Short Leakage Resilient and Non-malleable Secret Sharing Schemes

Nishanth Chandran[*] Bhavana Kanukurthi [†]
Sai Lakshmi Bhavana Obbattu [‡] Sruthi Sekar[§]

February 21, 2022

Abstract

Leakage resilient secret sharing (LRSS) allows a dealer to share a secret amongst n parties such that any authorized subset of the parties can recover the secret from their shares, while an adversary that obtains shares of any unauthorized subset of parties along with bounded leakage from the other shares learns no information about the secret. Non-malleable secret sharing (NMSS) provides a guarantee that even shares that are tampered by an adversary will reconstruct to either the original message or something independent of it.

The most important parameter of LRSS and NMSS schemes is the size of each share. For LRSS, in the *local leakage model* (i.e., when the leakage functions on each share are independent of each other and bounded), Srinivasan and Vasudevan (CRYPTO 2019), gave a scheme for threshold access structures with share size of approximately ($3 \cdot \mathsf{message\ length} + \mu$), where μ is the number of bits of leakage tolerated from every share. For the case of NMSS, the best known result (again due to the above work) has share size of ($11 \cdot \mathsf{message\ length}$).

In this work, we build LRSS and NMSS schemes with much improved share size. Additionally, our LRSS scheme obtains optimal share and leakage size. In particular, we get the following results:

- We build an information-theoretic LRSS scheme for threshold access structures with a share size of ($\mathsf{message\ length} + \mu$).
- As an application of the above result, we obtain an NMSS with a share size of ($4 \cdot \mathsf{message\ length}$). Further, for the special case of sharing random messages, we obtain a share size of ($2 \cdot \mathsf{message\ length}$).

[*] Microsoft Research, India, Email: nichandr@microsoft.com.

[†] Department of Computer Science and Automation, Indian Institute of Science, Email: bhavana@iisc.ac.in. Research supported by Microsoft Research, India.

[‡] Microsoft Research, India, Email: oslbhavana@gmail.com.

[§] UC Berkeley, California, Email: sruthi.sekar1@gmail.com. This work was done while at Indian Institute of Science, Bangalore.

Contents

1 Introduction

Secret sharing schemes [Sha79, Bla79] are fundamental cryptographic primitives that allow a dealer to distribute its secret m amongst n parties in such a way that any authorized subset of parties can recover m from their shares (*correctness*), and no unauthorized subset of parties get any information about m (*privacy*). For instance, in a t-out-of-n ($t \leq n$) threshold secret sharing scheme, any subset of t parties or more is an authorized set and can reconstruct the secret, while any subset of fewer than t parties is unauthorized. Secret sharing schemes have found several applications in literature, such as in multi-party computation [GMW87, BGW88, CCD88], leakage-resilient circuit compilers [ISW03, FRR$^+$10, Rot12] and threshold cryptographic systems [DF89, Fra89, SDFY94]. An assumption that secret sharing schemes make is that the adversary, controlling an unauthorized set of parties, gets no information about the shares of the honest parties. However, a rich study on side-channel attacks called *leakage attacks* points to the fact that such an assumption is idealized and may not hold in practice. This has led to much work on *leakage resilient* cryptography [Koc96, BBR88, BBCM95, Riv97, DSS01, CDH$^+$00, MR04, DP07, AGV09]. In the context of secret sharing, leakage attacks allow the adversary to additionally obtain some bounded leakage from the honest party shares, and such a leakage may help an adversary break privacy of the secret sharing[1]. To secure against such attacks, Dziembowski and Pietrzak [DP07] introduced the notion of leakage resilient secret sharing.

Leakage Resilient/Non-malleable Secret Sharing (LRSS/NMSS). Informally, an LRSS gives a guarantee that the adversary gets no information about the secret, given its shares from an unauthorized set, as well as leakage from the remaining honest shares. In particular, in the *local leakage model* [BDIR18, GK18, SV19, ADN$^+$19], the adversary is allowed to make a non-adaptive query to obtain a complete unauthorized set of shares, along with independent (bounded) leakage on the remaining shares. Privacy is then required to hold against such an adversary. LRSS schemes tolerating local leakage have been shown to have applications to leakage-resilient MPC [BGK11, GIM$^+$16, BDIR18, SV19], leakage-resilient non-malleable secret sharing [GK18, SV19, BS19], and more recently to zero knowledge PCPs [HVW21]. Non-malleable secret sharing (NMSS) was introduced by Goyal and Kumar in [GK18] and provides a guarantee that under a tampering attack by the adversary, the message recovered from the tampered shares will either be the same as the original message or will be independent of it.

Share Size of LRSS/NMSS schemes. The most important aspect of secret sharing schemes is their share size, which typically determines the efficiency of the application that relies on it. For example, in an application to MPC, the size of each share affects the overall communication complexity of the MPC protocol. For standard threshold secret sharing schemes, we know constructions [Sha79, LCG$^+$19] with optimal share size (i.e., same as the message length). However, the picture is quite different in the presence of leakage and/or tampering. For a very special case[2], Benhamouda, Degwekar, Ishai and Rabin [BDIR18] show that the Shamir secret sharing [Sha79] is leakage resilient with share size of approximately ($\mathsf{message\ length} + 4 \cdot \mu$), where μ is the number of

[1] In fact, Guruswami and Wooters [GW16] show that Shamir's secret sharing scheme over a field of characteristic 2 is completely insecure when the adversary gets some $t-1$ shares and just one-bit of leakage from other shares. Further, Nielsen and Simkin [NS20] show that for larger characteristic fields and large n, Shamir's secret sharing scheme is not leakage resilient for threshold $t \leq cn/\log n$, for constant $0 < c < 1$.

[2] where the underlying field is a large characteristic field, the number of parties n is large, the threshold t is at least $n - o(\log n)$, and the adversary can only obtain a constant number of full shares

bits of leakage from each share[3]. The work of Aggarwal, Damgard, Nielsen, Obremski, Purwanto, Ribeiro and Simkin [ADN$^+$19] constructs – as a stepping stone towards constructing an NMSS – an LRSS scheme; however, this scheme suffers a polynomial (in n) blowup in its share size and additionally obtains optimal leakage ($\approx ((1 - o(1))\mathsf{message\ length})$) only for the restricted case of constant number of parties. For the general case of arbitrary n and t, Srinivasan and Vasudevan [SV19] constructed an LRSS against the local leakage model with a share size of approximately $(3 \cdot \mathsf{message\ length}\ + \mu)$, to tolerate μ bits of leakage from each share. Most constructions of NMSS schemes implicitly require an LRSS and hence share size of LRSS schemes directly impact that of NMSS. The best known share size for an NMSS is $(11 \cdot \mathsf{message\ length})$, achieved by the construction of [SV19] (through the [BS19] compiler).

We remark here that obtaining LRSS/NMSS with short share size while simultaneously tolerating high leakage rate is an important problem noted in several works (e.g.: In [ADN$^+$19], the authors state: "*It would be interesting to give constructions of leakage-resilient schemes (even in the non-adaptive setting) with an improved tradeoff between leakage rate and share length* ").

1.1 Our Results

In this work, we construct the first information-theoretic LRSS scheme for threshold access structures against the local leakage model, with a share size of $(\mathsf{message\ length} + \mu)$, tolerating μ bits of leakage from each share. This result is obtained as a corollary of the following more general statement that we prove:

Result 1. *Given any secret sharing scheme*[4] *for general monotone access structure $\mathcal{A}$ with share size ℓ/R, where ℓ is the message length and $R \leq 1$, one can construct an LRSS for the same access structure $\mathcal{A}$, against the local leakage model allowing μ bits of leakage per share, with a share size of $\ell/R + \mu + o(\ell/R + \mu)$.*

Using our LRSS scheme from Result 1, along with the recent 1/3-rate non-malleable code of [AKO$^+$22] in the [GK18] NMSS compiler, we obtain an NMSS with a share size of only $(4 \cdot \mathsf{message\ length})$. Additionally, we also formalize a natural restriction of NMSS schemes to uniformly random secrets, called non-malleable randomness sharing (NMRS), and show how to construct this with a share size of $(2 \cdot \mathsf{message\ length})$. NMRS is useful in many practical applications of secret sharing that only require uniformly random secrets (e.g., when the secret to be shared and protected against malleability is an encryption key or a digital signature signing key, whose distribution is (typically) uniform). In particular, we show:

Result 2. *There exists a non-malleable secret sharing scheme against the independent tampering model for the threshold access structure, that achieves a share size of 4ℓ, for messages of length ℓ. There exists a non-malleable randomness sharing scheme against the independent tampering model for the threshold access structure that achieves a share size of 2ℓ, for messages of length ℓ.*

[3] In [BDIR18], under the same restrictions (on n, the field and the number of full shares allowed), the authors also consider the setting with threshold $t \leq \alpha n$, for $\alpha < 1$ for the Shamir secret sharing scheme, but this only allows constant bits of leakage per share. The work of Nielsen and Simkin [NS20] gives a lower bound for the share size of an LRSS in the local leakage model, which proves that the amount of leakage allowed on the Shamir scheme shown in [BDIR18] is the best possible. However, for the LRSS scheme of [SV19] or ours, their lower bound allows for leakage almost as large as the size of a single share.

[4] we require the secret sharing to satisfy an additional property of "local uniformity", which requires every share to individually have (an almost) uniform distribution. We show later that such a property is already satisfied by many natural secret sharing schemes (e.g.: Shamir secret sharing).

1.2 Technical Overview

One of the initial ideas [ADN$^+$19] to build an LRSS scheme against the local leakage model, was using linear invertible extractors in the following way: a) First, threshold secret share the message m into n shares $m_1, \cdots, m_n$; b) then invert each share m_i under an invertible extractor to get (w_i, s_i); c) Finally, the i-th share contains w_i and all s_j's except for $j = i$. This scheme, even when instantiated with the best known linear extractors, has a share size of $((n-1) \cdot \mathsf{message\ length} + |w_i|)$, which will not be optimal (even for a constant number of parties). This is because this construction mandates the size of the seeds s_j's to be as long as the message m in order to get a negligible leakage error. Furthermore, this scheme allows a leakage of size $((1 - o(1)) \cdot \mathsf{message\ length})$ only for a constant number of parties.

In a subsequent work of [SV19], the authors once again rely on the use of randomness extractors, but use a single seed s, across all the shares to get a rate improvement. In particular, they do the following: a) First m is threshold secret shared into $m_1, \cdots, m_n$ (referred to as "simple shares") using a threshold secret sharing scheme b) Next, each m_i is masked using an extractor output $\mathsf{Ext}(w_i, s)$ where s and w_i's are uniformly chosen. Now, let $sh_1, \cdots, sh_n$ denote these masked shares c) r is uniformly chosen to additionally mask each sh_i d) Finally, r and s are together secret shared using a 2-out-of-n secret sharing scheme into shares $(a_1, \cdots, a_n)$ and the i^{th} share of the scheme is then set to be $(w_i, sh_i \oplus r, a_i)$. At a high level, m_i was "doubly masked" in order to cast the leakage on the i^{th} share as leakage on the extractor source w_i. In order to add leakage resilience on top of the simple shares, they needed to be masked twice, and thus, information of both these masks and the masked value (each roughly of size $|m_i|$) is given as part of the final share, resulting in its length being approximately $3|m_i|$, and hence, giving a rate of $1/3$.

In our construction, we try to combine the best things from these two constructions, i.e., use of the invertibility of linear extractors with great parameters, and the use of a single seed across all shares to optimize the share size. Our techniques use linear extractors in such a way that we not only remove the dependence of the share size on the number of parties (which in itself is important), but also obtain an optimal rate of 1 while still allowing a leakage of size $((1 - o(1)) \cdot \mathsf{message\ length})$. We now proceed to describe our approach.

A simpler problem. Our goal is to compile simple shares m_i into leakage resilient shares in a share size-preserving manner (i.e., the size of the leakage resilient share needs to be about the same as $|m_i|$). As a first step, we relax the problem in two ways a) consider an LRSS only for the (n, n) access structure (i.e., where the set of all n parties is the only authorized set); and b) require that the sharing scheme only works for uniformly random messages.

We first construct an LRSS scheme with the desired share size of $\mathsf{message\ length} + \mu$, under these two relaxations. For this, we choose extractor sources $w_1, \cdots, w_n$ and a seed s uniformly, secret share s as $(s_1, \cdots, s_n)$ and define $share_i$ to be (w_i, s_i). Now, define reconstruction of $share_i$'s as $m = \oplus_{i \in [n]} \mathsf{Ext}(w_i; s)$, where s is obtained by reconstructing s_i's. Now, first observe that, by extractor security, the reconstructed value m has (almost) uniform distribution. Also, each share supports local leakage resilience as s_i (a share) is devoid of information about s and hence any bounded leakage of the form $f(w_i, s_i)$ is only dependent on w_i and is (almost) independent of the extractor output and hence m too. This scheme infact has a share size of $|m| + \mu$ (for μ bits of leakage per share) as there are explicit extractor constructions with good parameters such that $|s| \ll |w_i| \approx |\mathsf{Ext}(w_i, s)|$.

Final construction overview. Unfortunately, the above construction does not extend to either support threshold access structures or for secret sharing a specific message m. In order to reconstruct to a message m, the extractor outputs $\mathsf{Ext}(w_i, s)$'s ($i \in [n]$) would have to be correlated. However, the fact that the extractor outputs $\mathsf{Ext}(w_i, s)$ are uncorrelated is what gives leakage resilience in the scheme above for sharing random messages. The main technical hurdle which we overcome in this work is to ensure correlation in the shares while retaining enough independence (via extractors) so that we can argue leakage resilience.

In our construction, we first generate simple shares of m, denoted $(m_1, \cdots, m_n)$ using a standard secret sharing scheme. Next, we aim to cast each of these simple shares m_i as an extractor output. This, however, has two challenges a) the distribution of m_i could be arbitrary and need not have any entropy; and b) it is not clear how to express m_i's as the output of an extractor. To address (a), we observe that many natural secret sharing schemes (for example, the Shamir secret sharing scheme) satisfy the property that each share individually has (an almost) uniform distribution. We formalize this property as "local uniformity" of a secret sharing scheme and generate simple shares of m using such a locally uniform secret sharing scheme. To solve the challenge (b), we make use of seeded extractors that are linear functions - i.e., where the extractor function is guaranteed to be a linear map (over the source) for any fixed value of the seed, called linear seeded extractors [Tre99, Tre01, RRV02]. We show that such extractors provide an efficient way to find an (entropic sample of an) extractor source such that the extractor output on this source takes a given value under a given seed. With this useful property, each of our simple shares can indeed be expressed as extractor outputs[5].

To summarize our construction, we a) secret share m into simple shares $m_1, \cdots, m_n$ using any locally uniform secret sharing scheme for the given general access structure; b) choose a seed s uniformly and generate its shares $s_1, \cdots, s_n$ such that s can be reconstructed from any two shares s_i and s_j; c) for each m_i, sample w_i such that $\mathsf{Ext}(w_i, s) = m_i$; d) Finally, each share is set to be (w_i, s_i) for all $i \in [n]$. Leakage resilience of this scheme follows from a careful argument using extractor security and local uniformity. In this scheme, the length of each share (w_i, s_i) is only negligibly larger than the length of m_i as there are explicit constructions of linear extractors that extract out almost all the entropy from the source while only using very short seeds.

Non-malleable randomness sharing. We obtain a non-malleable secret sharing (NMSS) scheme with a share size of $4(\mathsf{message\ length})$ by instantiating the NMSS compiler from [GK18] with our rate-1 LRSS scheme, along with the recent rate-1/3 NMC from [AKO$^+$22]. Hence, our focus in the main section will be in formalizing and building the NMRS scheme with a share size of $2(\mathsf{message\ length})$. Our NMRS construction follows the same blueprint as [GK18], but uses a non-malleable randomness encoder (NMRE) (instead of using a non-malleable code) and our LRSS scheme with rate 1. NMREs [KOS18], outputs a random message along with its encoding L, R, with the guarantee that whenever an adversary tampers L, R (in a split-state manner, i.e., tamper L and R independent of each other), the original message looks uniformly random, even given this tampered message. Now, our NMRS construction outputs the random message m output by the NMRE, and to generate its shares: first secret share L using our LRSS scheme for the 2-out-of-n threshold setting and then share R using a t-out-of-n threshold secret sharing scheme.

[5] A similar technique of using linear and invertible extractors to get rate optimality has been used in two prior settings before: information-theoretic privacy of communication data in the wiretap channel setting in [BT12, CDS12] and binary secret sharing schemes in [LCG$^+$19]

1.3 Related Work

The problems of leakage resilient and non-malleable secret sharing has seen much research in recent times [DDV10, LL12, BDIR18, GK18, BS19, SV19, ADN$^+$19], [FV19, BFV19, KMS19, LCG$^+$19, CGG$^+$20, BFO$^+$20, CKOS21]. In the information -theoretic setting, majority of these works focus on improving the leakage model, such as allowing the adversary to obtain adaptive (leakage queries dependent on prior leakage responses) and joint (combined leakage from multiple shares) leakages, and such strong leakage models come at the expense of poor and sub-optimal share size (typically $\omega(\mathsf{message\ length})$). For the case where the adversary is restricted to be computationally bounded, the works of [BFO$^+$20, FV19] show NMSS and LRSS schemes achieving optimal rate for strong adaptive and joint leakage and tampering models.

1.4 Organization of the Paper

We give the preliminary definitions and lemmata in Section 2. Then, we build our leakage resilient secret sharing scheme in Section 3. Finally, we define and build our non-malleable randomness sharing scheme in Section 4.

2 Preliminaries

2.1 Notation

We begin by describing a few notations that we use. For any two sets S and S', $S\backslash S'$ denotes the set of elements that are present in S, but not in S'. For any natural number n, $[n]$ denotes the set $\{1, 2, \cdots, n\}$. $x \leftarrow X$ denotes sampling from a probability distribution X. $x||y$ represents concatenation of two binary strings x and y. $|x|$ denotes length of binary string x. U_l denotes the uniform distribution on $\{0,1\}^l$. All logarithms are base 2. If S is a subset of $[n]$ and $x_1, .., x_n$ are some variables or elements, then x_S denotes the set $\{x_i \text{ such that } i \in S\}$. $\chi(a = b)$ indicates equality of the strings a and b (i.e $\chi(a = b) = 1$ is an only if a is equal to b). In this paper we assume natural one-to-one correspondence between the set $\{0,1\}^n$ and the field $GF(2^n)$.

We give standard definitions of statistical distance and entropy along with some preliminary lemmata of the same in Appendix A.

2.2 Randomness Extractors

Extractors (introduced by Nissan and Zuckerman [NZ96]) output an almost uniform string from a (η, τ)-entropic source, using a short uniform string, called *seed*, as a catalyst. Average-case extractors are extractors whose output remains close to uniform, even given the seed and some auxiliary information about the source (but independent of the seed), whenever the source has enough average entropy given the auxiliary information. We formally define them as below.

Definition 1. *[DORS08] Let* $\mathsf{Ext} : \{0,1\}^\eta \times \{0,1\}^d \to \{0,1\}^l$ *be a polynomial time computable function. We say that* Ext *is an* ***efficient average-case*** $(\eta, \tau, d, l, \epsilon)$*-**strong extractor** if for all pairs of random variables* (W, Z) *such that* W *is a random variable over* η*-bit strings satisfying* $\widetilde{\mathbf{H}}_\infty(W|Z) \geq \tau$*, we have*

$$\mathsf{Ext}(W; U_d), U_d, Z \approx_\epsilon U_l, U_d, Z$$

Linear Extractors. *Further, the average-case strong extractor* Ext *is said to be linear if for every* $s \in \{0,1\}^d$, $\mathsf{Ext}(\cdot, s)$ *is a linear function.*

In this paper, we instantiate linear extractors with extractors due to Raz et.al [RRV02], which extracts almost all the randomness and is an improvement of Trevisan's extractor [Tre99]. Particularly, we use the following instantiation of the same given in [LCG+19].

Lemma 1. *[LCG+19, Lemma 6] There is an explicit* $(\eta, \tau, d, l, \epsilon)$*-strong linear extractor with* $d = \mathcal{O}(\log^3(\frac{\eta}{\epsilon}))$ *and* $l = \tau - \mathcal{O}(d)$.

In our application of linear extractors we will often require to uniformly sample an extractor source such that the extractor output on this source and a given seed s takes a given value y. Basically, given a seed s and some $y \in \{0,1\}^l$, the inverting function needs to sample an element uniformly from the set $\mathsf{Ext}(\cdot, s)^{-1}(y)$ (which is $\{w : \mathsf{Ext}(w; s) = y\}$). We formalize this procedure[6] as InvExt and show that linear extractors allow such sampling in the following lemma.

Lemma 2. *For every efficient linear extractor* Ext*, there exists an efficient randomized function* $\mathsf{InvExt} : \{0,1\}^l \times \{0,1\}^d \to \{0,1\}^\eta \cup \{\perp\}$ *(termed inverter) such that*

1. $U_\eta, U_d, \mathsf{Ext}(U_\eta; U_d) \equiv \mathsf{InvExt}(\mathsf{Ext}(U_\eta; U_d), U_d), U_d, \mathsf{Ext}(U_\eta; U_d)$
2. *For each* $(s, y) \in \{0,1\}^d \times \{0,1\}^l$,

(a) $\Pr[\mathsf{InvExt}(y, s) = \perp] = 1$*, if and only if there exists no* $w \in \{0,1\}^\eta$ *such that* $\mathsf{Ext}(w; s) = y$.

(b) $\Pr[\mathsf{Ext}(\mathsf{InvExt}(y, s); s) = y] = 1$*, if there exists some* $w \in \{0,1\}^\eta$ *such that* $\mathsf{Ext}(w; s) = y$.

Proof. Recall that for a linear extractor, for any seed $s \in \{0,1\}^d$, $\mathsf{Ext}(\cdot, s)$ is a linear map from the vector space $\{0,1\}^\eta$ to the vector space $\{0,1\}^l$. Let $\mathcal{I}_s$ and $\mathcal{K}_s$ denote the image and kernel of this linear map $\mathsf{Ext}(\cdot, s)$. We now define InvExt as follows. Fix any arbitrary input y, s to InvExt.
$\mathsf{InvExt}(y, s)$:

```
• If $y \in \mathcal{I}_s$
    – Let $w$ be such that $\mathsf{Ext}(w; s) = y$
    – Sample $z$ uniformly from $\mathcal{K}_s$
    – Output $w + z$
  Else output $\perp$
```

InvExt is efficient because the bases for the linear sub-spaces $\mathcal{K}_s$, $\mathcal{I}_s$ and the preimage space on the value y (corresponding to the linear map $\mathsf{Ext}(\cdot, s)$) can be determined efficiently. By the definition, it is easy to see that InvExt satisfies property (2) of the Lemma statement. We now proceed to prove property (1) about statistical distance. Consider the set $\mathcal{S} = \{(w, s, y) : \mathsf{Ext}(w; s) = y\}$.
For any $(w, s, y) \in \mathcal{S}$,

$$\begin{aligned} &\Pr[(\mathsf{InvExt}(\mathsf{Ext}(U_\eta; U_d), U_d), U_d, \mathsf{Ext}(U_\eta; U_d)) = (w, s, y)] \\ &= \sum_{w' \in \{0,1\}^\eta} \Pr[U_\eta = w', U_d = s] \cdot \Pr[\mathsf{InvExt}(y, s) = w] \cdot \chi(\mathsf{Ext}(w'; s) = y) \end{aligned}$$

[6] In literature, invertible (seeded) extractors (see [CDS12] for an exposition on the same) are well-studied which allow efficient sampling of a source w and a seed s such that the extractor output on w and s equals a given value y. Note that our requirement to sample a source w given a seed s and a value y is stronger than the guarantee provided by invertible extractors. Hence we explicitly show that certain extractors allow such sampling.

Since $\mathsf{Ext}(w;s) = y$ by definition of $\mathcal{S}$, we know that w lies in the set of $|\mathcal{K}_s|$ elements from which $\mathsf{InvExt}(y,s)$ chooses its output uniformly. Therefore $\Pr[\mathsf{InvExt}(y,s) = w] = \frac{1}{|\mathcal{K}_s|}$. Further, since $\mathsf{Ext}(\cdot;s)$ is a linear map and $y \in \mathcal{I}_s$, we know that there are exactly $|\mathcal{K}_s|$ number of values $w' \in \{0,1\}^\eta$ such that $\mathsf{Ext}(w';s) = y$. With these observations, we conclude that

$$\Pr[(\mathsf{InvExt}(\mathsf{Ext}(U_\eta;U_d),U_d),U_d,\mathsf{Ext}(U_\eta;U_d)) = (w,s,y)] = \frac{1}{2^{\eta+d}}$$

Also for any $(w,s,y) \in \mathcal{S}$, $\Pr[(U_\eta,U_d,\mathsf{Ext}(U_\eta;U_d)) = (w,s,y)] = \frac{1}{2^{\eta+d}}$.

For any $(w,s,y) \notin \mathcal{S}$, it holds that $\mathsf{Ext}(w;s) \neq y$. With this we have

$$Pr[(U_\eta,U_d,\mathsf{Ext}(U_\eta;U_d)) = (w,s,y)] = 0$$

and

$$\Pr[(\mathsf{InvExt}(\mathsf{Ext}(U_\eta;U_d),U_d),U_d,\mathsf{Ext}(U_\eta;U_d)) = (w,s,y)] = 0$$

The last equation is true because a) if $y \in \mathcal{I}_s$, then $\Pr[\mathsf{InvExt}(y,s) = w] = 0$ b) if $y \notin \mathcal{I}_s$, then $\Pr[(U_d,\mathsf{Ext}(U_\eta;U_d)) = (s,y)] = 0$. Further note that $\Pr[\mathsf{InvExt}(\mathsf{Ext}(U_\eta;U_d),U_d) = \perp] = 0$ as $\mathsf{Ext}(U_\eta;U_d) \in \mathcal{I}_{U_d}$ with probability 1. Combining these observations, it follows that the statistical distance between the distributions $(\mathsf{InvExt}(\mathsf{Ext}(U_\eta;U_d),U_d),U_d,\mathsf{Ext}(U_\eta;U_d))$ and $(U_\eta,U_d,\mathsf{Ext}(U_\eta;U_d))$ is zero, which concludes the proof. □

2.3 Secret Sharing Schemes

Secret sharing schemes provide a mechanism to distribute a secret into shares such that only an authorized subset of shares can reconstruct the secret and any unauthorized subset of shares has "almost" no information about the secret. We now define secret sharing schemes formally.

Definition 2. *Let $[n]$ be a set of identities (indices) of n parties. A sharing function* $\mathsf{Share} : \{0,1\}^l \to (\{0,1\}^{l'})^n$ *is an $(n,\mathcal{A})$- **secret sharing scheme** that is ϵ_s-private with respect to a monotone access structure*[7] *$\mathcal{A}$ if the following two properties hold:*

1. ***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* $\mathsf{Rec} : (\{0,1\}^{l'})^{|T|} \to \{0,1\}^l$ *such that for every $m \in \mathcal{M}$,*

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

where the probability is over the randomness of the Share *function and if $(sh_1,..,sh_n) \leftarrow$* $\mathsf{Share}(m)$*, then* $\mathsf{Share}(m)_T$ *denotes $\{sh_i\}_{i\in T}$. 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.*

2. ***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 $U \notin \mathcal{A}$, and for every pair of secrets $m, m' \in \{0,1\}^l$,*

$$\Delta((\mathsf{Share}(m))_U; (\mathsf{Share}(m'))_U) \leq \epsilon_s$$

[7] $\mathcal{A}$ is a monotone access structure if for all A, B such that $A \subset B \subseteq [n]$ and $A \in \mathcal{A}$, it holds that $B \in \mathcal{A}$. Throughout this paper whenever we consider a general access structure, we mean a monotone access structure.

$(\mathsf{Share}, \mathsf{Rec})$ *is said to be perfectly private if* $\epsilon_s = 0$. *An access structure* $\mathcal{A}$ *is said to be* (n, t)*-threshold if and only if* $\mathcal{A}$ *contains all subsets of* $[n]$ *of size atleast* t.

Rate *of a secret sharing scheme is defined as* $\frac{\textsf{message size}}{\textsf{share size}}$*(which would be equal to* l/l'*).*

Leakage Resilience. *A secret sharing scheme* $(\mathsf{Share}, \mathsf{Rec})$ *is said to be* ϵ_{lr}*-leakage resilient against a leakage function family* $\mathcal{F}$ *if for all messages* $m, m' \in \{0,1\}^l$ *and every function* $f \in \mathcal{F}$,

$$f((\mathsf{Share}(m))_{[n]}) \approx_{\epsilon_{lr}} f((\mathsf{Share}(m'))_{[n]})$$

We use secret sharing schemes augmented with the following property as a building block to our leakage resilient secret sharing scheme.

Local Uniformity. We say a secret sharing scheme $(\mathsf{Share}, \mathsf{Rec})$ satisfies *local uniformity* if the distribution of each individual share given out by the Share function is ϵ_u-statistically close to the uniform distribution in its share space. Formally, any sharing function $\mathsf{Share} : \{0,1\}^l \to \{\{0,1\}^{l'}\}^n$ is ϵ_u-locally uniform if for each message $m \in \{0,1\}^l$ it holds that

$$\mathsf{Share}(m)_{\{i\}} \approx_{\epsilon_u} U_{l'}, \ \forall\, i \in [n]$$

Note that Shamir secret sharing scheme [Sha79, Bla79] and Benaloh-Leichter secret sharing scheme [BL88] are instantiations of a locally uniform secret sharing schemes for threshold access structures and general monotone access structures respectively[8] .

Consistent Resampling. For any $(n, \mathcal{A})$-secret sharing scheme $(\mathsf{Share}, \mathsf{Rec})$ which is ϵ_s-private, and for any message m and a subset $\mathcal{L} \subseteq [n]$, when we say "$(sh_1, .., sh_n) \leftarrow \mathsf{Share}(m|sh^*_{\mathcal{L}})$" we mean the following procedure:

- Sample and output $(sh_1, .., sh_n)$ uniformly from the distribution $\mathsf{Share}(m)$ conditioned on the event that $sh_{\mathcal{L}} = sh^*_{\mathcal{L}}$
- If the above event is a zero probability event then output a string of all zeroes (of appropriate length).

Note that for any $\mathcal{L} \subseteq [n]$, the distributions $\mathsf{Share}(m)$ and $\mathsf{Share}(m|sh^*_{\mathcal{L}})$ are identical when $(sh^*_1, \cdots, sh^*_n) \leftarrow \mathsf{Share}(m)$.

3 Leakage Resilient Secret Sharing Schemes

3.1 Local Leakage Family

The local leakage family allows bounded leakage queries $\{f_i : \{0,1\}^{l'} \to \{0,1\}^{\mu}\}_{i \in \mathcal{K}}$, on each share corresponding to an arbitrary set of indices $\mathcal{K}(\subseteq [n])$, and further allows full share queries corresponding to an unauthorised subset $\mathcal{U}$. Formally, for any access structure $\mathcal{A}$ and leakage amount $\mu > 0$, we define this family as

$$\mathcal{F}_{\mathcal{A},\mu} = \{(\mathcal{U}, \mathcal{K}, \{f_i\}_{i \in \mathcal{K}}) : \mathcal{U} \notin \mathcal{A}, \mathcal{K} \subseteq [n] \text{ and } \forall\ i \in \mathcal{K}, f_i : \{0,1\}^{l'} \to \{0,1\}^{\mu}\}$$

[8] This is formally proven in [CKOS21, Claim 2].

where for any secret sharing scheme $(\mathsf{Share}, \mathsf{Rec})$, the leakage response corresponding to a leakage query $(\mathcal{U}, \mathcal{K}, \{f_i\}_{i\in\mathcal{K}}) \in \mathcal{F}_{\mathcal{A},\mu}$ on any secret m is $(sh_{\mathcal{U}}, \{f_i(sh_i)\}_{i\in\mathcal{K}})$ when $(sh_1, \cdots, sh_n) \leftarrow \mathsf{Share}(m)$.

Remark 1. *Consider a leakage family which is the set of all functions $(\mathcal{U}, \mathcal{K}, \{f_i\}_{i\in\mathcal{K}}) \in \mathcal{F}_{\mathcal{A},\mu}$ such that $\mathcal{U} \cap \mathcal{K} = \phi$. Intuitively, this is the leakage query which doesn't ask to reveal a full share and also query bounded leakage on the same share. Though this may seem like a restriction on $\mathcal{F}_{\mathcal{A},\mu}$, we would like to emphasize that leakage resilience against this weaker family guarantees leakage resilience against $\mathcal{F}_{\mathcal{A},\mu}$ itself. This is because leakage response to any function $(\mathcal{U}, \mathcal{K}, \{f_i\}_{i\in\mathcal{K}}) \in \mathcal{F}_{\mathcal{A},\mu}$ can be simulated from leakage response to $(\mathcal{U}, \mathcal{K}\backslash\mathcal{U}, \{f_i\}_{i\in\mathcal{K}\backslash\mathcal{U}}) \in \mathcal{F}_{\mathcal{A},\mu}$, as $\{f_i(sh_i)\}_{i\in\mathcal{K}\cap\mathcal{U}}$ can be trivially computed given $sh_{\mathcal{U}}$(which is part of the leakage response to $(\mathcal{U}, \mathcal{K}\backslash\mathcal{U}, \{f_i\}_{i\in\mathcal{K}\backslash\mathcal{U}})$).*

3.2 Construction

3.2.1 Building Blocks

- $(\mathsf{MShare}, \mathsf{MRec})$ be any ϵ_p-private and ϵ_u-locally uniform secret sharing scheme for the message space $\{0,1\}^l$ and a monotone access structure $(n, \mathcal{A})$. Let l' denote the share size of MShare (that is $\mathsf{MShare} : \{0,1\}^l \to (\{0,1\}^{l'})^n$).
- $(\mathsf{SdShare}, \mathsf{SdRec})$ be any ϵ'_p-private secret sharing scheme for the message space $\{0,1\}^d$ and against the $(2,n)$-threshold access structure with share length d'.
- Ext be an $(\eta, \tau, d, l', \epsilon_{ext})$-strong linear extractor. InvExt be the inverter function corresponding to Ext given by Lemma 2.

3.2.2 Construction Overview

We now build our LRSS scheme. Informally, to share a message m, we first share it using MShare to get $m_1, \cdots, m_n$, pick a random extractor seed s and then use InvExt to get the source w_i corresponding to the extractor output m_i and seed s, for each $i \in [n]$. If any of the w_i is $\perp$, then we output each of the i-th share to be $(\perp, m_i)$. Else, we share s using $\mathsf{SdShare}$ to get $s_1, \cdots, s_n$, and set the i-th share to be (w_i, s_i). The reconstruction procedure either directly reconstructs using m_i's (in case of $\perp$), else reconstructs s, evaluates the extractor Ext on w_i and s to get the m_i's and recovers m.

```
$\mathsf{LRShare}(m)$
  • $(m_1, \cdots, m_n) \leftarrow \mathsf{MShare}(m)$.
  • Sample $s \in_R \{0,1\}^d$.
  • $(s_1, \cdots, s_n) \leftarrow \mathsf{SdShare}(s)$.
  • For $i \in [n]$, $w_i \leftarrow \mathsf{InvExt}(m_i, s)$.
  • If $w_j = \perp$ for some $j \in [n]$, set $share_i = (\perp, m_i)$ for each $i \in [n]$.
  • Else, for each $i \in [n]$, set $share_i = (w_i, s_i)$.
  • Output $(share_1, \cdots, share_n)$.
```

```
$\mathsf{LRRec}(share_T)$ (where $T \in \mathcal{A}$)
  • If for any $i \in T$, $share_i$ is of the form $(\perp, m_i)$, then parse each $share_j$ as $(\perp, m_j)$ for each $j \in T$
  • Else, for $i \in T$, parse $share_i$ as $(w_i, s_i)$ and do:
      – $s \leftarrow \mathsf{SdRec}(s_{i_1}, s_{i_2})$, where $i_1, i_2$ are two indices from $T$.
      – For $i \in T$, set $m_i = \mathsf{Ext}(w_i; s)$.
  • Output $m \leftarrow \mathsf{MRec}(m_T)$.
```

Theorem 1. *Let* $(\mathsf{MShare}, \mathsf{MRec})$*,* $(\mathsf{SdShare}, \mathsf{SdRec})$ *and* $(\mathsf{Ext}, \mathsf{InvExt})$ *be the secret sharing schemes and a strong linear extractor as given in Section 3.2.1. Then* $(\mathsf{LRShare}, \mathsf{LRRec})$ *is a leakage resilient secret sharing scheme for messages in* $\{0,1\}^l$ *against the access structure* $(n, \mathcal{A})$ *which is* ϵ_p*-private and* $(6n(\epsilon_{ext} + \epsilon'_p + \epsilon_u) + \epsilon_p)$ *-leakage resilient against the local leakage family* $\mathcal{F}_{\mathcal{A},\mu}$*.*

Also, for every instantiation of $(\mathsf{MShare}, \mathsf{MRec})$ *with rate* $R(l)$ [9] *on secrets of size* l*, there exists an instantiation of* $(\mathsf{LRShare}, \mathsf{LRRec})$ *with a share size of approximately* $(l/R(l) + \mu)$*, for* μ *bits of leakage per share. In particular, for* $\mu = o(l/R(l))$*, we get the same rate* $R(l)$ *for our LRSS scheme.*

Further, there exists an efficient instantiation of $(\mathsf{LRShare}, \mathsf{LRRec})$ *for threshold access structures on secrets of size* l*, which has a share size of approximately* $(l + \mu)$*, for* μ *bits of leakage (and in particular gives rate 1, when* $\mu = o(l)$*), that is perfectly private and* $6n \cdot 2^{-\Omega(\sqrt[3]{(l/\log l)})}$*-leakage resilient against* $\mathcal{F}_{\mathcal{A},\mu}$*.*

3.3 Security Proof

Correctness and Privacy

Correctness of the scheme follows from the correctness of $(\mathsf{MShare}, \mathsf{MRec})$ in case any InvExt outputs $\perp$, else it follows from correctness of both $(\mathsf{MShare}, \mathsf{MRec})$, $(\mathsf{SdShare}, \mathsf{SdRec})$ and properties of InvExt (property 2(b) of Lemma 2). It is easy to see that $(\mathsf{LRShare}, \mathsf{LRRec})$ is ϵ_p-private by ϵ_p-privacy of $(\mathsf{MShare}, \mathsf{MRec})$.

Leakage resilience against the local leakage family

Choose an arbitrary leakage function $(\mathcal{U}, \mathcal{K}, \{f_i\}_{i \in \mathcal{K}}) \in \mathcal{F}_{\mathcal{A},\mu}$. Note that by Remark 1, it suffices to show leakage resilience against leakage functions such that $\mathcal{U} \cap \mathcal{K} = \phi$. For the sake of simplicity assume $\mathcal{K} = \{1, 2, \cdots, |\mathcal{K}|\}$.

Our goal is to show that the distributions of leakage response to the query $(\mathcal{U}, \mathcal{K}, \{f_i\}_{i \in \mathcal{K}})$ on shares of two distinct messages m and m' are statistically close. We denote the distribution of these leakage responses on m and m' by Leak_0^m and $\mathsf{Leak}_0^{m'}$ respectively.

In case either of the shares corresponding to m or m' contain $\perp$, then we do not get leakage resilience, however in Claim 1, we show that the shares corresponding to any message m contain

[9] Here, we let R denote the function that computes the rate to secret share l-size secrets.

$\perp$, only with probability $(n(\epsilon_{ext} + \epsilon_u))$.

Claim 1. *For any message m, $\mathsf{LRShare}(m) = ((\perp, m_1), \cdots, (\perp, m_n))$ with probability $\leq (n(\epsilon_{ext} + \epsilon_u))$.*

Proof. Let M_i, W_i, S_i (for $i \in [n]$) and S denote the distributions of the samples m_i, w_i, s_i(for $i \in [n]$) and s respectively in the sharing procedure $\mathsf{LRShare}(m)$. By definition of $\mathsf{LRShare}$, for any m, the probability that $\mathsf{LRShare}(m) = ((\perp, m_1), \cdots, (\perp, m_n))$ is $= \Pr[\exists i \in [n] : W_i = \perp]$. We now analyze this probability. Let $\mathcal{I}_s$ denote the image of the linear map $\mathsf{Ext}(\cdot, s)$ for $s \in \{0,1\}^d$. By Lemma 2, note that $\mathsf{InvExt}(m_i, s)$ outputs $\perp$ if and only if $m_i \notin \mathcal{I}_s$. Therefore,

$$\Pr[\exists i \in [n] : W_i = \perp] \leq \sum_{i \in [n]} \Pr[M_i \notin \mathcal{I}_S]$$

Since Ext is a strong linear extractor, we know

$$\mathsf{Ext}(U_\eta, S), S \approx_{\epsilon_{ext}} U_{l'}, S.$$

By local uniformity of MShare, for each $i \in [n]$ we have $M_i \approx_{\epsilon_u} U_{l'}$. Since S is independent of M_i and $U_{l'}$ it follows that,

$$\forall i \in [n],\ \mathsf{Ext}(U_\eta, S), S \approx_{\epsilon_{ext}+\epsilon_u} M_i, S.$$

By the definition of statistical distance, for each $i \in [n]$

$$\Pr[M_i \notin \mathcal{I}_S] \leq \epsilon_{ext} + \epsilon_u + \Pr[\mathsf{Ext}(U_\eta, S) \notin \mathcal{I}_S] = \epsilon_{ext} + \epsilon_u.$$

The last implication follows because $\mathsf{Ext}(U_\eta, S) \in \mathcal{I}_S$ with probability 1. Therefore,

$$\Pr[\exists i \in [n] : W_i = \perp] \leq n(\epsilon_{ext} + \epsilon_u).$$

■ □

Thus, assuming this error of $(2n(\epsilon_{ext}+\epsilon_u))$, we can consider all shares of m and m' to not contain $\perp$, and through a sequence of hybrids $\mathsf{Leak}^m_0, \mathsf{Leak}^m_1, \cdots, \mathsf{Leak}^m_{|\mathcal{K}|}$ we show that the responses to the leakage functions f_i's are (almost) independent of the choice of m in Claim 2. A similar sequence of hybrids is followed for the message m'. Then, we show that the distributions of the shares of the messages m and m' corresponding to the set $\mathcal{U}$ are statistically close in Claim 3. Together, these claims prove leakage resilience. We formally describe these hybrids below (recall we assume the non-$\perp$ case here).

Leak_0^m

- $(m_1, \cdots, m_n) \leftarrow \mathsf{MShare}(m)$.
- Sample $s \in_R \{0,1\}^d$.
- $(s_1, \cdots, s_n) \leftarrow \mathsf{SdShare}(s)$.
- For $i \in [n]$, $w_i \leftarrow \mathsf{InvExt}(m_i, s)$.
- For $i \in [n]$, set $share_i = (w_i, s_i)$.
- Output $(\{f_i(share_i)\}_{i \in [|\mathcal{K}|]}, share_{\mathcal{U}})$

$\mathsf{Leak}_{\mathsf{j}}^m$ $(j \in [|\mathcal{K}|])$

- $(m_1, \cdots, m_n) \leftarrow \mathsf{MShare}(m)$.
- Sample $s \in_R \{0,1\}^d$.
- $(s_1, \cdots, s_n) \leftarrow \mathsf{SdShare}(s)$.
- For $1 \leq i \leq j$, $w_i \in_R \{0,1\}^\eta$.
- For $j < i \leq n$, $w_i \leftarrow \mathsf{InvExt}(m_i, s)$.
- For $i \in [n]$, set $share_i = (w_i, s_i)$.
- Output $(\{f_i(share_i)\}_{i \in [|\mathcal{K}|]}, share_{\mathcal{U}})$

Leak_0^m captures the response to the leakage query $(\mathcal{U}, \mathcal{K}, \{f_i\}_{i \in \mathcal{K}})$ on message m corresponding to the sharing function $\mathsf{LRShare}$. Particularly, in Leak_0^m all responses $f_j(share_j)$ have dependence on m via w_j (as w_j is correlated to m_j, a share of m). Informally, hybrids $\mathsf{Leak}_{\mathsf{j}}^m$ and $\mathsf{Leak}_{\mathsf{j-1}}^m$ differ only in the computation of w_j, where w_j is chosen uniformly in $\mathsf{Leak}_{\mathsf{j}}^m$ while it is sampled using InvExt in $\mathsf{Leak}_{\mathsf{j-1}}^m$ (as in the actual leakage distribution Leak_0^m). We now use the security guarantees provided by $(\mathsf{Ext}, \mathsf{InvExt})$ and local uniformity of MShare to prove that the successive hybrids $\mathsf{Leak}_{\mathsf{j-1}}^m$ and $\mathsf{Leak}_{\mathsf{j}}^m$ are statistically close, for each $j \in [|\mathcal{K}|]$.

Claim 2. *By ϵ_u-local uniformity of* MShare*, ϵ'_p-privacy of* $(\mathsf{SdShare}, \mathsf{SdRec})$ *and security of the strong linear extractor* Ext*, for each $j \in [|\mathcal{K}|]$,* $\mathsf{Leak}_{\mathsf{j-1}}^m \approx_{2(\epsilon_{ext}+\epsilon'_p+\epsilon_u)} \mathsf{Leak}_{\mathsf{j}}^m$.

Proof. For any $j \in [|\mathcal{K}|]$, the distributions $\mathsf{Leak}_{\mathsf{j-1}}^m$ and $\mathsf{Leak}_{\mathsf{j}}^m$ only differ in computation of w_j (which in turn influences computation of $share_j$ and $f_j(share_j)$). Let W and S denote uniform distributions on $\{0,1\}^\eta$ and $\{0,1\}^d$ respectively. From Lemma 2 we have,

$$W, S, \mathsf{Ext}(W;S) \equiv \mathsf{InvExt}(\mathsf{Ext}(W;S), S), S, \mathsf{Ext}(W;S)$$

Let $\tilde{S}_j \equiv \mathsf{SdShare}(0^d)_{\{j\}}$. By Lemma 6 we have,

$$f_j(W, \tilde{S}_j), \tilde{S}_j, S, \mathsf{Ext}(W;S) \equiv f_j(\mathsf{InvExt}(\mathsf{Ext}(W;S), S), \tilde{S}_j), \tilde{S}_j, S, \mathsf{Ext}(W;S)$$

Since $\widetilde{\mathbf{H}}_\infty(W | (f_j(W, \tilde{S}_j), \tilde{S}_j)) \geq \eta - \mu \geq \tau$ (by our setting of parameters), we invoke extractor security of Ext to get,

$$f_j(W, \tilde{S}_j), \tilde{S}_j, S, U_{l'} \approx_{\epsilon_{ext}} f_j(W, \tilde{S}_j), \tilde{S}_j, S, \mathsf{Ext}(W;S)$$

By triangle inequality on the above two inequalities,

$$f_j(W, \tilde{S}_j), \tilde{S}_j, S, U_{l'} \approx_{\epsilon_{ext}} f_j(\mathsf{InvExt}(\mathsf{Ext}(W;S), S), \tilde{S}_j), \tilde{S}_j, S, \mathsf{Ext}(W;S) \tag{1}$$

Observe that RHS of the inequality 1 is a randomised function (with randomness being independent of the input) of $(\tilde{S}_j, S, \mathsf{Ext}(W;S))$. Let g_1 denote this function. From Inequality 1 and Lemma 6 we have

$$g_1(\tilde{S}_j, S, U_{l'}) \approx_{\epsilon_{ext}} g_1(\tilde{S}_j, S, \mathsf{Ext}(W;S))$$

Then, by definition of g_1 we have

$$f_j(\mathsf{InvExt}(U_{l'}, S), \tilde{S}_j), \tilde{S}_j, S, U_{l'} \approx_{\epsilon_{ext}} f_j(\mathsf{InvExt}(\mathsf{Ext}(W; S), S), \tilde{S}_j), \tilde{S}_j, S, \mathsf{Ext}(W; S) \tag{2}$$

Applying triangle inequality on inequalities 1 and 2 we have,

$$f_j(W, \tilde{S}_j), \tilde{S}_j, S, U_{l'} \approx_{2\epsilon_{ext}} f_j(\mathsf{InvExt}(U_{l'}, S), \tilde{S}_j), \tilde{S}_j, S, U_{l'} \tag{3}$$

By privacy of $\mathsf{SdShare}$, it holds that $S_j, S \approx_{\epsilon'_p} \tilde{S}_j, S$. Further, by local uniformity of MShare, it holds that $U_{l'} \approx_{\epsilon_u} M_j$. Since (S, S_j), $U_{l'}$ and M_j are mutually independent we get

$$S_j, S, M_j \approx_{\epsilon'_p + \epsilon_u} \tilde{S}_j, S, U_{l'} \tag{4}$$

Note that the LHS and RHS of Inequality 3 can each be expressed as randomised functions of $(\tilde{S}_j, S, U_{l'})$. Formally, there exists randomised functions (whose randomness is independent of the input) g_2 and g_3 such that $g_2(\tilde{S}_j, S, U_{l'}) \equiv (f_j(W, \tilde{S}_j), \tilde{S}_j, S, U_{l'})$ and $g_3(\tilde{S}_j, S, U_{l'}) \equiv (f_j(\mathsf{InvExt}(U_{l'}, S), \tilde{S}_j), \tilde{S}_j, S, U_{l'})$. Now, by Lemma 6

$$g_2(\tilde{S}_j, S, M_j) \approx_{\epsilon'_p + \epsilon_u} g_2(\tilde{S}_j, S, U_{l'}) \tag{5}$$

$$g_3(\tilde{S}_j, S, M_j) \approx_{\epsilon'_p + \epsilon_u} g_3(\tilde{S}_j, S, U_{l'}) \tag{6}$$

From Inequality 3 we know,

$$g_2(\tilde{S}_j, S, U_{l'}) \approx_{2\epsilon_{ext}} g_3(\tilde{S}_j, S, U_{l'}) \tag{7}$$

Now, with applications of triangle inequality to inequalities 5, 7 and 6 and by definition of g_2 and g_3 we have

$$f_j(W, S_j), S_j, S, M_j \approx_{2(\epsilon_{ext} + \epsilon'_p + \epsilon_u)} f_j(\mathsf{InvExt}(M_j, S), S_j), S_j, S, M_j \tag{8}$$

Note that the distributions (S_j, S, M_j) are identical in both the distributions $\mathsf{Leak}^m_{\mathsf{j-1}}$ and $\mathsf{Leak}^m_{\mathsf{j}}$. The distribution W on the LHS of inequality 8 is identical to the distribution of w_j in $\mathsf{Leak}^m_{\mathsf{j}}$. The distribution $\mathsf{InvExt}(M_j, S)$ on the RHS of inequality 8 is identical to the distribution of w_j in $\mathsf{Leak}^m_{\mathsf{j-1}}$. To compute the output of the distributions $\mathsf{Leak}^m_{\mathsf{j}}$ and $\mathsf{Leak}^m_{\mathsf{j-1}}$ we invoke the following function on the above LHS and RHS respectively.

$func(a, s_j, s, m_j)$

- $(m_1, \cdots, m_n) \leftarrow \mathsf{MShare}(m | m_{\{j\}})$
- Sample $s \in_R \{0,1\}^d$
- $(s_1, \cdots, s_n) \leftarrow \mathsf{SdShare}(s | s_{\{j\}})$
- For $1 \leq i < j$, $w_i \in_r \{0,1\}^\eta$
- For $j < i \leq n$, $w_i \leftarrow \mathsf{InvExt}(m_i, s)$
- For $i \in [n] \backslash \{j\}$, define (w_i, s_i) as $share_i$ and $a_i = f_i(share_i)$.
- Set $a_j = a$.
- Output $(\{a_i\}_{i \in [|\mathcal{K}|]}, share_{\mathcal{U}})$

By application of Lemma 6 and by the definition of consistent resampling, we have

$$\mathsf{Leak}_{\mathsf{j}}^{m} \equiv func(f_j(W, S_j), S_j, S, M_j)$$

$$\approx_{2(\epsilon_{ext}+\epsilon'_p+\epsilon_u)} func(f_j(\mathsf{InvExt}(M_j, S), S_j), S_j, S, M_j) \equiv \mathsf{Leak}_{\mathsf{j-1}}^{m}$$

■ □

Claim 3. *By ϵ_p-privacy of* $(\mathsf{MShare}, \mathsf{MRec})$*, for any two messages $m \neq m'$,* $\mathsf{Leak}_{|\mathcal{K}|}^{m} \approx_{\epsilon_p} \mathsf{Leak}_{|\mathcal{K}|}^{m'}$.

Proof. Note that for any message m, the distribution $\mathsf{Leak}_{|\mathcal{K}|}^{m}$ only depends on shares of the unauthorised set $\mathcal{U}$. By privacy of MShare, for m, m'

$$\mathsf{MShare}(m)_{\mathcal{U}} \approx_{\epsilon_p} \mathsf{MShare}(m')_{\mathcal{U}}$$

Note that given $\mathsf{MShare}(m)_{\mathcal{U}}$, the output of $\mathsf{Leak}_{|\mathcal{K}|}^{m}$ can be computed by choosing $\{w_i\}_{i\in\mathcal{K}}$ and s uniformly, generating shares of s and performing the remaining computation using f_i's. Similar is the case for $\mathsf{Leak}_{|\mathcal{K}|}^{m'}$ given $\mathsf{MShare}(m')_{\mathcal{U}}$. Therefore, we have

$$\mathsf{Leak}_{|\mathcal{K}|}^{m} \approx_{\epsilon_p} \mathsf{Leak}_{|\mathcal{K}|}^{m'}$$

. ■ □

Using Claims 1,2 and 3, with applications of triangle equality, we get

$$\mathsf{Leak}_{\mathsf{0}}^{m} \approx_{2n(\epsilon_{ext}+\epsilon_u)+4|\mathcal{K}|(\epsilon_{ext}+\epsilon'_p+\epsilon_u)+\epsilon_p} \mathsf{Leak}_{\mathsf{0}}^{m'}$$

This gives the leakage error of at most $6n(\epsilon_{ext} + \epsilon'_p + \epsilon_u) + \epsilon_p$.

3.4 Parameters

Recall that $\{0,1\}^l$ is the message space.

- For an (n,t)-threshold access structure
  - We instantiate $(\mathsf{MShare}, \mathsf{MRec})$ with the (n,t)-Shamir secret sharing scheme for messages. $\{0,1\}^l$, which is perfectly private and perfectly locally uniform (that is $\epsilon_p = \epsilon_u = 0$). With this instantiation $|m_i| = l' = l$.
  - We set $\epsilon_{ext} = 2^{-\Omega(\sqrt[3]{\frac{l'}{\log l'}})}$ and instantiate the $(\eta, \tau, d, l', \epsilon_{ext})$-strong linear extractor Ext (as in Lemma 1) with $\eta = l' + \mu + \mathcal{O}(\log^3(\frac{l'}{\epsilon_{ext}}))$, $\tau = l' + \mathcal{O}(\log^3(\frac{\eta}{\epsilon_{ext}}))$ and $d = \mathcal{O}(\log^3(\frac{\eta}{\epsilon_{ext}}))$.
  - We instantiate $(\mathsf{SdShare}, \mathsf{SdRec})$ with the $(n,2)$-Shamir secret sharing scheme for messages $\{0,1\}^d$, which is perfectly private (that is $\epsilon'_p = 0$). With this instantiation we have $|s_i| = d$ (for all $i \in [n]$).
  - With the above instantiations, the size of each share output by $\mathsf{LRShare}$ to support μ bits leakage (for the leakage family $\mathcal{F}_{\mathcal{A},\mu}$) is $\eta + d = l' + \mu + \mathcal{O}(\frac{l'}{\log l'} + \log^3 \mu) = l + \mu + o(l, \mu)$. The scheme is perfectly private and $6n \cdot 2^{-\Omega(\sqrt[3]{l/\log l})}$-leakage resilient against $\mathcal{F}_{\mathcal{A},\mu}$. Therefore rate of the scheme is asymptotically 1 when $\mu = o(l)$.

- For general access structures
  - Suppose R is the function specifying the rate of the scheme $(\mathsf{MShare}, \mathsf{SdShare})$ on a given message length l. Then $l' = \frac{l}{R(l)}$. Instantiate Ext and $(\mathsf{SdShare}, \mathsf{SdRec})$ as done in the above for threshold access structures with $l' = \frac{l}{R(l)}$. With this, we get the share size of $\mathsf{LRShare} = l' + \mu + o(l', \mu)$ and hence results in rate $R(l)$ whenever $\mu = o(\frac{l}{R(l)})$.

4 Non-malleable Secret Sharing Schemes

As we mentioned in the introduction, we can get the NMSS scheme with the improved rate of 1/4, by directly instantiating the NMSS scheme of [GK18] with our LRSS scheme[10] and the rate-1/3 non-malleable code [AKO$^+$22]. Hence, our focus in this section will be on formalizing and building non-malleable randomnesss sharing schemes with the further improved rate. We begin by defining non-malleable randomness sharing, which specially gives secret sharing and non-malleability guarantees for uniform random messages. The sharing procedure outputs a (uniform random) message m along with its shares. The privacy guarantee is that, given any unautorized set of shares, the message m still looks random. The non-malleability guarantee is that, when the shares are tampered with respect to some tampering family $\mathcal{F}$, the original message m looks random, even given the recovered tampered message (using any authorized (adversarially mentioned) set for reconstruction).

Definition 3 (Non-malleable Randomness Sharing)**.** *Let* $\mathsf{RNMShare}$ *be a function such that* $\mathsf{RNMShare} : \{0,1\}^\alpha \to \{0,1\}^\ell \times (\{0,1\}^{\ell'})^n$ *is defined as* $\mathsf{RNMShare}(r) := (\mathsf{RNMShare}_1(r), \mathsf{RNMShare}_2(r)) = (m, (\mathsf{Share}_1, \cdots, \mathsf{Share}_n))$ *We say that* $\mathsf{RNMShare}$ *is a* (t,n)*-non-malleable randomness sharing with* ϵ_s*-privacy and* ϵ_{nm}*-non-malleability, message space* $\{0,1\}^\ell$*, shares from* $\{0,1\}^{\ell'}$*, for the distribution* $\mathcal{R}$ *on* $\{0,1\}^\alpha$*, and with respect to a tampering family* $\mathcal{F}$ *if it satisfies the following properties.*

1. ***Correctness.*** *For any* $T \subseteq [n]$ *with* $|T| \geq t$*, there exists a deterministic reconstruction function* $\mathsf{RNMRec} : (\{0,1\}^{\ell'})^{|T|} \to \{0,1\}^\ell$ *such that*
$$\Pr_{r \leftarrow \mathcal{R}}[\mathsf{RNMRec}(\mathsf{RNMShare}_2(r)_T) = \mathsf{RNMShare}_1(r)] = 1$$
2. ***Statistical Privacy.*** *For any unauthorized set* $U \subseteq [n]$ *such that* $|U| < t$*,*
$$(\mathsf{RNMShare}_1(\mathcal{R}), \mathsf{RNMShare}_2(\mathcal{R})_U) \approx_{\epsilon_s} (U_\ell, \mathsf{RNMShare}_2(\mathcal{R})_U))$$
3. ***Non-malleability.*** *For each* $f \in \mathcal{F}$ *and every authorized set* $T \subseteq [n]$ *containing* t *indices, there exists a simulator* $\mathsf{Sim}_{f,T}$ *over* $\{0,1\}^\ell \cup \{same^*, \perp\}$*, such that*
$$\mathsf{Tamper}_{f,T} \approx_{\epsilon_{nm}} Copy(U_\ell, \mathsf{Sim}_{f,T})$$
where $\mathsf{Tamper}_{f,T}$ *denotes the distribution* $(\mathsf{RNMShare}_1(\mathcal{R}),\ \mathsf{RNMRec}(f(\mathsf{RNMShare}_2(\mathcal{R}))_T))$ *and* $Copy(U_\ell, \mathsf{Sim}_{f,T})$ *is defined as:*
$$Copy(U_\ell, \mathsf{Sim}_{f,T}) := \begin{Bmatrix} u \leftarrow U_\ell; & \tilde{m} \leftarrow \mathsf{Sim}_{f,T} \\ \textit{Output: } (u,u), & \textit{if } \tilde{m} = same^* \\ \textit{Output: } (u,\tilde{m}), & \textit{otherwise} \end{Bmatrix}$$
where $\mathsf{Sim}_{f,T}$ *should be efficiently samplable given oracle access to* $f(.)$*.*

[10] Particularly we share the larger of the two states of the [AKO$^+$22] encoding with our LRSS scheme.

*The **rate** of this random secret sharing scheme is defined as ℓ/ℓ'.*

We specifically consider the independent tampering family, first defined in [GK18], as given below.

Independent Tampering Family $\mathcal{F}_{ind}$. Specifically, we build non-malleable randomness sharing schemes for the independent tampering family, where each share is allowed to be tampered arbitrarily, but independent of each other. Let $\mathsf{RNMShare}_2(r) = (\mathsf{Share}_1, \cdots, \mathsf{Share}_n)$. Formally, $\mathcal{F}_{ind}$ consists of functions $(f_1, \cdots, f_n)$, such that, for each $i \in [n]$, $f_i : \{0,1\}^{\ell'} \to \{0,1\}^{\ell'}$ is an arbitrary tampering function that takes as input Share_i and outputs a tampered share.
Now we proceed to build such non-malleable randomness sharing schemes with respect to $\mathcal{F}_{ind}$, achieving rate 1/2.

4.1 Building Blocks

We begin by looking at the building blocks needed for the construction. Besides our leakage resilient secret sharing scheme, and any threshold secret sharing scheme, we require non-malleable randomness encoders, defined below.

4.1.1 Non-malleable Randomness Encoders

Non-malleable randomness encoders (NMRE) were introduced in [KOS18] and give non-malleability guarantees for random messages, which we formally define below.

Definition 4 (Non-malleable Randomness Encoders [KOS18])**.** *Let* $(\mathsf{NMREnc}, \mathsf{NMRDec})$ *be s.t.* $\mathsf{NMREnc} : \{0,1\}^\alpha \to \{0,1\}^\ell \times (\{0,1\}^{\beta_1} \times \{0,1\}^{\beta_2})$ *is defined as* $\mathsf{NMREnc}(r) = (\mathsf{NMREnc}_1(r), \mathsf{NMREnc}_2(r)) = (m, (L, R))$ *and* $\mathsf{NMRDec} : \{0,1\}^{\beta_1} \times \{0,1\}^{\beta_2} \to \{0,1\}^\ell$. *We say that* $(\mathsf{NMREnc}, \mathsf{NMRDec})$ *is an* ϵ*-non-malleable randomness encoder with message space* $\{0,1\}^\ell$*, codeword space* $\{0,1\}^{\beta_1} \times \{0,1\}^{\beta_2}$*, for the distribution* $\mathcal{R}$ *over* $\{0,1\}^\alpha$*, and with respect to the* 2*-split-state tampering family* $\mathcal{F}_{split}$ *(consisting of functions* (f, g) *such that* $f : \{0,1\}^{\beta_1} \to \{0,1\}^{\beta_1}$ *and* $g : \{0,1\}^{\beta_2} \to \{0,1\}^{\beta_2}$ *are arbitrary functions acting on* L *and* R *respectively), if it satisfies the following properties.*

1. ***Correctness.*** $\Pr_{r \leftarrow \mathcal{R}}[\mathsf{NMRDec}(\mathsf{NMREnc}_2(r)) = \mathsf{NMREnc}_1(r)] = 1$.
2. ***Non-malleability.*** *For each* $(f, g) \in \mathcal{F}_{split}$*,* $\exists$ *a distribution* $\mathsf{NMRSim}_{f,g}$ *over* $\{0,1\}^\ell \cup \{same^*, \perp\}$ *such that*
$$\mathsf{NMRTamper}_{f,g} \approx_\epsilon Copy(U_\ell, \mathsf{NMRSim}_{f,g})$$
where $\mathsf{NMRTamper}_{f,g}$ *denotes the distribution* $(\mathsf{NMREnc}_1(\mathcal{R}), \mathsf{NMRDec}((f, g)(\mathsf{NMREnc}_2(\mathcal{R}))))$ *and* $Copy(U_\ell, \mathsf{NMRSim}_{f,g})$ *is defined as:*
$$Copy(U_\ell, \mathsf{NMRSim}_{f,g}) := \begin{Bmatrix} u \leftarrow U_\ell; & \tilde{m} \leftarrow \mathsf{NMRSim}_{f,g} \\ \text{Output: } (u, u), & \text{if } \tilde{m} = same^* \\ \text{Output: } (u, \tilde{m}), & \text{otherwise} \end{Bmatrix}$$
where $\mathsf{NMRSim}_{f,g}$ *should be efficiently samplable given oracle access to* $(f, g)(.)$.

We also require the following secret sharing property of the NMRE, which states that the message of an NMRE looks random, even given one of the states.

Lemma 3. *Let* $(\mathsf{NMREnc}, \mathsf{NMRDec})$ *and an* ϵ*-non-malleable randomness encoder over the message space* $\{0,1\}^\ell$*, using the distribution* $\mathcal{R}$*, and against the* 2*-split-state* $\mathcal{F}_{split}$*. Then,* $(\mathsf{NMREnc}_1(\mathcal{R}), L) \approx_{3\epsilon} (U_\ell, L)$*, where* $(L, R) \leftarrow \mathsf{NMREnc}_2(\mathcal{R})$*.*

The proof of this lemma is very similar to an analogous property satisfied of non-malleable codes, shown in [ADKO15, Lemma 6.1]. For completion, we give a full proof of the above lemma in Appendix B.

4.1.2 Instatiations of our Building Blocks

Specifically, we can now list the building blocks required for our construction.

- $(\mathsf{NMREnc}, \mathsf{NMRDec})$ be an ϵ_{nmre}-non-malleable randomness encoder, outputting messages from $\{0,1\}^\ell$ and codewords from $\{0,1\}^{\beta_1} \times \{0,1\}^{\beta_2}$, using randomness from some distribution $\mathcal{R}$, and against the split-state family $\mathcal{F}_{split}$. Further, the NMRE satifies ϵ'_p-secret sharing property (Lemma 3) that $(\mathsf{NMREnc}_1(\mathcal{R}), L) \approx_{\epsilon'_p} (U_\ell, L)$, where $(L, R) \leftarrow \mathsf{NMREnc}_2(\mathcal{R})$.

- $(\mathsf{LRShare}^2_n, \mathsf{LRRec}^2_n)$ be a $(n, 2)$-leakage resilient secret sharing scheme with ϵ_{lr}-leakage resilience against $\mathcal{F}_{2,\mu}$ taking messages from $\{0,1\}^{\beta_1}$, specifically for 2-threshold setting, i.e., the adversary can query independent leakage on $n-1$ shares, non-adaptively (upto μ bits from each share) and get one full share.

- $(\mathsf{Share}^t_n, \mathsf{Rec}^t_n)$ be any (n, t)-secret sharing scheme with ϵ_p-privacy against the (n, t)-threshold access structure, taking messages from $\{0,1\}^{\beta_2}$.

4.2 Our Construction

We now build a non-malleable randomness sharing scheme. Informally, we first use the non-malleable randomness encoder to generate a message m along with its encoding (L, R). Then, we secret share L and R using the leakage resilient and threshold secret sharing schemes respectively, to get the shares $(L_1, \cdots, L_n)$ and $(R_1, \cdots, R_n)$. Finally, we set the i-th share Share_i to be (L_i, R_i). The reconstruction procedure first reconstructs L and R, and subsequently decodes it to recover m.

$\mathsf{RNMShare}(r)$:

1. $(m, (L, R)) \leftarrow \mathsf{NMREnc}(r)$.
2. We further secret share L and R as follows:
$$(L_1, \cdots, L_n) \leftarrow \mathsf{LRShare}^2_n(L)$$
$$(R_1, \cdots, R_n) \leftarrow \mathsf{Share}^t_n(R)$$
3. For each $i \in [n]$, set $\mathsf{Share}_i = (L_i, R_i)$.
4. Output $(m, (\mathsf{Share}_1, \cdots, \mathsf{Share}_n))$.

$\mathsf{RNMRec}(\mathsf{Share}_T)$: Parse $T = \{i_1, \cdots, i_t\}$ and do the following:

1. For each $j \in T$, parse Share_j as (L_j, R_j).
2. Recover L and R as:
$$L \leftarrow \mathsf{LRRec}^2_n(L_{i_1}, L_{i_2})$$
$$R \leftarrow \mathsf{Rec}^t_n(R_{i_1}, \cdots, R_{i_t})$$
3. Output $m = \mathsf{NMRDec}(L, R)$.

Theorem 2. *Let* $(\mathsf{NMREnc}, \mathsf{NMRDec})$, $(\mathsf{LRShare}^2_n, \mathsf{LRRec}^2_n)$ *and* $(\mathsf{Share}^t_n, \mathsf{Rec}^t_n)$ *be building blocks as in Section 4.1.2. Then, the construction above gives an (n,t)-non-malleable randomness sharing scheme with $2\epsilon_p + \epsilon'_p$-privacy and $\epsilon_{nmre} + \epsilon_{lr} + \epsilon_p$-non-malleability against $\mathcal{F}_{ind}$.*
Further, we give an instantiation of the above construction in Section 4.3, which achieves an asymptotic rate of $1/2$, has a privacy error of $2^{-\Omega(\ell/\log^{\rho+1}(\ell))}$, and a non-malleablity error of $6n \cdot 2^{-\Omega(\ell/\log^{\rho+1}(\ell))}$, for any $\rho > 0$, for messages of length ℓ.

Proof. Correctness. The correctness of the scheme is straightforward from the correctness of the underlying non-malleable randomness encoder, the threshold secret sharing scheme and the leakage resilient secret sharing.

Privacy. We prove the statistical privacy using a hybrid argument. We wish to show that, for any unauthorized set $U \subseteq [n]$ with $|U| < t$, $(\mathsf{RNMShare}_1(\mathcal{R}), \mathsf{RNMShare}_2(\mathcal{R})_U) \approx_{2\epsilon_p+\epsilon'_p} (U_\ell, \mathsf{RNMShare}_2(\mathcal{R})_U))$. Let U be any arbitrary unauthorized set. Consider the following sequence of hybrids.

- Hyb_0: This hybrid corresponds to the case where the NMRE encoder is used to generate the message m.
Generate $(m, (L, R)) \leftarrow \mathsf{NMREnc}(r)$, for $r \leftarrow \mathcal{R}$. Further generate $(L_1, \cdots, L_n) \leftarrow \mathsf{LRShare}^2_n(L)$ and $(R_1, \cdots, R_n) \leftarrow \mathsf{Share}^t_n(R)$. Set $\mathsf{Share}_i = (L_i, R_i)$, for each $i \in U$ and output $(m, \{\mathsf{Share}_i\}_{i\in U})$.
- Hyb_1: Replace the shares of R in the set U with shares of an R' corresponding to a message m' output by the NMRE encoder.
Generate $(m, (L, R)) \leftarrow \mathsf{NMREnc}(r)$ and $(L', R') \leftarrow \mathsf{NMREnc}_2(r')$, for $r, r' \leftarrow \mathcal{R}$. Further generate $(L_1, \cdots, L_n) \leftarrow \mathsf{LRShare}^2_n(L)$ and $(R'_1, \cdots, R'_n) \leftarrow \mathsf{Share}^t_n(R')$. Set $\mathsf{Share}_i = (L_i, R'_i)$, for each $i \in U$ and output $(m, \{\mathsf{Share}_i\}_{i\in U})$.
- Hyb_2: Replace the m with a random message u, and use the L corresponding to m, as in Hyb_1.
Generate $u \leftarrow U_\ell$, $(L, R) \leftarrow \mathsf{NMREnc}_2(r)$ and $(L', R') \leftarrow \mathsf{NMREnc}_2(r')$, for $r, r' \leftarrow \mathcal{R}$. Further generate $(L_1, \cdots, L_n) \leftarrow \mathsf{LRShare}^2_n(L)$ and $(R'_1, \cdots, R'_n) \leftarrow \mathsf{Share}^t_n(R')$. Set $\mathsf{Share}_i = (L_i, R'_i)$, for each $i \in U$ and output $(u, \{\mathsf{Share}_i\}_{i\in U})$.
- Hyb_3: This final hybrid corresponds to the case where L and R are generated corresponding to some message m, but an independent uniform message u is output.
Generate $u \leftarrow U_\ell$, $(L, R) \leftarrow \mathsf{NMREnc}_2(r)$, for $r \leftarrow \mathcal{R}$. Further generate $(L_1, \cdots, L_n) \leftarrow$

$\mathsf{LRShare}^2_n(L)$ and $(R_1, \cdots, R_n) \leftarrow \mathsf{Share}^t_n(R)$. Set $\mathsf{Share}_i = (L_i, R_i)$, for each $i \in U$ and output $(u, \{\mathsf{Share}_i\}_{i \in U})$.

Clearly, $\mathsf{Hyb}_0 \equiv (\mathsf{RNMShare}_1(\mathcal{R}), \mathsf{RNMShare}_2(\mathcal{R})_U)$ and $\mathsf{Hyb}_3 \equiv (U_\ell, \mathsf{RNMShare}_2(\mathcal{R})_U))$. By statistical privacy of $(\mathsf{Share}^t_n, \mathsf{Rec}^t_n)$, it follows that $\mathsf{Hyb}_0 \approx_{\epsilon_p} \mathsf{Hyb}_1$ and $\mathsf{Hyb}_2 \approx_{\epsilon_p} \mathsf{Hyb}_3$. By the privacy property of NMRE, it follows that $\mathsf{Hyb}_1 \approx_{\epsilon'_p} \mathsf{Hyb}_2$. Hence, $(\mathsf{RNMShare}_1(\mathcal{R}), \mathsf{RNMShare}_2(\mathcal{R})_U) \equiv \mathsf{Hyb}_0 \approx_{2\epsilon_p + \epsilon'_p} \mathsf{Hyb}_3 \equiv (U_\ell, \mathsf{RNMShare}_2(\mathcal{R})_U))$.

Non-malleability. We prove this using a hybrid argument. We begin by describing the simulator $\mathsf{Sim}_{f_1, \cdots, f_n, T}$, for arbitrary tampering functions $f_1, \cdots, f_n \in \mathcal{F}_{ind}$ and reconstruction set $T = \{i_1, \cdots, i_t\}$.

$\mathsf{Sim}_{f_1, \cdots, f_n, T}$:

1. Let $(L^*, R^*) \leftarrow \mathsf{NMREnc}_2(r)$, for $r \leftarrow \mathcal{R}$.
2. $(L^*_1, \cdots, L^*_n) \leftarrow \mathsf{LRShare}^2_n(L^*)$
$(R^*_1, \cdots, R^*_n) \leftarrow \mathsf{Share}^t_n(R^*)$
3. Set $h = (R^*_{i_1}, \cdots, R^*_{i_{t-1}}, \widetilde{R^*_{i_1}}, \cdots, \widetilde{R^*_{i_{t-1}}}, L^*_{i_t})$, where $(\widetilde{L^*_j}, \widetilde{R^*_j}) = f_j(L^*_j, R^*_j)$, for $j = i_1, \cdots, i_{t-1}$. Define the tampering functions F_h and G_h, acting on inputs L and R respectively as:
   $F_h(L)$:
   - Pick $L_{i_1}, \cdots, L_{i_{t-1}}$ such that the reconstruction using any two shares among $L^*_{i_t}$ and $L_{i_1}, \cdots, L_{i_{t-1}}$ gives L.
   - For each $j \in \{i_1, \cdots, i_{t-1}\}$, evaluate $(\widetilde{L_j}, \widetilde{R_j}) = f_j(L_j, R^*_j)$. Then the sampling should be such that $\widetilde{R_j} = \widetilde{R^*_j}$ for each $j = i_1, \cdots, i_{t-1}$.
   - If such a sampling is not possible then output $\perp$.
     Else output $\widetilde{L} \leftarrow \mathsf{LRRec}^2_n(\widetilde{L_{i_1}}, \widetilde{L_{i_2}})$.

   $G_h(R)$:
   - Pick R_{i_t} such that it is consistent with $R^*_{i_1}, \cdots, R^*_{i_{t-1}}$ and R.
   - If such a sampling is not possible, then output $\perp$.
   - Else evaluate $(., \widetilde{R_{i_t}}) = f_{i_t}(L^*_{i_t}, R_{i_t})$.
   - Output $\widetilde{R} \leftarrow \mathsf{Rec}^t_n(\widetilde{R^*_{i_1}}, \cdots, \widetilde{R^*_{t_{t-1}}}, \widetilde{R_{i_t}})$.
4. Output $\tilde{m} \leftarrow \mathsf{NMRSim}_{F_h, G_h}$.

Now, we describe a sequence of hybrids to show that $Copy(U_\ell, \mathsf{Sim}_{f_1, \cdots, f_n, T}) \approx_{\epsilon_{nmre} + \epsilon_{lr} + \epsilon_p} \mathsf{Tamper}_{f_1, \cdots, f_n, T}$.

$\mathsf{Hyb}_1^{f_1, \cdots, f_n, T}$: This hybrid is the same as $Copy(U_\ell, \mathsf{Sim}_{f_1, \cdots, f_n, T})$, except that we **change step 4**, using $\mathsf{NMRSim}_{F_h, G_h}$, and use $\mathsf{NMRTamper}_{F_h, G_h}$ to output $m, \tilde{m}$ instead of using $Copy(U_\ell, \mathsf{NMRSim}_{F_h, G_h})$.

Claim 4. *If* $(\mathsf{NMREnc}, \mathsf{NMRDec})$ *is an* ϵ_{nmre}*-NMRE against* $\mathcal{F}_{split}$*, using the distribution* $\mathcal{R}$*, then* $Copy(U_\ell, \mathsf{Sim}_{f_1,\cdots,f_n,T}) \approx_{\epsilon_{nmre}} \mathsf{Hyb}_1^{f_1,\cdots,f_n,T}$

Proof. The proof of this claim is straightforward. Clearly, $(F_h, G_h) \in \mathcal{F}_{split}$ and hence, by the non-malleability of the NMRE, we know that $\mathsf{NMRTamper}_{F_h,G_h} \approx_{\epsilon_{nmre}} Copy(U_\ell, \mathsf{NMRSim}_{F_h,G_h})$. Thus, the reduction can generate h and forward the functions F_h, G_h to the NMRE challenger, and the response directly gives either the distribution $Copy(U_\ell, \mathsf{Sim}_{f_1,\cdots,f_n,T})$ or $\mathsf{Hyb}_1^{f_1,\cdots,f_n,T}$. Hence, the proof of the claim follows. □

$\underline{\mathsf{Hyb}_2^{f_1,\cdots,f_n,T}}$: In this hybrid, we generate $(L, R) \leftarrow \mathsf{NMREnc}_2(r)$ for $r \leftarrow \mathcal{R}$ and use the same R to generate the shares $R_1, \cdots, R_n$, used in h and as an input to the function G_h in $\mathsf{NMRTamper}_{F_h,G_h}$. The remaining steps are exactly same as in $\mathsf{Hyb}_1^{f_1,\cdots,f_n,T}$.

Claim 5. *If* $(\mathsf{Share}_n^t, \mathsf{Rec}_n^t)$ *is an* ϵ_p*-secure* (t, n)*-threshold secret sharing scheme, then* $\mathsf{Hyb}_1^{f_1,\cdots,f_n,T} \approx_{\epsilon_p} \mathsf{Hyb}_2^{f_1,\cdots,f_n,T}$.

Proof. Suppose for contradiction that the statistical distance between $\mathsf{Hyb}_1^{f_1,\cdots,f_n,T}$ and $\mathsf{Hyb}_2^{f_1,\cdots,f_n,T}$ is greater than ϵ_p. Then we describe a reduction below, to break the privacy of $(\mathsf{Share}_n^t, \mathsf{Rec}_n^t)$:

1. The reduction generates $(L^*, R^*) \leftarrow \mathsf{NMREnc}_2(r)$ and $(L, R) \leftarrow \mathsf{NMREnc}_2(r')$, for $r, r' \leftarrow \mathcal{R}$.
2. Further, generate $(L_1^*, \cdots, L_n^*) \leftarrow \mathsf{LRShare}_n^2(L^*)$.
3. Now, the reduction sends R^*, R and receives $t-1$ shares $R_{i_1}^b, \cdots, R_{i_{t-1}}^b$, from the secret sharing challenger, which correspond to either R or R^*.
4. Now, set $h = (R_{i_1}^b, \cdots, R_{i_{t-1}}^b, \widetilde{R_{i_1}^b}, \cdots, \widetilde{R_{i_{t-1}}^b}, L_{i_t}^*)$, where $(\widetilde{L_j^*}, \widetilde{R_j^b}) = f_j(L_j^*, R_j^b)$, for $j = i_1, \cdots, i_{t-1}$.
5. Now, the reduction evaluates $F_h(L) = \widetilde{L}$ and $G_h(R) = \widetilde{R}$ and outputs $(\mathsf{NMREnc}_1(r'), \mathsf{NMRDec}(\widetilde{L}, \widetilde{R}))$.

Clearly, if R^* was used by the secret sharing challenger, then the reduction output is identical to $\mathsf{Hyb}_1^{f_1,\cdots,f_n,T}$ and if R was used, then it is identical to $\mathsf{Hyb}_2^{f_1,\cdots,f_n,T}$. Hence, this breaks the privacy of $(\mathsf{Share}_n^t, \mathsf{Rec}_n^t)$. □

$\underline{\mathsf{Hyb}_3^{f_1,\cdots,f_n,T}}$: In this hybrid, all steps are exactly same as in $\mathsf{Hyb}_2^{f_1,\cdots,f_n,T}$, except that, instead of G_h reverse sampling R_{i_t}, satisfying the consistency condition, we use the same share R_{i_t} generated while setting h.

Claim 6. $\mathsf{Hyb}_2^{f_1,\cdots,f_n,T} \equiv \mathsf{Hyb}_3^{f_1,\cdots,f_n,T}$.

Proof. The reverse sampling of R_{i_t} in $\mathsf{Hyb}_2^{f_1,\cdots,f_n,T}$ uses the same R as used in generating h. Hence, G_h doesn't output $\perp$ and successfully samples R_{i_t}. This directly proves the claim. □

$\underline{\mathsf{Hyb}_4^{f_1,\cdots,f_n,T}}$: In this hybrid, we generate $(L, R) \leftarrow \mathsf{NMREnc}_2(r)$ for $r \leftarrow \mathcal{R}$ and use the same L to generate the shares $L_1, \cdots, L_n$, used in h and as an input to the function F_h in $\mathsf{NMRTamper}_{F_h,G_h}$. The remaining steps are exactly same as in $\mathsf{Hyb}_3^{f_1,\cdots,f_n,T}$.

Claim 7. *If* $(\mathsf{LRShare}_n^2, \mathsf{LRRec}_n^2)$ *is an* ϵ_{lr}*-LRSS against* $\mathcal{F}_{2,\mu}$*, then* $\mathsf{Hyb}_3^{f_1,\cdots,f_n,T} \approx_{\epsilon_{lr}} \mathsf{Hyb}_4^{f_1,\cdots,f_n,T}$.

Proof. Suppose for contradiction that the statistical distance between $\mathsf{Hyb}_3^{f_1,\cdots,f_n,T}$ and $\mathsf{Hyb}_4^{f_1,\cdots,f_n,T}$ is greater than ϵ_{lr}. Then we descirbe a reduction below, to break the leakage resilience of $(\mathsf{LRShare}_n^2, \mathsf{LRRec}_n^2)$:

1. The reduction generates $(L^*, R^*) \leftarrow \mathsf{NMREnc}_2(r)$ and $(L, R) \leftarrow \mathsf{NMREnc}_2(r')$, for $r, r' \leftarrow \mathcal{R}$.
2. Generate $R_1, \cdots, R_n \leftarrow \mathsf{Share}_n^t(R)$.
3. Now, send L, L^* as the two challenge messages to the leakage resilience challenger. Query the i_t-th full share and the leakages $g_{i_1}, \cdots, g_{i_{t-1}}$, each hardcoded with $R_{i_1}, \cdots, R_{i_{t-1}}$ respectively, defined as below. For each $j = i_1, \cdots, i_{t-1}$:
   $g_j(L_j^b)$: Evaluate $(., \widetilde{R_j}) = f_j(L_j^b, R_j)$ and output $\widetilde{R_j}$.
4. On receiving $L_{i_t}^b$ and $\widetilde{R_{i_1}}, \cdots, \widetilde{R_{i_{t-1}}}$ from the leakage resilience challenger, the reduction evaluates $(., \widetilde{R_{i_t}}) = f_{i_t}(L_{i_t}^b, R_{i_t})$ and sets $h = (R_{i_1}, \cdots, R_{i_{t-1}}, \widetilde{R_{i_1}}, \cdots, \widetilde{R_{i_{t-1}}}, L_{i_t}^b)$.
5. Now, evaluate $F_h(L) = \widetilde{L}$ and $\widetilde{R} \leftarrow \mathsf{Rec}_n^t(\widetilde{R_{i_1}}, \cdots, \widetilde{R_{i_{t-1}}}, \widetilde{R_{i_t}})$, and output $(\mathsf{NMREnc}_1(r'), \mathsf{NMRDec}(\widetilde{L}, \widetilde{R}))$.

Clearly, if the challenger picks L^*, the reduction output is identical to $\mathsf{Hyb}_3^{f_1,\cdots,f_n,T}$ and if it picks L, then it is identical to $\mathsf{Hyb}_4^{f_1,\cdots,f_n,T}$ and further, the reduction makes queries from $\mathcal{F}_{2,\mu}$, with $\mu = |R_j|$. Hence, this breaks the leakage resilience of $(\mathsf{LRShare}_n^2, \mathsf{LRRec}_n^2)$. □

$\underline{\mathsf{Hyb}_5^{f_1,\cdots,f_n,T}}$: In this hybrid, all steps are exactly same as in $\mathsf{Hyb}_4^{f_1,\cdots,f_n,T}$, except that, instead of F_h reverse sampling $L_{i_1}, \cdots, L_{i_{t-1}}$, satisfying the consistency condition, we use the same share L_j's generated while setting h.

Claim 8. $\mathsf{Hyb}_4^{f_1,\cdots,f_n,T} \equiv \mathsf{Hyb}_5^{f_1,\cdots,f_n,T}$.

Proof. The reverse sampling of $L_{i_1}, \cdots, L_{i_{t-1}}$ in $\mathsf{Hyb}_4^{f_1,\cdots,f_n,T}$ uses the same L as used in generating h. Hence, F_h doesn't output $\perp$, which directly proves the claim. □

Note that $\mathsf{Hyb}_5^{f_1,\cdots,f_n,T} \equiv \mathsf{Tamper}_{f_1,\cdots,f_n,T}$. Hence, by Claims 4, 5, 6, 7 and 8, using triangle inequality we get $Copy(U_\ell, \mathsf{Sim}_{f_1,\cdots,f_n,T}) \approx_{\epsilon_{nmre}+\epsilon_p+\epsilon_{lr}} \mathsf{Tamper}_{f_1,\cdots,f_n,T}$, which proves the non-malleability. □

4.3 Instantiation of our Scheme

We instantiate our scheme with the following primitives, where the NMRE message space is $\{0,1\}^\ell$.

- We use the following rate-1/2 NMRE from [KOS18].

  Lemma 4 (Theorem 1, [KOS18]). *There exists an NMRE for uniform messages in the two-split-state model $\mathcal{F}_{split}$, achieving a constant rate $1/(2+\zeta)$, for any $\zeta > 0$ and an error of $2^{-\Omega(\ell/\log^{\rho+1}(\ell))}$, for any $\rho > 0$.*

  Specifically, the above construction has codeword with each block of lengths: $|L| = \beta_1 = \ell(2+\zeta)$ and $|R| = \beta_2 = o(\ell)$.

- We instantiate the threshold secret sharing scheme with a perfectly private t-out-of-n Shamir secret sharing scheme for messages from $\{0,1\}^{\beta}$, which gives the shares of size $|R_i| = |R| = \beta_2 = o(\ell)$, for each $i \in [n]$.

- Further, we instantiate the LRSS $(\mathsf{LRShare}_n^2, \mathsf{LRRec}_n^2)$ against the leakage family $\mathcal{F}_{2,\mu}$ with the scheme from Section 3, with $\mu = |R_i| = \beta_2 = o(\ell)$. This gives $|L_i| = |L| + \mu + o(|L|, \mu) = \ell(2+\zeta')$, for a small $\zeta' > 0$ (ignoring the small order terms). This instantiation has a leakage error ϵ_{lr} of $6n \cdot 2^{-\Omega(\sqrt[3]{(\beta_1/\log \beta_1)})} = 6n \cdot 2^{-\Omega(\sqrt[3]{(\ell/\log \ell)})}$.

Combining these instantiations, we get a rate of $1/(2+\zeta')$, for any $\zeta' > 0$, a privacy error of $2\epsilon_p + \epsilon'_p = 2^{-\Omega(\ell/\log^{\rho+1}(\ell))}$, for any $\rho > 0$ and non-malleability error of $\epsilon_{nmre} + \epsilon_{lr} + \epsilon_p = 6n \cdot 2^{-\Omega(\ell/\log^{\rho+1}(\ell))}$, for any $\rho > 0$.

References

[ACM88] *Proceedings of the Twentieth Annual ACM Symposium on Theory of Computing*, Chicago, Illinois, 2–4 May 1988.

[ADKO15] Divesh Aggarwal, Stefan Dziembowski, Tomasz Kazana, and Maciej Obremski. Leakage-resilient non-malleable codes. In Yevgeniy Dodis and Jesper Buus Nielsen, editors, *Theory of Cryptography - 12th Theory of Cryptography Conference, TCC 2015, Warsaw, Poland, March 23-25, 2015, Proceedings, Part I*, volume 9014 of *Lecture Notes in Computer Science*, pages 398–426. Springer, 2015.

[ADN$^+$19] Divesh Aggarwal, Ivan Damgård, Jesper Buus Nielsen, Maciej Obremski, Erick Purwanto, João Ribeiro, and Mark Simkin. Stronger leakage-resilient and non-malleable secret sharing schemes for general access structures. In *Advances in Cryptology - CRYPTO 2019 - 39th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 18-22, 2019, Proceedings, Part II*, pages 510–539, 2019.

[AGV09] Adi Akavia, Shafi Goldwasser, and Vinod Vaikuntanathan. Simultaneous hardcore bits and cryptography against memory attacks. In Omer Reingold, editor, *Theory of Cryptography, 6th Theory of Cryptography Conference, TCC 2009, San Francisco, CA, USA, March 15-17, 2009. Proceedings*, volume 5444 of *Lecture Notes in Computer Science*, pages 474–495. Springer, 2009.

[AKO$^+$22] Divesh Aggarwal, Bhavana Kanukurthi, Sai Lakshmi Bhavana Obbattu, Maciej Obremski, and Sruthi Sekar. Rate one-third non-malleable codes. In *Proceedings of the Symposium on Theory of Computing, STOC 2022*, 2022.

[BBCM95] Charles H. Bennett, Gilles Brassard, Claude Crépeau, and Ueli M. Maurer. Generalized privacy amplification. *IEEE Transactions on Information Theory*, 41(6):1915–1923, 1995.

[BBR88] Charles Bennett, Gilles Brassard, and Jean-Marc Robert. Privacy amplification by public discussion. *SIAM Journal on Computing*, 17(2):210–229, 1988.

[BDIR18] Fabrice Benhamouda, Akshay Degwekar, Yuval Ishai, and Tal Rabin. On the local leakage resilience of linear secret sharing schemes. In Hovav Shacham and Alexandra Boldyreva, editors, *Advances in Cryptology - CRYPTO 2018 - 38th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 19-23, 2018, Proceedings, Part I*, volume 10991 of *Lecture Notes in Computer Science*, pages 531–561. Springer, 2018.

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

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

[BGK11] Elette Boyle, Shafi Goldwasser, and Yael Tauman Kalai. Leakage-resilient coin tossing. In David Peleg, editor, *Distributed Computing - 25th International Symposium, DISC 2011, Rome, Italy, September 20-22, 2011. Proceedings*, volume 6950 of *Lecture Notes in Computer Science*, pages 181–196. Springer, 2011.

[BGW88] Michael Ben-Or, Shafi Goldwasser, and Avi Wigderson. Completeness theorems for non-cryptographic fault-tolerant distributed computation (extended abstract). In *STOC*, pages 1–10, 1988.

[BL88] Josh Cohen Benaloh and Jerry Leichter. Generalized secret sharing and monotone functions. In *Advances in Cryptology - CRYPTO '88, 8th Annual International Cryptology Conference*. Springer, 1988.

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

[BS19] Saikrishna Badrinarayanan and Akshayaram Srinivasan. Revisiting non-malleable secret sharing. In *Advances in Cryptology - EUROCRYPT 2019 - 38th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Darmstadt, Germany, May 19-23, 2019, Proceedings, Part I*, pages 593–622, 2019.

[BT12] Mihir Bellare and Stefano Tessaro. Polynomial-time, semantically-secure encryption achieving the secrecy capacity. *CoRR*, abs/1201.3160, 2012.

[CCD88] David Chaum, Claude Crépeau, and Ivan Damgård. Multiparty unconditionally secure protocols (extended abstract). In *STOC*, pages 11–19, 1988.

[CDH$^+$00] Ran Canetti, Yevgeniy Dodis, Shai Halevi, Eyal Kushilevitz, and Amit Sahai. Exposure-resilient functions and all-or-nothing transforms. In Bart Preneel, editor, *Advances in*

Cryptology—EUROCRYPT 2000, volume 1807 of *LNCS*, pages 453–469. Springer-Verlag, 2000.

[CDS12] Mahdi Cheraghchi, Frédéric Didier, and Amin Shokrollahi. Invertible extractors and wiretap protocols. *IEEE Trans. Inf. Theory*, 58(2):1254–1274, 2012.

[CGG$^+$20] Eshan Chattopadhyay, Jesse Goodman, Vipul Goyal, Ashutosh Kumar, Xin Li, Raghu Meka, and David Zuckerman. Extractors and secret sharing against bounded collusion protocols. In *61st IEEE Annual Symposium on Foundations of Computer Science, FOCS 2020, Durham, NC, USA, November 16-19, 2020*, pages 1226–1242. IEEE, 2020.

[CKOS21] Nishanth Chandran, Bhavana Kanukurthi, Sai Lakshmi Bhavana Obbattu, and Sruthi Sekar. Adaptive extractors and their application to leakage resilient secret sharing. In *CRYPTO*. IACR, Springer, May 2021.

[DDV10] Francesco Davì, Stefan Dziembowski, and Daniele Venturi. Leakage-resilient storage. In Juan A. Garay and Roberto De Prisco, editors, *Security and Cryptography for Networks, 7th International Conference, SCN 2010, Amalfi, Italy, September 13-15, 2010. Proceedings*, volume 6280 of *Lecture Notes in Computer Science*, pages 121–137. Springer, 2010.

[DF89] Yvo Desmedt and Yair Frankel. Threshold cryptosystems. In G. Brassard, editor, *Advances in Cryptology—CRYPTO '89*, volume 435 of *LNCS*, pages 307–315. Springer-Verlag, 1990, 20–24 August 1989.

[DORS08] Yevgeniy Dodis, Rafail Ostrovsky, Leonid Reyzin, and Adam Smith. Fuzzy extractors: How to generate strong keys from biometrics and other noisy data. *SIAM Journal on Computing*, 38(1):97–139, 2008. arXiv:cs/0602007.

[DP07] Stefan Dziembowski and Krzysztof Pietrzak. Intrusion-resilient secret sharing. In *Proceedings of the 48th Annual IEEE Symposium on Foundations of Computer Science*, FOCS '07, pages 227–237, Washington, DC, USA, 2007. IEEE Computer Society.

[DSS01] Yevgeniy Dodis, Amit Sahai, and Adam Smith. On perfect and adaptive security in exposure-resilient cryptography. In Birgit Pfitzmann, editor, *Advances in Cryptology—EUROCRYPT 2001*, volume 2045 of *LNCS*, pages 301–324. Springer-Verlag, 2001.

[Fra89] Yair Frankel. A practical protocol for large group oriented networks. In Jean-Jacques Quisquater and Joos Vandewalle, editors, *Advances in Cryptology - EUROCRYPT '89, Workshop on the Theory and Application of of Cryptographic Techniques, Houthalen, Belgium, April 10-13, 1989, Proceedings*, volume 434 of *Lecture Notes in Computer Science*, pages 56–61. Springer, 1989.

[FRR$^+$10] Sebastian Faust, Tal Rabin, Leonid Reyzin, Eran Tromer, and Vinod Vaikuntanathan. Protecting circuits from leakage: the computationally-bounded and noisy cases. In Henri Gilbert, editor, *Advances in Cryptology – EUROCRYPT 2010*, pages 135–156, Berlin, Heidelberg, 2010. Springer Berlin Heidelberg.

[FV19] Antonio Faonio and Daniele Venturi. Non-malleable secret sharing in the computational setting: Adaptive tampering, noisy-leakage resilience, and improved rate. In *Advances in Cryptology - CRYPTO 2019 - 39th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 18-22, 2019, Proceedings, Part II*, pages 448–479, 2019.

[GIM$^+$16] Vipul Goyal, Yuval Ishai, Hemanta K. Maji, Amit Sahai, and Alexander A. Sherstov. Bounded-communication leakage resilience via parity-resilient circuits. In Irit Dinur, editor, *IEEE 57th Annual Symposium on Foundations of Computer Science, FOCS 2016, 9-11 October 2016, Hyatt Regency, New Brunswick, New Jersey, USA*, pages 1–10. IEEE Computer Society, 2016.

[GK18] Vipul Goyal and Ashutosh Kumar. Non-malleable secret sharing. In *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2018, Los Angeles, CA, USA, June 25-29, 2018*, pages 685–698, 2018.

[GMW87] Oded Goldreich, Silvio Micali, and Avi Wigderson. How to play any mental game or a completeness theorem for protocols with honest majority. In *Proceedings of the Nineteenth Annual ACM Symposium on Theory of Computing*, pages 218–229, New York City, 25–27 May 1987.

[GW16] Venkatesan Guruswami and Mary Wootters. Repairing reed-solomon codes. In *Proceedings of the Forty-eighth Annual ACM Symposium on Theory of Computing*, STOC '16, pages 216–226, New York, NY, USA, 2016. ACM.

[HVW21] Carmit Hazay, Muthuramakrishnan Venkitasubramaniam, and Mor Weiss. Zk-pcps from leakage-resilient secret sharing. *IACR Cryptol. ePrint Arch.*, 2021, 2021.

[ISW03] Yuval Ishai, Amit Sahai, and David Wagner. Private circuits: Securing hardware against probing attacks. In Dan Boneh, editor, *Advances in Cryptology—CRYPTO 2003*, volume 2729 of *LNCS*. Springer-Verlag, 2003.

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

[Koc96] Paul Kocher. Timing attacks on implementations of Diffie-Hellman, RSA, DSS, and other systems. In Neal Koblitz, editor, *Advances in Cryptology—CRYPTO '96*, volume 1109 of *LNCS*, pages 104–113. Springer-Verlag, 18–22 August 1996.

[KOS18] Bhavana Kanukurthi, Sai Lakshmi Bhavana Obbattu, and Sruthi Sekar. Non-malleable randomness encoders and their applications. In *Advances in Cryptology - EUROCRYPT 2018 - 37th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Tel Aviv, Israel, April 29 - May 3, 2018 Proceedings, Part III*, pages 589–617, 2018.

[LCG$^+$19] Fuchun Lin, Mahdi Cheraghchi, Venkatesan Guruswami, Reihaneh Safavi-Naini, and Huaxiong Wang. Secret sharing with binary shares. In Avrim Blum, editor, *10th*

Innovations in Theoretical Computer Science Conference, ITCS 2019, January 10-12, 2019, San Diego, California, USA, volume 124 of *LIPIcs*, pages 53:1–53:20. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2019.

[LL12] Feng-Hao Liu and Anna Lysyanskaya. Tamper and leakage resilience in the split-state model. In Reihaneh Safavi-Naini and Ran Canetti, editors, *Advances in Cryptology - CRYPTO 2012 - 32nd Annual Cryptology Conference, Santa Barbara, CA, USA, August 19-23, 2012. Proceedings*, volume 7417 of *Lecture Notes in Computer Science*, pages 517–532. Springer, 2012.

[MR04] Silvio Micali and Leonid Reyzin. Physically observable cryptography (extended abstract). In Moni Naor, editor, *First Theory of Cryptography Conference — TCC 2004*, volume 2951 of *LNCS*, pages 278–296. Springer-Verlag, February 19–21 2004.

[NS20] Jesper Buus Nielsen and Mark Simkin. Lower bounds for leakage-resilient secret sharing. In Anne Canteaut and Yuval Ishai, editors, *Advances in Cryptology - EUROCRYPT 2020 - 39th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Zagreb, Croatia, May 10-14, 2020, Proceedings, Part I*, volume 12105 of *Lecture Notes in Computer Science*, pages 556–577. Springer, 2020.

[NZ96] Noam Nisan and David Zuckerman. Randomness is linear in space. *Journal of Computer and System Sciences*, 52(1):43–53, 1996.

[Riv97] Ronald L. Rivest. All-or-nothing encryption and the package transform. In Eli Biham, editor, *Fast Software Encryption, 4th International Workshop, FSE '97, Haifa, Israel, January 20-22, 1997, Proceedings*, volume 1267 of *Lecture Notes in Computer Science*, pages 210–218. Springer, 1997.

[Rot12] Guy N. Rothblum. How to compute under ac0 leakage without secure hardware. In *Proceedings of the 32Nd Annual Cryptology Conference on Advances in Cryptology — CRYPTO 2012 - Volume 7417*, pages 552–569, Berlin, Heidelberg, 2012. Springer-Verlag.

[RRV02] Ran Raz, Omer Reingold, and Salil Vadhan. Extracting all the randomness and reducing the error in trevisan's extractors. *Journal of Computer and System Sciences*, 65(1):97–128, 2002.

[SDFY94] Alfredo De Santis, Yvo Desmedt, Yair Frankel, and Moti Yung. How to share a function securely. In Frank Thomson Leighton and Michael T. Goodrich, editors, *Proceedings of the Twenty-Sixth Annual ACM Symposium on Theory of Computing, 23-25 May 1994, Montréal, Québec, Canada*, pages 522–533. ACM, 1994.

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

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

[Tre99] Luca Trevisan. Construction of extractors using pseudo-random generators (extended abstract). In *STOC*, pages 141–148, 1999.

[Tre01] Luca Trevisan. Extractors and pseudorandom generators. *J. ACM*, 48(4):860–879, 2001.

[Vad12] Salil Vadhan. *Pseudorandomness.* Foundations and Trends in Theoretical Computer Science. Now Publishers, 2012. Available at `http://people.seas.harvard.edu/~salil/pseudorandomness/`.

A Statistical Distance and Entropy - Definitions and Lemmata

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

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

(they are said to be ε-*close* if $\mathbf{SD}\left(X_1, X_2\right) \leq \varepsilon$ and denoted by $X_1 \approx_\varepsilon X_2$).
For an event E, $\mathbf{SD}_E(A; B)$ denotes $\mathbf{SD}\left(A|E; B|E\right)$.

Entropy. The *min-entropy* of a random variable W is $\mathbf{H}_\infty(W) = -\log(\max_w \Pr[W = w])$.
For a joint distribution (W, Z), following [DORS08], we define the *(average) conditional min-entropy* of W given Z as

$$\widetilde{\mathbf{H}}_\infty(W \mid Z) = -\log(\mathop{\mathbf{E}}_{z \leftarrow Z}(2^{-\log(\max_w \Pr[W=w|Z=z])}))$$

(here the expectation is taken over z for which $\Pr[Z = z]$ is nonzero).
For any two random variable W, Z, $(W|Z)$ is said to be an (n, t')-average source if W is over $\{0,1\}^n$ and $\widetilde{\mathbf{H}}_\infty(W|Z) \geq t'$.
We require some basic properties of entropy and statistical distance, which are given by the following lemmata and propositions.

Lemma 5. *[DORS08] Let A, B, C be random variables. If B has at most 2^λ possible values, then $\widetilde{\mathbf{H}}_\infty(A \mid B) \geq \mathbf{H}_\infty(A, B) - \lambda \geq \mathbf{H}_\infty(A) - \lambda$. and, more generally, $\widetilde{\mathbf{H}}_\infty(A \mid B, C) \geq \widetilde{\mathbf{H}}_\infty(A, B \mid C) - \lambda \geq \widetilde{\mathbf{H}}_\infty(A \mid C) - \lambda$.*

Proposition 1. *For any three random variables A,B and C, $\widetilde{\mathbf{H}}_\infty(A|B) \geq \widetilde{\mathbf{H}}_\infty(A|B, C)$.*

Proof. Let A, B, C be random variables over $\mathcal{A}, \mathcal{B}, \mathcal{C}$. Then,

$$\begin{aligned}\widetilde{\mathbf{H}}_\infty(A|B) &= -\log(\mathop{\mathbf{E}}_{b \leftarrow B}(2^{-\mathbf{H}_\infty(A|B=b)})) \\ &= -\log\sum_{b \in \mathcal{B}}\max_{a \in \mathcal{A}}\Pr[A = a, B = b] \\ &= -\log\sum_{b \in \mathcal{B}}\max_{a \in \mathcal{A}}\sum_{c \in \mathcal{C}}\Pr[A = a, B = b, C = c]\end{aligned}$$

Similarly,

$$\widetilde{\mathbf{H}}_\infty(A|B,C) = -\log \sum_{b\in\mathcal{B}} \sum_{c\in\mathcal{C}} \max_{a\in\mathcal{A}} \Pr[A=a, B=b, C=c]$$

The proposition follows from the observation that for any $b \in \mathcal{B}$,

$$\sum_{c\in\mathcal{C}} \max_{a\in\mathcal{A}} \Pr[A=a, B=b, C=c] \geq \max_{a\in\mathcal{A}} \sum_{c\in\mathcal{C}} \Pr[A=a, B=b, C=c]$$

□

Lemma 6. *[Vad12] For any random variables A, B, if $A \approx_\epsilon B$, then for any function f, $f(A) \approx_\epsilon f(B)$.*

B Proof of Lemma 3

We begin by observing that to prove the secret sharing property of $(\mathsf{NMREnc}, \mathsf{NMRDec})$, i.e., $(\mathsf{NMREnc}_1(\mathcal{R}), L) \approx_{3\epsilon_{nmre}} (U_\ell, L)$, it is sufficient to prove that $(M_1, L_1) \approx_{2\epsilon_{nmre}} (M_1, L_2)$, where $M_1, (L_1, R_1) \leftarrow \mathsf{NMREnc}(r_1)$ and $(L_2, R_2) \leftarrow \mathsf{NMREnc}_2(r_2)$, for $r_1, r_2 \leftarrow \mathcal{R}$[11]. Further, non-malleability of $(\mathsf{NMREnc}, \mathsf{NMRDec})$ implies that there must exist codewords (X_0, Y) and (X_1, Y) such that $\mathsf{NMRDec}(X_0, Y) = m_0 \neq m_1 = \mathsf{NMRDec}(X_1, Y)$.

Now, suppose for contradiction that the statistical distance between (M_1, L_1) and (M_1, L_2) is greater than $2\epsilon_{nmre}$, with M_1, L_1, L_2 as described above. Then, there exists an adversary $\mathcal{A}$ (with output from $\{0,1\}$), such that

$$\Pr[\mathcal{A}(M_1, L_1) = 1] - \Pr[\mathcal{A}(M_1, L_2) = 1] > 2\epsilon_{nmre}$$

Then, we build split-state functions (f, g), which break the ϵ_{nmre}-non-malleability of $(\mathsf{NMREnc}, \mathsf{NMRDec})$.

Let $g(r) = Y$ and $f(l) = X_{\mathcal{A}(m_0, l)}$. Then, $\mathsf{NMRDec}(f(X_0), g(Y)) = m_1$ with probability $\Pr[\mathcal{A}(m_0, X_0) = 1]$ and $\mathsf{NMRDec}(f(X_1), g(Y)) = m_1$ with probability $\Pr[\mathcal{A}(m_0, X_1) = 1]$. Thus, by our assumption we get that $\Pr[\mathsf{NMRDec}(f(X_0), g(Y)) = m_1] - \Pr[\mathsf{NMRDec}(f(X_1), g(Y)) = m_1] > 2\epsilon_{nmre}$. But, the non-malleability of the NMRE directly implies that $\mathsf{NMRDec}((f(X_0), g(Y)) \approx_{\epsilon_{nmre}} \mathsf{Sim}_{f,g} \approx_{\epsilon_{nmre}} \mathsf{NMRDec}(f(X_1), g(Y))$, which gives us the contradiction! Hence, the proof of the lemma is complete.

[11] This is because if $(M_1, L_1) \approx_{2\epsilon_{nmre}} (M_1, L_2)$, then since $(M_1, L_2) \approx_{\epsilon_{nmre}} (U_\ell, L_2) \equiv (U_\ell, L_1)$, it follows that $(M_1, L_1) \approx_{3\epsilon_{nmre}} (U_\ell, L_1)$, which is exactly the secret sharing property.