dodis, Y., Nielsen, J.B. (eds.) TCC 2015, Part I. LNCS, vol. 9014, pp. 451–480. Springer, Heidelberg (2015). https://doi.org/10.1007/978-3-662-46494-6_19

[KLT18] Kiayias, A., Liu, F.-H., Tselekounis, Y.: Non-malleable codes for partial functions with manipulation detection. In: Shacham, H., Boldyreva, A. (eds.) CRYPTO 2018, Part III. LNCS, vol. 10993, pp. 577–607. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-96878-0_20

[KMS18] Kumar, A., Meka, R., Sahai, A.: Leakage-resilient secret sharing. Cryptology ePrint Archive, Report 2018/1138 (2018). https://eprint.iacr.org/2018/1138

[KOS17] Kanukurthi, B., Obbattu, S.L.B., Sekar, S.: Four-state non-malleable codes with explicit constant rate. In: Kalai, Y., Reyzin, L. (eds.) TCC 2017, Part II. LNCS, vol. 10678, pp. 344–375. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-70503-3_11

[KOS18] Kanukurthi, B., Obbattu, S.L.B., Sekar, S.: Non-malleable randomness encoders and their applications. In: Nielsen, J.B., Rijmen, V. (eds.) EUROCRYPT 2018, Part III. LNCS, vol. 10822, pp. 589–617. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-78372-7_19

[KW93] Karchmer, M., Wigderson, A.: On span programs. In: Proceedings of the Eigth Annual Structure in Complexity Theory Conference, San Diego, CA, USA, 18–21 May 1993, pp. 102–111 (1993)

[Li17] Li, X.: Improved non-malleable extractors, non-malleable codes and independent source extractors. In: STOC (2017)

[LL12] Liu, F.-H., Lysyanskaya, A.: Tamper and leakage resilience in the split-state model. In: Safavi-Naini, R., Canetti, R. (eds.) CRYPTO 2012. LNCS, vol. 7417, pp. 517–532. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-32009-5_30

[LV18] Liu, T., Vaikuntanathan, V.: Breaking the circuit-size barrier in secret sharing. In: Diakonikolas, I., Kempe, D., Henzinger, M. (eds.) 50th Annual ACM Symposium on Theory of Computing, pp. 699–708. ACM Press, June 2018

[NN93] Naor, J., Naor, M.: Small-bias probability spaces: efficient constructions and applications. SIAM J. Comput. **22**(4), 838–856 (1993)

[OPVV18] Ostrovsky, R., Persiano, G., Venturi, D., Visconti, I.: Continuously non-malleable codes in the split-state model from minimal assumptions. In: Shacham, H., Boldyreva, A. (eds.) CRYPTO 2018, Part III. LNCS, vol. 10993, pp. 608–639. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-96878-0_21

[Rot12] Rothblum, G.N.: How to compute under $\mathcal{AC}^0$ leakage without secure hardware. In: Safavi-Naini, R., Canetti, R. (eds.) CRYPTO 2012. LNCS, vol. 7417, pp. 552–569. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-32009-5_32

[Sha79] Shamir, A.: How to share a secret. Commun. ACM **22**(11), 612–613 (1979)

[SNW01] Safavi-Naini, R., Wang, H.: Robust additive secret sharing schemes over ZM. In: Lam, K.-Y., Shparlinski, I., Wang, H., Xing, C. (eds.) Cryptography and Computational Number Theory, pp. 357–368. Birkhäuser, Basel (2001)

[SS90] Schmidt, J.P., Siegel, A.: The spatial complexity of oblivious k-probe hash functions. SIAM J. Comput. **19**(5), 775–786 (1990)