# Adaptive Extractors and Their Application to Leakage Resilient Secret Sharing

Nishanth Chandran[1(✉)], Bhavana Kanukurthi[2], Sai Lakshmi Bhavana Obbattu[1], and Sruthi Sekar[3]

[1] Microsoft Research, Bangalore, India
{nichandr,t-saobb}@microsoft.com
[2] Indian Institute of Science, Research Supported by Microsoft Research Grant, Bangalore, India
bhavana@iisc.ac.in
[3] Indian Institute of Science, Research Supported by TCS Research Grant, Bangalore, India
sruthisekar@iisc.ac.in

**Abstract.** We introduce Adaptive Extractors, which unlike traditional randomness extractors, guarantee security even when an adversary obtains leakage on the source *after* observing the extractor output. We make a compelling case for the study of such extractors by demonstrating their use in obtaining adaptive leakage in secret sharing schemes.

Specifically, at FOCS 2020, Chattopadhyay, Goodman, Goyal, Kumar, Li, Meka, Zuckerman, built an adaptively secure leakage resilient secret sharing scheme (LRSS) with both rate and leakage rate being $\mathcal{O}(1/n)$, where $n$ is the number of parties. In this work, we build an adaptively secure LRSS that offers an interesting trade-off between rate, leakage rate, and the total number of shares from which an adversary can obtain leakage. As a special case, when considering $t$-out-of-$n$ secret sharing schemes for threshold $t = \alpha n$ (constant $0 < \alpha < 1$), we build a scheme with a constant rate, constant leakage rate, and allow the adversary leakage from all but $t - 1$ of the shares, while giving her the remaining $t - 1$ shares completely in the clear. (Prior to this, constant rate LRSS scheme tolerating adaptive leakage was unknown for *any* threshold.)

Finally, we show applications of our techniques to both non-malleable secret sharing and secure message transmission.

**Keywords:** Randomness extractors · Leakage resilient secret sharing · Information theoretic cryptography

## 1 Introduction

Randomness extractors [28] are a fundamental primitive in the world of theoretical computer science, which have found widespread applications in derandomization techniques, cryptography, and so on. A randomness extractor Ext is a


T. Malkin and C. Peikert (Eds.): CRYPTO 2021, LNCS 12827, pp. 595–624, 2021.
https://doi.org/10.1007/978-3-030-84252-9_20

function that takes as input an $n$-bit entropic source $W$, a uniformly random $d$-bit string $S$ (seed) and outputs $\mathsf{Ext}(W;S)$ such that $\mathsf{Ext}(W;S)$ "looks uniform" to an unbounded eavesdropper $\mathsf{Eve}$ even given the seed $S$. Unfortunately, the standard notion of extractors offers no guarantees whatsoever if the adversary $\mathsf{Eve}$ obtains some information about $W$, after observing, the output of the extractor. In this work, we address this gap.

*Does the security of extractors hold even after the adversary obtains some information on $W$, "after the fact"?*

Naturally, we have to be careful about what information $\mathsf{Eve}$ can learn about $W$ and $S$, after the fact. For instance, the function $f$, which on input $w, s$ and the extractor challenge $y$, outputs 1 if and only if $y = \mathsf{Ext}(w;s)$, is an after the fact leakage function, which can break extractor security, with high probability, with only 1 bit of leakage. Hence, one needs to define the leakage function family carefully.

In this work, we introduce the notion of *adaptive extractors* with respect to an after the fact leakage family $\mathcal{F}$. Formally, we say that an extractor is an adaptive extractor with respect to a function family $\mathcal{F}$, if for each $f \in \mathcal{F}$, an adversary cannot (statistically) distinguish $(S, f(W, \mathsf{Ext}(W;S)), \mathsf{Ext}(W;S))$ from $(S, f(W,U), U)$. Our notion of adaptive extractors can be seen as a generalization of exposure-resilient extractors introduced by Zimand [33] (Zimand's extractors allow the adversary to adaptively learn up to $n^{\delta}$ bits of the source, for some $\delta < 1$ bits; the adversary can determine which bits to query based on an arbitrary function of the extractor output.), and of the notion of adaptive non-malleable extractors introduced by Aggarwal *et al.* in [2] (where adaptive non-malleability particularly guarantees that the adversary cannot distibuish between $(S, \mathsf{Ext}(W; g(S, \mathsf{Ext}(W;S))), \mathsf{Ext}(W;S))$ and $(S, \mathsf{Ext}(W; g(S,U)), U))$. We then observe that every randomness extractor is also an adaptive extractor with respect to a leakage family depending arbitrarily on the source and the output, with some loss in parameters. We note that this observation is similar to how the authors in [2, Lemma 3.5] show that every non-malleable extractor is adaptive non-malleable, with some loss in parameters. We demonstrate that, in spite of the loss in parameters that adaptivity incurs, such extractors can be powerful. In particular, we use them to build constant-rate secret sharing schemes resilient to adaptive leakage. We now describe these contributions in greater detail.

*Secret Sharing.* Secret sharing schemes [10,30] are a fundamental cryptographic primitive and have many applications, such as in multi-party computation [7,14], and leakage-resilient circuit compilers [19,23,29]. These are cryptographic primitives that allow a dealer to distribute a secret to $n$ parties, such that only an authorized subset of parties can reconstruct the original secret and any unauthorized set of parties have no information about the underlying secret (*privacy*). For instance, in a $t-$out-of-$n$ threshold secret sharing scheme, there are $n$ parties, and any collection of $t$ ($t \leq n$) or more parties would correspond to an authorized set, and any collection of less than $t$ parties would be unauthorized.

Note that an implicit assumption is that the unauthorized set of parties has no information about secrets of the remaining shares. A rich study on leakage attacks initiated by Kocher [24] tells us that this is an idealized assumption that may not hold in practice. Such leakage can be dangerous and completely break the security of the underlying primitive[1].

*Leakage Resilient Secret Sharing (LRSS).* Dziembowski and Pietrzak in [17] introduced the problem of leakage resilience in secret sharing schemes. This problem has received much attention (for example, [1,3,9,12,15,18,20,25,27,31], [11,13]), wherein researchers have strived to improve various parameters such as its rate (defined as (message length)/(length of longest share)), leakage model as well as leakage rate (defined as (number of bits of leakage allowed)/(the size of a share)).

At a high level, in an LRSS, the adversary is allowed leakage on shares of the secret. This is captured by permitting the adversary to specify functions $\ell_1, \ell_2, \ldots,$ and receive, in response, $\ell_i(sh_i)$ (where $sh_i$ denotes the $i^{\text{th}}$ share). Informally, security of an LRSS requires that privacy should hold even given this leakage. In our work, we explore the stronger setting where the adversary specifies which share to receive leakage from, in an adaptive manner - i.e., the adversary specifies $i, \ell_i$ and upon learning $\ell_i(sh_i)$, it may make the next leakage query by specifying $j, \ell_j$. In this adaptive leakage setting[2], the construction of [13] achieved a rate of $\mathcal{O}(1/n)$ as well as a leakage rate of $\mathcal{O}(1/n)$. A consequence of this is that there currently does not exist a scheme with constant rate and leakage rate for any threshold in this strong leakage model, whereas we do know of such constructions for the non-adaptive leakage model. Our work fills this gap precisely.

### 1.1 Our Results

Our first and main result on the LRSS scheme in the adaptive leakage model is as follows. Here $n$ denotes the number of parties, $t$ denotes the threshold and $l$ denotes the message length.

**Result 1:** *We build an LRSS scheme, tolerating $\psi$ adaptive queries, each dependent on $X$ shares (with $\psi \cdot X \leq n - t + 1$) and the reveal of the remaining $t - 1$ shares, such that it achieves a rate of $(X^{\Theta(\psi X/t)})^{-1}$, while allowing $\Theta(l)$ bits of leakage per query, for threshold access structures. In particular, for a constant $X$ and $n = \Theta(t)$, this gives the first constant-rate adaptive LRSS scheme for the threshold access structure. Finally, we also generalize our constructions to the first constant-rate adaptive LRSS for general access structures.*

[1] For example, Guruswami and Wooters [22] show that Shamir's secret sharing scheme is completely insecure when the adversary gets some $t - 1$ shares and just one-bit of leakage from other shares.

[2] We note that here we only compare in an adaptive leakage model, without any joint leakage queries on multiple shares (which is called the *bounded collusion protocols* (BCP) model), for ease of expostion, and discuss the joint model in the technical section later.

Further, in the full version of our paper, we also show the following applications of our LRSS scheme.

**Result 2:** *As an application of our LRSS, we show compilers to get a leakage resilient non-malleable secret sharing (LRNMSS) scheme (which are LRSS schemes, additionally resilient to tampering attacks), and an information-theoretic secure message transmission protocol (SMT), tolerant against leakage and tampering attacks. The rates of both these schemes translate appropriately from the rate of the LRSS. In particular, for a constant LRSS, we get constant-rate schemes for both LRNMSS and SMTs.*

### 1.2 Our Techniques

We begin by describing the leakage model for LRSS and then give a technical overview of our scheme. For simplicity, we provide our technical overview for threshold access structures (which we can extend to general access structures as well). Let $t$ denote the threshold and $n$, the number of parties.

**Leakage Model.** We allow the adversary to obtain adaptive leakage on $n-(t-1)$ shares and then reveal the full shares of the remaining $t-1$ shares. Each adaptive query can be on a set of at most $X$ shares (where $X$ is some value between 1 and $t-1$), and different queries must be on sets that are disjoint from the prior queries. For the purposes of this exposition, we make the following restriction to our model: we assume that the adversary makes adaptive queries but only on a single share each time, i.e., it doesn't make any leakage query on multiple shares.

**Warm-up Construction.** To motivate our construction, we consider the following modification[3] of a construction due to Srinivasan and Vasudevan in [31, Section 3.2.1]. Take any $t$-out-of-$n$ secret sharing scheme $(\mathsf{MShare}, \mathsf{MRec})$ and then do as follows:

- Sample shares $(m_1, .., m_n)$ of the message $m$ using $\mathsf{MShare}$.
- Choose an extractor seed $s$ and split $s$ into $(sd_1, .., sd_n)$ using a $t$-out-of-$n$ secret sharing scheme.
- Now, for every $m_i$, choose an extractor source $w_i$ uniformly and compute $y_i = m_i \oplus \mathsf{Ext}(w_i; s)$.
- Finally, output the final shares $\{sh_i\}$ as $\{(w_i, y_i, sd_i)\}$.

For now, consider a weak model, where the adversary obtains only non-adaptive and independent leakage from a total of (say) $t-1$ shares, in addition to $t-1$ full shares. The hope is to show that the $t-1$ leakage queries are independent of the message shares $m_i$, following which the privacy of $\mathsf{MShare}$ can be used to get the $t-1$ full shares. One might hope to show this independence of leakage from the $m_i$'s, using the security of the extractor as follows: Pick $sd_i$ uniformly at random

[3] We note that the original construction of [31] only aimed to achieve non-adaptive security, and we consider a modification, with the aim to expand to adaptive security.

and independent of $s$; then the leakage function on $\{sh_i\}$, can be answered as an auxiliary leakage query on the source $w_i$. Once $s$ is revealed in the extractor security game, the reduction can pick the other $sd_j$ values in a consistent manner. However, this proof strategy has a flaw. For extractor security, it is important that the auxiliary leakage query on $w$ is independent of $s$; however, there is a dependence on $s$ via $y_i$. In other words, it is unclear how to prove that this construction satisfies leakage resilience even in a weak model where the adversary obtains leakage only independently and non-adaptively.

Fortunately, with adaptive extractors, we can show that this construction is secure not only in this weak model but also in a stronger model where the adversary is allowed to leak from $t-1$ shares adaptively, before receiving $t-1$ full shares. Furthermore, this construction even has a constant rate! The high-level idea of security is as follows. We wish to reduce the adaptive leakage queries on the shares to an adaptive extractor leakage query. Since the adaptive leakage query on $w_j$ cannot depend on the seed, we need to first show that the share $sd_j$ in the corresponding query is independent of the seed $s$. Indeed, using the privacy of secret sharing[4], we can show that for the first $t-1$ queries, the shares $sd_j$ in $sh_j$ can be replaced with shares of 0 (hence removing the dependence on $s$). Then, using the adaptive extractor security, we can replace the $y_j$'s (for the first $t-1$ queries) with uniform, where the leakage can be asked on the $w_j$'s. Now, the privacy of $\mathsf{MShare}$ can be invoked to get the $t-1$ full shares.

**Main Construction.** Our next goal is to leverage adaptive extractors to go beyond leaking from just $t-1$ shares. The main bottleneck is that for any subsequent leakage query (beyond $t-1$), the $sd_j$'s will reveal $s$, and hence the adaptive leakage query on subsequent $w_j$'s will no longer remain independent of the seed $s$. Thus, extractor security fails. This is the challenge we must address to achieve our main result where the adversary is allowed to obtain adaptive leakage on $n-(t-1)$ shares (in total) and reveal $t-1$ of the remaining shares.

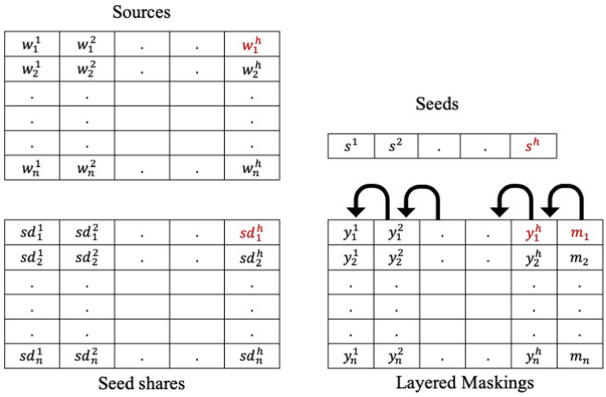
One approach to facilitating leakage from more than $t-1$ shares could be to use independent extractor seeds to extract independent random masks. Consider the following modification of the above construction: mask the share of a message $m_i$ not just with one extractor output but with many. In particular, let $y_i = m_i \oplus \mathsf{Ext}(w_i; s^1) \oplus \mathsf{Ext}(w_i; s^2) \ldots \oplus \mathsf{Ext}(w_i; s^h)$, for some parameter $h$, where $s^1 \ldots s^h$ are independent seeds. We might hope that because we are using $h$ seeds, we could hope to leak from $h(t-1)$ shares and use the security of each seed per batch of $t-1$ shares. Unfortunately, this doesn't work for the following reason: reconstruction is only possible if we recover all $h$ seeds. This means that we ultimately need to somehow share all the seeds in a manner where they can be reconstructed from $t-1$ shares. In other words, once we leak from $t-1$ shares, we can no longer argue security by leveraging any of the seeds because they can all be reconstructed from $t-1$ shares. We overcome this challenge by carefully using a multi-layered approach for both masking the message shares as well as for reconstructing the seeds.

[4] Since the leakage queries are adaptive, we require adaptive privacy of the underlying secret sharing scheme, and we show instantiations of the same.

*Construction Overview:*

1. Pick $h$ extractor seeds $s^1, \ldots, s^h$ and $hn$ extractor sources $w_1^1, \ldots w_1^h, \ldots, w_n^1, \ldots, w_n^h$.
2. Secret share each of the $h$ seeds using a $t$-out-of-$n$ secret sharing scheme to obtain shares; let the share of $s^j$ be $sd_1^j, \ldots, sd_n^j$
3. Each share $m_j$ is masked using the $h$ seeds in a layered manner as follows:
   (a) In level $h+1$: Set $y_j^{h+1} = m_j$.
   (b) For every subsequent lower level $i (i \geq 1)$, compute $x_j^i = y_j^{i+1} \oplus \mathsf{Ext}^i(w_j^i; s_i)$ and set $y_j^i = (x_j^i || sd_j^i)$. [Note that we use a different extractor per-level since the length of the extractor outputs (and the length of $y_j^i$s they mask) increase with level.]
   Finally set $Sh_j = (w_j^1, \cdots, w_j^h, y_j^1)$.
4. Output $(Sh_1, \cdots, Sh_n)$

A pictorial representation of the construction can be found in Fig. 1. In order to give an overview of the proof, we first recall that we are in a setting where each adaptive query of an adversary is a query on a single share – we can extend our results to the case of joint leakage but, for the sake of simplicity, we don't focus on that for now.




Each entry of the layered maskings matrix appropriately uses the corresponding entries of the sources, seeds and seed shares matrices. In addition, each entry $y_i^j$ $(j \leq h)$ also depends on the subsequent value i.e., $y_i^{j+1}$. Example: $y_1^h = m_1 \oplus \mathsf{Ext}^h(w_1^h; s^h) || sd_1^h$ (colored red)

**Fig. 1.** The main construction. (Color figure online)

At a high-level, the idea of the security proof is that we view the leakage queries in batches of $t-1$ queries. For the first set of $t-1$ queries, we rely on the adaptive security of the extractor outputs evaluated using seed $s^1$ and, in particular, all of these outputs can be replaced by uniform. (This also relies on the *adaptive* privacy of the secret sharing scheme, a notion we define and

instantiate.) For the second set of $t-1$ queries, we can no longer assume that $s^1$ is hidden, since we can not use the privacy of the secret sharing scheme any more. However, two things come to our rescue: first, the second batch of queries helps unmask at most $t-1$ shares of $s^2$ and therefore, adaptive extractor security on seed $s^2$ can be leveraged; second, the extractor outputs $\mathsf{Ext}(w_j^1; s^1)$ (where $j$ was a share that was leaked from in the first batch) continue to remain uniform. The reason for the latter is that all extractor sources are uniformly chosen, and our model requires a disjoint set of indices to be leaked from across batches. In short, for the first batch of queries, we use adaptive security of the extractor outputs evaluated on the first seed and, for every subsequent batch, we move to argue extractor security using the subsequent seed. Since we have $h$ independent seeds, we can do this $h$ times and therefore answer $h$ batches of queries, i.e., we can obtain leakage on $h(t-1)$ shares.

### 1.3 Related Work

We first list out some of the parameters that are relevant to LRSS schemes:

- *Rate*: This is defined as $\frac{\mathsf{messagelength}}{\mathsf{sharelength}}$.
- *Global Limit*: This refers to the total number of shares on which the leakage queries can depend on.
- *Per-query Limit*: This refers to the number of shares that a specific query can depend on.
- *Per-query Leakage Rate*: This is the ratio of the total allowable leakage from a single leakage query to the size of a share.

The problems of leakage resilient and non-malleable secret sharing have seen a flurry of activity in recent times [1,5,9,11–13,18,20,25–27,31]. Here we compare our work with only the most relevant works in this area. The only prior LRSS schemes allowing for a joint and adaptive leakage model are [13,25]. While our model allows adaptive queries on up to $n-t+1$ shares, each dependent on at most $X$ shares (where $X$ is some value between 1 and $t-1$), before fully revealing the remaining $t-1$ shares, [13] allows adaptive queries on all $n$ shares, each dependent on at most $t-1$ shares before revealing $t-1$ full shares. Both the schemes require the adaptive queries to be on disjoint sets of shares. However, our scheme/analysis offers a more fine-grained trade-off between the various parameters and allows us to obtain better results for certain settings. In particular, when we consider the instance where $X$ is constant (and $t = \alpha n$, for a constant $\alpha < 1$), we get a constant-rate adaptive LRSS achieving a constant leakage rate, while [13] gets a rate and leakage rate of $\mathcal{O}(1/n)$ each, in all instances. To put this in context, even if [13] makes independent adaptive leakage queries on all shares, their rate is $\mathcal{O}(1/n)$ and the maximum number of bits they can leak is at most a constant fraction of the size of a single share, while we can leak close $(n-t+1)$ times a constant fraction of the size of a single share!

The work of [13] also consider a variant of joint leakage, allowing overlap of the query sets, the detailed parameters of which are given in Table 1. We give a detailed comparison of the parameters achieved by the various schemes in Table 1, for the threshold setting with $t = \alpha n$ (for a constant $\alpha < 1$).

**Table 1.** LRSS prior work

| Work* | Rate | Joint Leakage | Global Limit | Per Query Limit | Leakage Rate (per query) | Adaptive | Full shares |
|---|---|---|---|---|---|---|---|
| [SV19] | 1/3 | No | $n$ | 1 | $\approx 1$ | No | Unauthorized |
| [ADN+19] | $O(1/n)$ | No | $n$ | 1 | $\approx (1-c)$ | No | Unauthorized |
| [KMS19] | $O\left(\frac{1}{poly(n)}\right)$ | Yes (overlapping) | $n$ | $\log(n)$ | $\approx \Theta\left(\frac{1}{poly(n)}\right)$ | Yes | $\log(n)$ |
| [CGG+20] | $O\left(\frac{1}{n}\right)$ | Yes | $n$ | $t-1$ | $\approx \Theta\left(\frac{1}{n}\right)$ | Yes | Unauthorized |
| [CGG+20] (threshold) | $O\left(\frac{1}{poly(n)}\right)$ | Yes (overlapping) | $n$ | $O\left(\frac{t}{\log(t)}\right)$ | $\approx \Theta\left(\frac{1}{poly(n)}\right)$ | Yes | Unauthorized |
| [CGG+20] ($n$-out of-$n$) | $O\left(\frac{1}{poly(l_{msg})}\right)$ | Yes (overlapping) | $n$ | $0.99n$ | $\approx \Theta\left(\frac{1}{poly(l_{msg})}\right)$ | Yes | Unauthorized |
| Our result | $\Theta(1)$ | Yes | $n-t+1$ | *constant* | $\approx \Theta(1)$ | Yes | Unauthorized** |

- *All works mentioned here are information-theoretic. We write all comparisons for the threshold setting with threshold $t = \alpha n$ (where $\alpha < 1$ is a constant and $n$ denotes the total number of parties).
- ** For our result, the unauthorized queries cannot overlap with the leakage queries.
- $c$ is a small constant and $l_{msg}$ is the message length.
- All schemes (except the joint overlapping schemes of [13] (threshold and $n$-out-of-$n$) actually work for general access structures.
- **Full Shares**: Number of complete shares that an adversary can see (at the end of all leakage queries, in the adaptive schemes).

***Open Problems.*** We believe that it would be interesting to explore the direction of building adaptive extractors against restricted classes of leakage families such as those captured by computational/bounded depth circuits, local functions, etc.

### 1.4 Organization of the Paper

We provide the preliminaries and definitions in Sect. 2. Then, we define and build adaptive extractors in Sect. 3. We define and build leakage resilient secret sharing schemes in Sect. 4.

## 2 Preliminaries and Definitions

### 2.1 Notation

We denote the security parameter by $\kappa$. 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 $\{1, 2, \cdots, n\}$ and $[0]$ denotes a null set. $s \in_R S$ denotes uniform sampling from set $S$. $x \leftarrow X$ denotes sampling from a probability distribution $X$. The notation $\Pr_X[x]$ denotes the probability assigned by $X$ to the value $x$. $x||y$ 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 $[n]$ :

- If $x_1, .., x_n$ are some variables or elements, then $x_S$ denotes the set $\{x_i$ such that $i \in S\}$.

– For some function $f$ outputting $n$ values $y_1, \cdots, y_n$ on input $x$, $f(x)_S$ denotes $(y_i)_{i \in S}$.
– If $T_1, .., T_n$ are sets, then $T_S$ denotes the union $\cup_{i \in S} T_i$.

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

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

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

***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 [16], we define the *(average) conditional min-entropy* of $W$ given $Z$ as

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

(here the expectation is taken over $e$ for which $\Pr[E = e]$ 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.

**Lemma 1.** *[16] Let $A, B, C$ be random variables. Then if $B$ has at most $2^\lambda$ 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, $\widetilde{\mathbf{H}}_\infty(A \mid B, C) \geq \widetilde{\mathbf{H}}_\infty(A, B \mid C) - \lambda \geq \widetilde{\mathbf{H}}_\infty(A \mid C) - \lambda$.*

**Lemma 2.** *[32] For any random variables $A, B$, if $A \approx_\epsilon B$, then for any function $f$, $f(A) \approx_\epsilon f(B)$.*

**Lemma 3.** *For any random variables $A, B$ over $\mathcal{A}$, and events $E, E'$ with non-zero probabilities,*

$$\mathbf{SD}(A \wedge E, B \wedge E') \leq |\Pr[E] - \Pr[E']| + \Pr[E'] \cdot \mathbf{SD}(A|E, B|E')$$

*where,*

$$\mathbf{SD}(A \wedge E, B \wedge E') \stackrel{def}{=} \frac{1}{2} \sum_{a \in \mathcal{A}} |\Pr[A = a \wedge E] - \Pr[B = a \wedge E']|$$

*and*

$$\mathbf{SD}(A|E, B|E') \stackrel{def}{=} \frac{1}{2} \sum_{a \in \mathcal{A}} |\Pr[A = a|E] - \Pr[B = a|E']|$$

**Lemma 4** *[4] Let $X, Y, X', Y'$ be random variables such that $\mathbf{SD}((X, Y), (X', Y')) \leq \epsilon$ and $S$ be any set such that $\Pr[Y \in S] > 0$ and $\Pr[Y' \in S] > 0$, then*

$$\mathbf{SD}(X|Y \in S, X'|Y' \in S) \leq \frac{2\epsilon}{\Pr[Y' \in S]}$$

### 2.2 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 1.** *Let $\mathcal{M}$ be a finite set of secrets, where $|\mathcal{M}| \geq 2$ . Let $[n]$ be a set of identities (indices) of $n$ parties. A sharing function* $\mathsf{Share} : \mathcal{M} \to (\{0,1\}^l)^n$ *is a* $(\mathcal{A}, n, \epsilon_s)$- ***secret sharing scheme*** *with respect to a monotone access structure*[5] *$\mathcal{A}$ 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* $\mathsf{Rec} : (\{0,1\}^l)^{|T|} \to \mathcal{M}$ *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 $m, m' \in \mathcal{M}$,*
$$\Delta((\mathsf{Share}(m))_U; (\mathsf{Share}(m'))_U) \leq \epsilon_s$$

*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 at least $t$.*
***Rate*** *of a secret sharing scheme is defined as $\frac{\text{message size}}{\text{share size}}$ (which would be equal to $\frac{\log |\mathcal{M}|}{l}$).*

We now study a stronger privacy requirement, *adaptive privacy* (introduced by Bellare and Rogaway [6][6]).

#### 2.2.1 Adaptive Privacy

Statistical privacy captures privacy against any non-adaptively chosen unauthorized set $U$. *Adaptive privacy* preserves privacy even when the choice of $U$ to be adaptive, which means the following. Let $U = \{i_1, .., i_q\}$. We say $i_j$ is chosen adaptively, if its choice depended on $\{share_j\}_{j \in \{i_1, .., i_{j-1}\}}$. The choice of which share to query next depends on all the previously observed shares. We give the formal definition below.

[5] $\mathcal{A}$ 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.

[6] In [6], the authors refer to adaptive privacy as privacy against dynamic adversaries.

We say a $(\mathcal{A}, n, \epsilon_s)$-secret sharing scheme satisfies adaptive privacy with error $\epsilon_{adp}$ if, for any distinguisher $\mathcal{D}$, the advantage in the following game is at most $\epsilon_{adp}$.

$\mathsf{Game}_{\mathsf{Ad-Privacy}}$ : For any arbitrary distinct messages $m_0, m_1 \in \mathcal{M}$

1. $(share_1, \cdots, share_n) \leftarrow \mathsf{Share}(m_b)$ where $b \in_R \{0, 1\}$
2. For $j = 1$ to $q$ [7]
   - $\mathcal{D}$ queries on a distinct index $i_j$ (such that $i_{[j]} \notin \mathcal{A}$) and receives $share_{i_j}$
3. $\mathcal{D}$ outputs the guess $b'$ for $b$ and wins if $b = b'$

While generally, any secret sharing scheme may not be adaptively private, we can show that for the threshold setting, the scheme of [30] and for the general access structures, the scheme of [8] are both adaptively private (which is proved in the full version of our paper). We use them to instantiate our schemes.

***Consistent Re-sampling.*** For any $(\mathcal{A}, n, \epsilon_s)$-secret sharing scheme ($\mathsf{Share}$, $\mathsf{Rec}$), for any message $m$ and a subset $\mathcal{L} \subseteq [n]$, when we say "$(sh_1, .., sh_n) \leftarrow \mathsf{Share}(m)$ *consistent with* $sh^*_{\mathcal{L}}$ on $\mathcal{L}$" or "$(sh_1, .., sh_n) \leftarrow \mathsf{Share}(m|sh^*_{\mathcal{L}})$" 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).

We require the following consistent re-sampling feature[8], which informally states that for any $(\mathcal{A}, n, \epsilon_s)$-secret sharing scheme and any message $m$, the distribution of shares which are re-sampled as shares of $m$, conditioned on some set $T$ of shares (which are also generated as shares of $m$) chosen adaptively, is identical to the distribution of shares of $m$ generated directly.

**Lemma 5.** *For any $(\mathcal{A}, n, \epsilon_s)$-secret sharing scheme ($\mathsf{Share}, \mathsf{Rec}$) and for any message $m$, the following two distributions are identical.*

$\mathcal{D}_1$ :

- $(sh'_1, .., sh'_n) \leftarrow \mathsf{Share}(m)$
- $(sh_1, .., sh_n) \leftarrow \mathsf{Share}(m|sh'_T)$
- *Output* $(sh_1, .., sh_n)$

$\mathcal{D}_2$ :

- $(sh_1, .., sh_n) \leftarrow \mathsf{Share}(m)$
- *Output* $(sh_1, .., sh_n)$

*Here, $T \subseteq [N]$ can be any subset chosen as: every index (except the first) depends arbitrarily on the shares corresponding to all the previous indices.*

We give a full proof of the above lemma in the full version of our paper.

[7] $q$ is arbitrary and chosen by $\mathcal{D}$. It need not be chosen a-priori. We only use it to denote the total number queries made by $\mathcal{D}$

[8] Note that we only use the re-sampling in proofs and do not require the procedure to be efficient.

## 3 Adaptive Extractors

Extractors (introduced by Nissan and Zuckerman [28]) output a near uniform string $y$, from a source $w$ that only has min-entropy, using a short uniform string $s$, called the *seed*, as a catalyst. Average-case extractors are extractors whose output remains close to uniform, even given the seed and some auxiliary information (or leakage) about the source (independent of the seed), as long as the source has enough average entropy given this leakage. We give their formal definition below.

**Definition 2.** *[16] Let* $\mathsf{Ext} : \{0,1\}^\eta \times \{0,1\}^d \to \{0,1\}^l$ *be a polynomial time computable function. We say that* $\mathsf{Ext}$ *is an efficient average-case* $(\eta, \mu, d, l, \epsilon)$*-strong extractor if for all pairs of random variables* $(W, Z)$ *such that* $W$ *is an* $\eta$*-bit string satisfying* $\widetilde{\mathbf{H}}_\infty(W|Z) \geq \mu$*, we have*

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

### 3.1 Definition

Average-case extractors, unfortunately, provide no guarantees on the extractor output being uniform when an adversary can obtain an 'adaptive' leakage on the source, that is *dependent on the extractor output* and the seed. This is not surprising, as if an adversary can obtain *arbitrary adaptive leakage* on the source, then we cannot hope for the extractor output to remain uniform. For example, given $y = \mathsf{Ext}(w, s)$, an adversary can distinguish the extractor output from uniform with high probability by querying a single bit of auxiliary information that tells her whether $\mathsf{Ext}(w, s) = y$. However, as we will see later, in many applications, the adaptive leakage that the adversary obtains comes from a specific function family. Hence, by carefully defining this function family, we show how to obtain useful notions of extractors that guarantee security even in the presence of an adaptive auxiliary information. We introduce and call this notion *adaptive extractors* and now proceed to formally define them.

**Definition 3.** *An* $(\eta, \mu, d, l, \epsilon)$*- extractor* $\mathsf{Ext}$ *is said to be an* $(\mathcal{F}, \delta)$*-**adaptive extractor** if for all pairs of random variables* $(W, Z)$ *such that* $W$ *is an* $\eta$*-bit string satisfying* $\widetilde{\mathbf{H}}_\infty(W|Z) \geq \mu$*, and any function* $f$ *in the function family* $\mathcal{F}$*, it holds that*

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

*We call* $\delta$*, the **adaptive error** of the extractor.*

### 3.2 Construction

*Generic Relation.* We show that every extractor is in fact an adaptive extractor for the family of leakage functions where the adaptive leakage depends only on

the source and the extractor output (i.e., it doesn't depend on the seed except via the extractor output), with some loss in security. This loss, in fact, depends only on the number of bits of the extractor output that the adaptive leakage function depends on. For ease of exposition, we omit auxiliary information $z$ that depends only on the source (but not on the extractor output or seed) from the notation below. We now explicitly define this family below:

$$\mathcal{F}_{a,\zeta} \subseteq \{f' : \{0,1\}^{\eta} \times \{0,1\}^{l} \to \{0,1\}^{\zeta}\}$$

such that for every $f' \in \mathcal{F}_{a,\zeta}$ there exists two functions $f : \{0,1\}^{l} \to \{0,1\}^{a}$ and

$$g : \{0,1\}^{\eta+a} \to \{0,1\}^{\zeta} \text{ such that } \forall w, y,\ f'(w,y) = g(w, f(y))\}$$

Here, '$\zeta$' denotes the number of bits of adaptive leakage and '$a$' denotes the number of bits of the extractor output (or the uniform string) that the adaptive leakage depends on. This is captured by requiring that every function $f'$ has an equivalent representation in terms of some $g$ and $f$ such that $f'(w,y) = g(w, f(y))$ where $f$'s output is only $a$ bits long. $w$ and $y$ should be interpreted as the source and the extractor output (or the uniform string) respectively.

The following theorem shows that any $(\eta, \mu, d, l, \epsilon)$- average case extractor can be shown to be adaptive secure against the above family $\mathcal{F}_{a,\zeta}$, with an adaptive error of $2^{a+2}\epsilon$. Informally, we can reduce the adaptive security to the extractor security (as in Definition 2) in the following way: to answer the adaptive leakage query, the reduction makes a guess, $v$, for the extractor challenge dependent value $f(y_b)$ (where, $y_b$ is the extractor challenge), which is of $a$-bits, and gets the leakage $g(w,v)$ from the source. Now, it gets the challenge $y_b$ from the extractor challenger and if $f(y_b)$ matches the guess $v$, then the reduction can successfully simulate the challenge and the adaptive leakage response, else it cannot proceed (and aborts). Hence, the winning probability in the extractor game is the probability of a correct guess ($2^{-a}$), multiplied with the winning probability of the adaptive extractor adversary. We formalize this proof in the theorem below.

**Theorem 1.** *Every $(\eta, \mu, d, l, \epsilon)$- average case extractor* $\mathsf{Ext}$ *is an $(\eta, \mu + \zeta, d, l, \epsilon)$- extractor that is $(\mathcal{F}_{a,\zeta}, 2^{a+2}\epsilon)$-adaptive, for any $\mu + \zeta \leq \eta$ and $a \leq l$.*

*Proof.* For simplicity, we omit the auxiliary information $Z$, that depends only on the source (and not on the extractor output). Let $W$ be the source of $\eta$ bits, such that $\mathbf{H}_{\infty}(W) \geq \mu + \zeta$. Consider $f' \in \mathcal{F}_{a,\zeta}$, with the corresponding functions $(f, g)$ (recall $f'(w,y) = g(w, f(y))$, where $f$ outputs $a$ bits and $g$ outputs $\zeta$ bits). To prove adaptive security (Definition 3), we need to show that:

$$U_d, f'(W,Y), Y \approx_{2^{a+2}\epsilon} U_d, f'(W,U_l), U_l,$$

where $Y$ is the random variable $\mathsf{Ext}(W; U_d)$. Expanding the description of $f'$, this gives:

$$U_d, g(W, f(Y)), Y \approx_{2^{a+2}\epsilon} U_d, g(W, f(U_l)), U_l$$

To prove this, we consider the following two sets $\mathcal{B} = \{b : \Pr[f(Y) = b] > 0\}$ and $\mathcal{A} = \{0,1\}^{d+\zeta+l}$. For each $b \in \mathcal{B}$, we begin by using the statistical distance Lemma 3 with random variables $A, B$ and events $E, E'$ set as $(U_d, g(W, f(Y)), Y)$, $(U_d, g(W, f(U_l)), U_l)$, $f(Y) = b$ and $f(U_l) = b$, respectively. By use of law of total probability and Lemma 3, we get:

$$\begin{aligned}
&\mathbf{SD}((U_d, g(W, f(Y)), Y),\ (U_d, g(W, f(U_l)), U_l)) \\
&\quad \leq \Pr[f(U_l) \notin \mathcal{B}] + \sum_{b \in \mathcal{B}} \mathbf{SD}\left(A \wedge E, B \wedge E'\right) \\
&\quad \leq \Pr[f(U_l) \notin \mathcal{B}] + \sum_{b \in \mathcal{B}} ((|\Pr[E] - \Pr[E']|) + \Pr[E'] \cdot \mathbf{SD}\left(A|E, B|E'\right))
\end{aligned}$$

But now, note that, by extractor security, since $Y \approx_\epsilon U_l$, by applying Lemma 2, we have $f(Y) \approx_\epsilon f(U_l)$. Further, by the definition of statistical distance, we have that, for each $b \in \mathcal{B}$, $|\Pr[f(Y) = b] - \Pr[f(U_l) = b]| \leq \epsilon$ and $\Pr[f(U_l) \notin \mathcal{B}] \leq \epsilon$ (since $\Pr[f(Y) \notin \mathcal{B}] = 0$]). Applying this to above inequality, we get:

$$\begin{aligned}
&\mathbf{SD}((U_d, g(W, f(Y)), Y),\ (U_d, g(W, f(U_l)), U_l)) \\
&\quad \leq \epsilon + \sum_{b \in \mathcal{B}} (\epsilon + \Pr[E'] \cdot \mathbf{SD}\left(A|E, B|E'\right)) \\
&\quad = (|\mathcal{B}| + 1)\epsilon + \sum_{b \in \mathcal{B}} \Pr[E'] \cdot \mathbf{SD}(A|E,\ B|E')
\end{aligned}$$

Finally, we apply the statistical distance Lemma 4 on the random variables $(A, f(Y))$ and $(B, f(U_l))$ with set $S = \{b\}$. Note that, given events $E$ and $E'$ the value of $f(Y)$ and $f(U_l)$ are fixed to a $b$, which means the leakage $g(W, b)$ is only a leakage on $W$. Thus, we can use extractor security to get: $(U_d, g(W, b), Y) \approx_\epsilon (U_d, g(W, b), U_l)$. Hence, applying this to the above inequality, we get:

$$\begin{aligned}
&\mathbf{SD}((U_d, g(W, f(Y)), Y),\ (U_d, g(W, f(U_l)), U_l)) \\
&\quad \leq (|\mathcal{B}| + 1)\epsilon + \sum_{b \in \mathcal{B}} \Pr[E'] \cdot \frac{2\epsilon}{\Pr[f(U_l) = b]} \\
&\quad \leq 4|\mathcal{B}|\epsilon \leq 2^{a+2}\epsilon
\end{aligned}$$

*Concrete Instantiation.* We show that the extractor due to Guruswami *et al.* [21] is an adaptive extractor even when the leakage depends on the entire extractor output. We state the result from [21] below.

**Lemma 6.** *[21] For every constant $\nu > 0$ all integers $\eta \geq \mu$ and all $\epsilon \geq 0$, there is an explicit (efficient) $(\eta, \mu, d, l, \epsilon)-$strong extractor with $l = (1 - \nu)\mu - \mathcal{O}(\log(\eta) + \log(\frac{1}{\epsilon}))$ and $d \leq \mathcal{O}(\log(\eta) + \log(\frac{1}{\epsilon}))$.*

Let $\mathsf{Full}_\zeta$ $(= \mathcal{F}_{l,\zeta})$, denote the leakage function family which computes leakage (of size $\zeta$) dependent on the entire extractor output and the source. The following lemma shows that one can appropriately set the parameters of the [21] extractor to get negligible error, while extracting a constant fraction of the bits from the source, and while adaptively leaking a constant fraction of bits from it.

**Lemma 7.** *For all positive integers $l, \zeta$, every constant $\nu > 1$ and $\epsilon \geq 0$, there is an explicit (efficient) $(\eta, \mu+\zeta, d, l, \epsilon)-$extractor that is $(\mathsf{Full}_\zeta, \delta)$-adaptive with $d = \mathcal{O}(\log(\frac{\eta}{\epsilon}))$, $\mu = \nu l + \mathcal{O}(\log(\frac{\eta}{\epsilon}))$, any $\eta \geq \mu + \zeta$ and $\delta = \epsilon \cdot 2^{l+2}$.*
*On further implication, for any $c > 1$, there exists constants $\alpha, \beta$ such that $d \leq \alpha l$, $\mu \leq \beta l$, $\eta \geq \beta l + \zeta$, $\epsilon = 2^{-cl}$ and $\delta = 2^{(1-c)l+2}$ when $l = \omega(\log \eta)$.*

*Proof.* The proof of the first part of the lemma follows directly from Theorem 1 and Lemma 6 and the further implication can be obtained by simple substitution.

Further, we use the following generalization of adaptive extractors: for an adaptive extractor $\mathsf{Ext}$, if we consider $k$ independent sources $W_1, \cdots, W_k$ and a single seed $S$, all the extractor outputs $(\mathsf{Ext}(W_i; S))_{i\in[k]}$ look uniform, even given adaptive leakage on each $W_i$, dependent on not just $\mathsf{Ext}(W_i; S)$ (or uniform), but also all the prior extractor outputs and adaptive leakages (queried before $i$). As the sources are independent, this lemma can be proved using a simple hybrid argument (the detailed proof is given in our full version).

**Lemma 8.** *Let $k$ be an arbitrary positive integer, $W_1, \cdots, W_k$ be $k$ independent $(\eta, \mu + \zeta)$ sources and $S$ be the uniform distribution on $\{0,1\}^d$. Let $\mathsf{Ext}$ be an $(\eta, \mu + \zeta, d, l, \delta')$-extractor that is $(\mathsf{Full}_\zeta, \delta)$-adaptive. For each $i \in [k]$, let $E_i^0$ denotes $\mathsf{Ext}(W_i; S)$, $E_i^1$ denotes uniform distribution on $\{0,1\}^l$. For $b \in \{0,1\}$, we define $\mathsf{AdLeak}^b$ as follows. Then for any stateful distinguisher $\mathcal{D}'$ we have $\mathsf{AdLeak}^0 \approx_{k\delta} \mathsf{AdLeak}^1$.*
$\mathsf{AdLeak}^b$ :

- *Let $Tr$ and $\mathcal{S}$ be a null string and null set respectively.*
- *For upto $k$ times*
  - *$(j, g_j) \leftarrow \mathcal{D}'(Tr)$ where $j \in [k]\backslash\mathcal{S}$ and $g_j : \{0,1\}^{\eta+l} \rightarrow \{0,1\}^\zeta$.*
  - *Append $(j, g_j, g_j(w_j, E_j^b), E_j^b)$ to $Tr$.*
  - *Add $j$ to $\mathcal{S}$.*
- *Output $Tr$.*

## 4 Leakage Resilient Secret Sharing

Leakage-resilience of a secret sharing scheme is defined specific to a leakage model/ leakage family. We begin by formally defining leakage-resilience and then describe the leakage model.

**Definition 4.** *An $(\mathcal{A}, n, \epsilon_s)$-secret sharing scheme is said to be an $(\mathcal{A}, n, \epsilon_s, \epsilon_l)$-**leakage resilient secret sharing scheme** against a leakage family $\mathcal{F}$ if for all functions $f \in \mathcal{F}$ and for any two messages $m, m'$, $\mathbf{SD}\left(f(Share(m)), f(Share(m'))\right) \leq \epsilon_l$.*

### 4.1 Leakage Models

We consider two leakage models in this paper. For now, we restrict our discussion to an $(n, t)$-threshold access structure.

- **Adaptive Leakage and Reveal Model**: The adversary can adaptively obtain leakage on individual shares for any $n-t+1$ shares. After this, he can additionally even get all the remaining $t-1$ shares in their entirety.
- **Joint Leakage and Reveal Model**: The adversary can ask any number of joint leakage queries on disjoint sets of size $X$ (a parameter). After this, he can additionally get any (at most $t-1$) of the remaining shares in their entirety. While this model completely subsumes the adaptive leakage and reveal model, the amount of leakage per share supported in the latter would be lesser.

We provide a formal description of the adaptive leakage and reveal model and the joint leakage and reveal model in Sect. 4.1.1 and Sect. 4.5 respectively. We give a construction that is leakage resilient with respect to both these models in Sect. 4.2. We prove leakage resilience of this scheme in the adaptive leakage and reveal model in Sect. 4.3. We provide a proof sketch of leakage resilience in the joint adaptive and reveal model in Sect. 4.5.2.

#### 4.1.1 Adaptive Leakage and Reveal Model $\mathcal{F}_{leak}^{\psi,\tau}$

The model allows for leakage on individual shares and then also reveals at most $t-1$ of the remaining shares in clear. We have two parameters in the model $\tau$ and $\psi$ where $\tau$ denotes the amount of leakage provided in each leakage query and $\psi$ captures the maximum number of leakage queries allowed. We allow $\psi$ ranging from 1 to $n-t+1$. Though we allow $\psi$ to be $n-t+1$, we have it as an explicit parameter because lower $\psi$ would imply a weaker leakage model and possibly have better constructions. In fact, our multi-layered construction in Sect. 4.2 becomes compact (and offers better rate) as $\psi$ decreases.

$\mathsf{Leak}_{Share}^{m}$:

- Initialize $Z$ to be a null string and $\mathcal{S}$ to be a null set.
- $(Sh_1, \cdots, Sh_n) \leftarrow Share(m)$
- **Leakage Phase:**
  For upto $\psi$ times
  - $(j, f_j) \leftarrow \mathcal{D}(Z)$ where $f_j : \{0,1\}^{\gamma} \rightarrow \{0,1\}^{\tau}$
  - If $j \in [n] \backslash \mathcal{S}$, add $j$ to $\mathcal{S}$ and append $(j, f_j, f_j(Sh_j))$ to $Z$
- **Reveal phase**

  For upto $t-1$ times
  - $j \leftarrow \mathcal{D}(Z)$
  - If $j \in [n] \backslash \mathcal{S}$, append $(j, Sh_j)$ to $Z$
- $\mathcal{D}$ updates $Z$ with any relevant state information.
- Output $Z$.

**Fig. 2.** LRSS definition- $\mathsf{Leak}_{Share}^{m}$ distribution

Let $(Share, Rec)$ (where $Share : \{0,1\}^l \to (\{0,1\}^\gamma)^n$) be a $t$-out-of-$n$ secret sharing scheme. We formalize leakage obtained in this model on shares of a message $m$ as $\mathsf{Leak}^m_{Share}$ in Fig. 2, where an arbitrary stateful distinguisher $\mathcal{D}$ makes the queries. For any two messages $m$ and $m'$, we require $\mathsf{Leak}^m_{Share} \approx_{\epsilon_{lr}} \mathsf{Leak}^{m'}_{Share}$, for $(Share, Rec)$ to be $\epsilon_{lr}$ leakage resilient against the adaptive leakage and reveal model.

### 4.2 LRSS Construction for the Adaptive Leakage and Reveal Model

We refer the reader to the Introduction (Sect. 1.2) for a high-level overview of the construction and proof. We proceed to describe the construction in detail in Fig. 3 and prove its security in Sect. 4.3.

Let $n$ be the number of parties and $t$ be the reconstruction threshold. Let $h > 0$ be a parameter guaranteed to be less than $\lceil n/(t-1) \rceil$.

**Building Blocks.** Let $(\mathsf{MShare}, \mathsf{MRec})$ be an $((n,t), \varepsilon, \epsilon)$-adaptive secret sharing scheme for messages in $\{0,1\}^l$ with share space being $\{0,1\}^{l'}$. For $i \in [h]$, let $(\mathsf{SdShare}^i, \mathsf{SdRec}^i)$ be an $((n,t), \varepsilon'_i, \epsilon'_i)$-adaptive secret sharing scheme for messages in $\{0,1\}^{d_i}$ with share space being $\{0,1\}^{d'_i}$. For $i \in [h]$, let $\mathsf{Ext}^i$ be an $(\eta_i, \mu_i + \tau, d_i, \ell_i, \delta'_i)$-extractor that is $(\mathsf{Full}_\tau, \delta_i)$-adaptive. We set $\ell_1 = l'$ and for $i \in [h] \backslash \{1\}$ we set $l_i = l_{i-1} + d'_{i-1}$.

$\mathsf{Share}^h(m)$:

- $(m_1, \cdots, m_n) \leftarrow \mathsf{MShare}(m)$.
- For $i \in [h]$, pick seeds $s^i \in_R \{0,1\}^{d_i}$ and compute their shares $(sd^i_1, \cdots, sd^i_n) \leftarrow \mathsf{SdShare}^i(s^i)$.
- For $i \in [h]$ and $j \in [n]$, pick sources $w^i_j \in_R \{0,1\}^{\eta_i}$.
- For $j \in [n]$:
  - Define $y^{h+1}_j = m_j$.
  - For $i \leftarrow h$ to 1, compute $x^i_j = y^{i+1}_j \oplus \mathsf{Ext}^i(w^i_j; s^i)$ and $y^i_j = (x^i_j || sd^i_j)$.
- For $j \in [n]$, define $Sh_j = (w^1_j, \cdots, w^h_j, y^1_j)$.
- Output $(Sh_1, \cdots, Sh_n)$.

$\mathsf{Rec}^h(Sh_T)$ : (where $T$ is the reconstruction set)

- For $j \in T$, parse $Sh_j$ as $(w^1_j, \cdots, w^h_j, y^1_j)$, where $y^1_j = x^1_j || sd^1_j$.
- For $i \leftarrow 1$ to $h$:
  - $s^i = \mathsf{SdRec}^i(sd^i_T)$.
  - For each $j \in T$, $y^{i+1}_j = x^i_j \oplus \mathsf{Ext}^i(w^i_j; s^i)$. For each $i \in [h-1]$, parse $y^{i+1}_j$ as $x^{i+1}_j || sd^{i+1}_j$.
- Parse $y^{h+1}_j$ as $m_j$. Recover $m = \mathsf{MRec}(m_T)$.
- Output $m$.

**Fig. 3.** LRSS construction

### 4.3 Proof of Leakage Resilience in the Adaptive Leakage and Reveal Model

**Theorem 2.** *For any $\psi \leq n-t+1$ and $l, \tau > 0$, $(\mathsf{Share}^h, \mathsf{Rec}^h)$ is an $((n,t),\varepsilon)$-secret sharing scheme for $l$ bit messages and is $2(\epsilon + h(\epsilon' + (t-1)\delta))$-leakage resilient in the Adaptive Leakage and Reveal model $\mathcal{F}_{leak}^{\psi,\tau}$ where $h = \lceil \psi/(t-1) \rceil$. Further, there exists an instantiation of the scheme with rate is $(2^{\Theta(h)} + h\tau/l)^{-1}$. When $\tau = \Theta(l)$ and either $n = \Theta(t)$ or $h$ is a constant, the scheme achieves constant rate and constant leakage rate asymptotically.*

*Proof.* The correctness of the scheme follows directly from the correctness of underlying extractors and secret sharing schemes. The (adaptive) privacy of the scheme is directly implied by the leakage resilience (against the adaptive leakage and reveal model).

***Leakage Resilience.*** For any message $m$ we define the following the sequence of hybrids. In these hybrids we assume that $\mathcal{D}$ always asks legitimate queries as per the model and won't write explicit checks for legitimacy (for example, we assume that $\mathcal{D}$ doesn't ask leakage on same share twice).

We analyze the leakage queries made by $\mathcal{D}$ as bunches of $(t-1)$ queries. We now introduce some useful notation. Let $\mathcal{S}_1, \cdots, \mathcal{S}_h$ denote the sets of indices queried by $\mathcal{D}$, where $\mathcal{S}_i$ contains the indices queried by $\mathcal{D}$ from the $((i-1)(t-1)+1)^{th}$ query to $i(t-1)^{th}$ leakage queries (i.e., $\mathcal{S}_1$ contains the first $t-1$ queries, $S_2$ the next $t-1$ queries and so on). For $i \in [h]$, we use $\mathcal{S}_{[i]}$ to denote $\bigcup_{j=1}^{i} \mathcal{S}_j$, which captures the set of indices queried in the first $i(t-1)$ leakage queries. For $i \in [h]$, let $Z_{[i]}$ denotes the set of leakage queries and the corresponding responses to the first $i(t-1)$ leakage queries. $Z_{[h+1]}$ denotes $Z_{[h]}$ together with the final reveal queries as well as any relevant state information. We prove leakage resilience using a hybrid argument, with the following sequence of hybrids, $\mathsf{LeakB}_0^m$, $\{\mathsf{LeakA}_q^m, \mathsf{LeakB}_q^m\}_{q\in[h]}$ and $\mathsf{LeakC}^m$. The order of the hybrids is $\mathsf{LeakB}_0^m, \mathsf{LeakA}_1^m, \mathsf{LeakB}_1^m, \cdots, \mathsf{LeakA}_h^m, \mathsf{LeakB}_h^m, \mathsf{LeakC}^m$, where we will show that $\mathsf{LeakC}^m$ is independent of $m$, and $\mathsf{LeakB}_0^m$ will correspond to the distribution $\mathsf{Leak}_{\mathsf{Share}^h}^m$. This will allow us to show that $\mathsf{Leak}_{\mathsf{Share}^h}^m$ is indistinguishable from $\mathsf{Leak}_{\mathsf{Share}^h}^{m'}$. We begin by giving an informal description of these hybrids.

$\mathsf{LeakA}_q^m$*:* We start with $q = 1$. $\mathsf{LeakA}_1^m$ follows the actual leakage game i.e., $\overline{\mathsf{Leak}_{Share^h}^m} (\equiv \mathsf{LeakB}_0^m)$ except for the following change: we replace the shares $sd_j^1$, for each $j \in \mathcal{S}_1$ (the shares of $s^1$ corresponding to the first $t-1$ leakage queries), with shares of a dummy seed $\widetilde{s}^1 = 0^d$. In general, for each $1 < q \leq h$, the only change we make in $\mathsf{LeakA}_q^m$ (in comparison to the previous hybrid $\mathsf{LeakB}_{q-1}^m$) is that we replace the shares $sd_j^q$, for each $j \in \mathcal{S}_q$ (the shares of $s^q$ corresponding to the $q$-th set of $t-1$ leakage queries), with shares of a dummy seed $\widetilde{s}^q$. After answering the leakage queries corresponding to $\mathcal{S}_q$, shares of $s^q$ are re-sampled consistent with the dummy seed shares used so far. The hybrid is formally described in Fig. 4.

$\mathsf{LeakA}_q^m$:

1. Initialize $Z$ to be a null string and $\mathcal{S}_1, \cdots, \mathcal{S}_h$ to be null sets.
2. $(m_1, \cdots, m_n) \leftarrow \mathsf{MShare}(m)$
3. For $i \in [h]$, choose $s^i \in_R \{0,1\}^{d_i}$
4. For $i \in [h]$ and $j \in [n]$, choose $w_j^i \in_R \{0,1\}^{\eta_i}$
5. For $i \in [h] \backslash [q]$, compute $(sd_1^i, \cdots, sd_n^i) \leftarrow \mathsf{SdShare}^i(s^i)$
6. For $i \in [q]$, let $\widetilde{s}^i = 0^d$
7. For $j \in [n]$, define $y_j^{h+1} = m_j$
8. **Leakage Phase:**
   (a) For $c \leftarrow 1$ to $q$
      i. $(\widetilde{sd}_1^c, \cdots, \widetilde{sd}_n^c) \leftarrow \mathsf{SdShare}^c(\widetilde{s}^c)$
      ii. For up to $(t-1)$ times
         A. $(j, f_j) \leftarrow \mathcal{D}(Z)$
         B. If $c < q$,
            * Choose $x_j^c \in_R \{0,1\}^{l_c}$ and compute $y_j^c = (x_j^c || \widetilde{sd}_j^c)$
            * For $i \leftarrow c-1$ down to 1, compute $x_j^i = y_j^{i+1} \oplus \mathsf{Ext}^i(w_j^i; s^i)$ and $y_j^i = (x_j^i || sd_j^i)$
         C. If $c = q$, for $i \leftarrow h$ down to 1 compute
            $\begin{cases} x_j^i = y_j^{i+1} \oplus \mathsf{Ext}^i(w_j^i; s^i) \text{ and } y_j^i = (x_j^i || sd_j^i) \text{ when } i \neq q \\ x_j^i = y_j^{i+1} \oplus \mathsf{Ext}^i(w_j^i; s^i) \text{ and } y_j^i = (x_j^i || \widetilde{sd}_j^i) \text{ when } i = q \end{cases}$
         D. Define $Sh_j = (w_j^1, \cdots, w_j^h, y_j^1)$
         E. Add $j$ to $\mathcal{S}_c$ and append $(j, f_j, f_j(Sh_j))$ to $Z$
      iii. $(sd_1^c, \cdots, sd_n^c) \leftarrow \mathsf{SdShare}^c(s^c | \widetilde{sd}_{\mathcal{S}_c}^c)$
   (b) For $j \in [n] \backslash (\mathcal{S}_{[q]})$ and $i \leftarrow h$ down to 1, compute $x_j^i = y_j^{i+1} \oplus \mathsf{Ext}^i(w_j^i; s^i)$ and $y_j^i = (x_j^i || sd_j^i)$
   (c) Define $Sh_j = (w_j^1, \cdots, w_j^h, y_j^1)$
   (d) For $c \leftarrow q+1$ to $h$
      i. For upto $t-1$ times
         A. $(j, f_j) \leftarrow \mathcal{D}(Z)$
         B. Add $j$ to $\mathcal{S}_c$ and append $(j, f_j, f_j(Sh_j))$ to $Z$
9. **Reveal phase**
   (a) For upto $t-1$ times
      i. $j \leftarrow \mathcal{D}(Z)$
      ii. Append $(j, Sh_j)$ to $Z$
10. $\mathcal{D}$ updates $Z$ with any relevant state information.
11. Output $Z$.

**Fig. 4.** Hybrid $\mathsf{LeakA}_q^m$

$\underline{\mathsf{LeakB}_q^m}$: For $q = 1$, $\mathsf{LeakB}_1^m$ follows the hybrid $\mathsf{LeakA}_1^m$ except for the following change: in $\mathsf{LeakB}_1^m$, we replace the values $x_j^1$, for each $j \in \mathcal{S}_1$ with random, instead of evaluating the $h$ layers of masking to get $x_j^1$ (and hence $x_j^1$'s for $j \in \mathcal{S}_1$ are independent of $m_{\mathcal{S}_1}$, $s^i$ and the shares of $s^i$, for each $1 < i \leq h$). Note that in $\mathsf{LeakA}_1^m$, the shares $Sh_j$ corresponding to $\mathcal{S}_1$ no longer depend on the seed $s^1$. We carefully use the adaptive extractor security of $\mathsf{Ext}^1$ to move to $\mathsf{LeakB}_1^m$. In general, for each $1 < q \leq h$, the only change we make in $\mathsf{LeakB}_q^m$ (in comparison

$\mathsf{LeakB}_q^m$

1. Initialize $Z$ to be a null string and $\mathcal{S}_1, \cdots, \mathcal{S}_h$ to be null sets.
2. $(m_1, \cdots, m_n) \leftarrow \mathsf{MShare}(m)$
3. For $i \in [h]$, choose $s^i \in_R \{0,1\}^{d_i}$
4. For $i \in [h]$ and $j \in [n]$, choose $w_j^i \in_R \{0,1\}^{\eta_i}$
5. For $i \in [h] \backslash [q]$, compute $(sd_1^i, \cdots, sd_n^i) \leftarrow \mathsf{SdShare}^i(s^i)$
6. For $i \in [q]$, let $\widetilde{s}^i = 0^d$
7. For $j \in [n]$, define $y_j^{h+1} = m_j$
8. **Leakage Phase:**
   (a) For $c \leftarrow 1$ to $q$
      i. $(\widetilde{sd}_1^c, \cdots, \widetilde{sd}_n^c) \leftarrow \mathsf{SdShare}^c(\widetilde{s}^c)$
      ii. For upto $(t-1)$ times
         A. $(j, f_j) \leftarrow \mathcal{D}(Z)$
         B. Choose $x_j^c \in_R \{0,1\}^{l_c}$ and compute $y_j^c = (x_j^c || \widetilde{sd}_j^c)$
         C. For $i \leftarrow c-1$ down to 1
            compute $x_j^i = y_j^{i+1} \oplus \mathsf{Ext}^i(w_j^i; s^i)$ and $y_j^i = (x_j^i || sd_j^i)$
         D. Define $Sh_j = (w_j^1, \cdots, w_j^h, y_j^1)$
         E. Add $j$ to $\mathcal{S}_c$ and append $(j, f_j, f_j(Sh_j))$ to $Z$
      iii. $(sd_1^c, \cdots, sd_n^c) \leftarrow \mathsf{SdShare}^c(s^c | \widetilde{sd}_{\mathcal{S}_c}^c)$
   (b) For $j \in [n] \backslash \mathcal{S}_{[q]}$ and $i \leftarrow h$ to 1, ($\mathcal{S}_{[q]}$ denotes a null set when $q = 0$)
   compute $x_j^i = y_j^{i+1} \oplus \mathsf{Ext}^i(w_j^i; s^i)$ and $y_j^i = (x_j^i || sd_j^i)$
   (c) Define $Sh_j = (w_j^1, \cdots, w_j^h, y_j^1)$
   (d) For $c \leftarrow q+1$ to $h$
      i. For upto $t-1$ times
         A. $(j, f_j) \leftarrow \mathcal{D}(Z)$
         B. Add $j$ to $\mathcal{S}_c$ and append $(j, f_j, f_j(Sh_j))$ to $Z$
9. **Reveal phase**
   (a) For upto $t-1$ times
      i. $j \leftarrow \mathcal{D}(Z)$
      ii. Append $(j, Sh_j)$ to $Z$
10. $\mathcal{D}$ updates $Z$ with any relevant state information.
11. Output $Z$.

**Fig. 5.** Hybrid $\mathsf{LeakB}_q^m$

to the previous hybrid $\mathsf{LeakA}_q^m$) is that we replace the values $x_j^q$, for each $j \in \mathcal{S}_q$ with random, instead of evaluating the $h-(q-1)$ layers of masking to get $x_j^q$ (and hence, for these queries in $\mathcal{S}_q$, $s^i$ and the shares of $s^i$, for each $q < i \leq h$, and the shares $m$ are not used to evaluate $x_j^q$). Further, we continue the steps of masking to evaluate $x_j^{q-1}, x_j^{q-2}, \cdots, x_j^1$, for each $j \in \mathcal{S}_q$ as in the previous hybrid. The hybrid is formally described in Fig. 5.

$\underline{\mathsf{LeakC}^m}$*:* In the hybrid $\mathsf{LeakB}_h^m$, all the shares used in the leakage phase are independent of the shares of the message $m$. Hence, the only part of the view of $\mathcal{D}$ that depends on the shares of $m$ corresponds to the reveal phase. In the final hybrid $\mathsf{LeakC}^m$, we replace the $t-1$ shares of $m$ used in the reveal phase by shares of $0^l$. This hybrid is formally described in Fig. 6.

The formal descriptions of all hybrids are given below with the change from the prior hybrid highlighted in red color.

$\mathsf{LeakC}^m$

1. Initialize $Z$ to be a null string and $\mathcal{S}_1, \cdots, \mathcal{S}_h$ to be null sets.
2. Let $\tilde{m} = 0^l$ and $(\tilde{m}_1, \cdots, \tilde{m}_n) \leftarrow \mathsf{MShare}(\tilde{m})$
3. For $i \in [h]$, choose $s^i \in_R \{0,1\}^{d_i}$
4. For $i \in [h]$, let $\widetilde{s}^i = 0^d$
5. For $i \in [h]$ and $j \in [n]$, choose $w_j^i \in_R \{0,1\}^{\eta_i}$
6. **Leakage Phase:**
   (a) For $c \leftarrow 1$ to $h$
      i. $(\widetilde{sd}_1^c, \cdots, \widetilde{sd}_n^c) \leftarrow \mathsf{SdShare}^c(\widetilde{s}^c)$
      ii. For upto $(t-1)$ times
         A. $(j, f_j) \leftarrow \mathcal{D}(Z)$
         B. Choose $x_j^c \in_R \{0,1\}^{l_c}$ and compute $y_j^c = (x_j^c || \widetilde{sd}_j^c)$
         C. For $i \leftarrow c-1$ down to 1
            compute $x_j^i = y_j^{i+1} \oplus \mathsf{Ext}^i(w_j^i; s^i)$ and $y_j^i = (x_j^i || sd_j^i)$
         D. Define $Sh_j = (w_j^1, \cdots, w_j^h, y_j^1)$
         E. Add $j$ to $\mathcal{S}_c$ and append $(j, f_j, f_j(Sh_j))$ to $Z$
      iii. $(sd_1^c, \cdots, sd_n^c) \leftarrow \mathsf{SdShare}^c(s^c | \widetilde{sd}_{\mathcal{S}_c}^c)$
7. **Reveal phase**
   (a) For upto $t-1$ times
      i. $j \leftarrow \mathcal{D}(Z)$
      ii. Define $y_j^{h+1} = \tilde{m}_j$
      iii. For $i \leftarrow h$ to 1, compute $x_j^i = y_j^{i+1} \oplus \mathsf{Ext}^i(w_j^i; s^i)$ and $y_j^i = (x_j^i || sd_j^i)$
      iv. Define $Sh_j = (w_j^1, \cdots, w_j^h, y_j^1)$
      v. Append $(j, Sh_j)$ to $Z$
8. $\mathcal{D}$ updates $Z$ with any relevant state information.
9. Output $Z$.

**Fig. 6.** Hybrid $\mathsf{LeakC}^m$. (Color figure online)

We begin by proving the statistical closeness of $\mathsf{LeakA}_q^m$ and $\mathsf{LeakB}_{q-1}^m$, for each $q \in [h]$, which follows from adaptive privacy of $\mathsf{SdShare}^q$, as atmost only $t-1$ dummy seed shares are used.

**Claim 1.** *For $q \in [h]$, if $\mathsf{SdShare}^q$ is $\epsilon'_q$-adaptively private against $(n,t)$-threshold access structures, then $\mathsf{LeakA}_q^m \approx_{\epsilon'_q} \mathsf{LeakB}_{q-1}^m$.*

*Proof.* **Answering the first $(q-1)$ sets of leakage queries (when $q > 1$):** Observe that the hybrids are identical up to answering the first $(q-1)(t-1)$ leakage queries and differ in answering the remaining queries. For any $k \in [q-1]$ and, $j \in \mathcal{S}_k$ the leakage response only depends on $\widetilde{sd}_j^k$, $w_j^1, \cdots, w_j^h$ and $\{s^i, sd_j^i\}_{1 \le i < k}$ (as $x_j^k$ is chosen uniformly). We let $\mathsf{Pre}$ denote the union of these random variables upon which the leakage responses to $j \in \mathcal{S}_{[q-1]}$ depend.

**Answering the $q^{th}$ Set of Leakage Queries:** Consider $j \in \mathcal{S}_q$. To answer this leakage query, it suffices to compute $Sh_j = (w_j^1, \cdots, w_j^h, y_j^1)$. The hybrids only differ in computation of $y_j^1$ (particularly in computation of $y_j^q$, which is used to compute $y_j^1$) and the distribution of extractor sources is identical in both. We highlight the differences here. $\mathsf{LeakA}_q^m$ (Step 8-(a)-ii-C), iteratively computes $y_j^h, \cdots, y_j^q, \cdots, y_j^1$ as follows.

- $(y_j^h, \cdots, y_j^{q+1})$ are computed using $y_j^{h+1}$ and $\{w_j^i, sd_j^i, s^i\}_{i \in [h] \setminus [q]}$. Note that the distribution of $y_j^h, \cdots, y_j^{q+1}$ is identical in both hybrids.
- $x_j^q$ is computed using $y_j^{q+1}, w^q$ and $s^q$. $x_j^q$ is also identical in both hybrids.
- $y_j^q$ is computed as $x_j^q || \widetilde{sd}_j^q$ (where $\widetilde{sd}_{[n]}^q$ are shares of a dummy seed $\widetilde{s}^q$ which are generated before answering any queries in $\mathcal{S}_q$ in Step 8-(a)-i (when $c = q$)). Whereas in $\mathsf{LeakB}_{q-1}^m$, $y_j^q = x_j^q || sd_j^q$ (where $sd_{[n]}^q$ are shares of $s^q$)
- $(y_j^{q-1}, \cdots, y_j^1)$ are computed using $y_j^q$ and $\{sd_j^i, w_j^i, s^i\}_{i \in [q-1]}$. The computation of $(y_j^{q-1}, \cdots, y_j^1)$ given the later random variables is again identical to $\mathsf{LeakB}_{q-1}^m$.
- Now $\mathsf{LeakA}_q^m$ defines $Sh_j = (w_j^1, \cdots, w_j^h, y_j^1)$

For convenience, in this proof we distinguish (whenever necessary) the random variables that have same literal in both the hybrids but are distributionally different with subscripts A and B respectively. For example, $y_{j,A}^q$ and $y_{j,B}^q$ denote the distributions of $y_j^q$ in $\mathsf{LeakA}_q^m$ and, $\mathsf{LeakB}_{q-1}^m$ respectively.

Let $\mathsf{Pre}' = (\{w_j^q, \{sd_j^i, w_j^i, s^i\}_{i \in [h] \setminus \{q\}}\}_{j \in [n] \setminus \mathcal{S}_{[q-1]}})$. $\mathsf{Pre}'$ captures the information required to answer all queries after the first $q-1$ sets of leakage queries, except for any information regarding $s^q, \widetilde{s}^q$ and their shares. Note that $\mathsf{Pre}'$ is identical in both hybrids[9]. Since, $|\mathcal{S}_q| \le t-1$, with a reduction to adaptive privacy of $\mathsf{SdShare}^q$ we have

[9] $\mathsf{Pre}'$ possibly repeats some information already there in $\mathsf{Pre}$. For example for $q = 2$, $s^1$ is there in both $\mathsf{Pre}$ and $\mathsf{Pre}'$. It is for the ease of exposition that we have this repetition.

$$\mathsf{Pre}, \mathsf{Pre}', s^q, \widetilde{s}^q, \{\widetilde{sd}_j^q\}_{j\in\mathcal{S}_{q,A}} \approx_{\epsilon'_q} \mathsf{Pre}, \mathsf{Pre}', s^q, \widetilde{s}^q, \{sd_j^q\}_{j\in\mathcal{S}_{q,B}}$$

as $(\mathsf{Pre}, \mathsf{Pre}')$ is independent of the randomness used to generate the shares of $\widetilde{s}^q$ and $s^q$. Note that the information on LHS suffices to answer the first $q$ sets of queries as per $\mathsf{LeakA}_q^m$. Similarly, RHS suffices to answer queries in $\mathcal{S}_{[q]}$ as per $\mathsf{LeakB}_{q-1}^m$. Therefore, we have,

$$\mathsf{Pre}, \mathsf{Pre}', s^q, \tilde{s}^q, \{\widetilde{sd}_j^q\}_{j\in\mathcal{S}_{q,A}}, Z_{[q],A} \approx_{\epsilon'_q} \mathsf{Pre}, \mathsf{Pre}', s^q, \tilde{s}^q, \{sd_j^q\}_{j\in\mathcal{S}_{q,B}}, Z_{[q],B} \quad (1)$$

**Answering the Leakage and Reveal Queries Made After the $q^{th}$ Set of Leakage Queries:** After all the $q^{th}$ set leakage queries are answered, $\mathsf{LeakA}_q^m$ computes $(sd_1^q, \cdots, sd_n^q) \leftarrow \mathsf{SdShare}^q(s^q|\widetilde{sd}^q_{\mathcal{S}_{q,A}})$. Given $(sd_1^q, \cdots, sd_n^q), s^q, \mathsf{Pre}$ and $\mathsf{Pre}'$, for any $j \in [n]\backslash\mathcal{S}_q$, $Sh_j$ is easily computed (Steps 8-(b) and 8-(c)). With this, any further queries can be correctly answered as per $\mathsf{LeakA}_q^m$. Let $(\widehat{sd}_1^q, \cdots, \widehat{sd}_n^q) \leftarrow \mathsf{SdShare}^q(s^q|sd^q_{\mathcal{S}_{q,B}})$. By Lemma 2, we have

$$\mathsf{Pre}, \mathsf{Pre}', s^q, \tilde{s}^q, Z_{[q],A}, sd^q_{[n],A} \approx_{\epsilon'_q} \mathsf{Pre}, \mathsf{Pre}', s^q, \tilde{s}^q, Z_{[q],B}, \widehat{sd}^q_{[n],B}$$

Note that $\widehat{sd}^q_{[n]}$ is identical to $sd^q_{[n],B}$ (of $\mathsf{LeakB}_{q-1}^m$) even given $s^q$ and $\{sd_j^q\}_{j\in\mathcal{S}_q}$ by the property of consistent resampling in Claim 5. Therefore, we have,

$$\mathsf{Pre}, \mathsf{Pre}', s^q, Z_{[q],A}, sd^q_{[n],A} \approx_{\epsilon'_q} \mathsf{Pre}, \mathsf{Pre}', s^q, Z_{[q],B}, sd^q_{[n],B}$$

Since the above LHS and RHS are sufficient to answer any further queries, we have

$$Z_{[h+1],A} \approx_{\epsilon'_q} Z_{[h+1],B}$$

which proves the claim.

Now, we prove the statistical closeness of $\mathsf{LeakA}_q^m$ and $\mathsf{LeakB}_q^m$, for each $q \in [h]$ using the adaptive extractor security. The high-level idea behind the reduction is that in hybrid $\mathsf{LeakA}_q^m$, the shares corresponding to the first $q(t-1)$ queries (i.e., $\mathcal{S}_{[q]}$) no longer depend on the seed $s^q$ and hence, we can use the adaptive extractor security of $\mathsf{Ext}^q$ to move to $\mathsf{LeakB}_q^m$.

**Claim 2.** *For $q \in [h]$, if $\mathsf{Ext}^q$ is an $(\eta_q, \mu_q + \tau, d_q, l_q, \delta'_q)$- extractor that is $(\mathsf{Full}_\tau, \delta_q)$-adaptive, then $\mathsf{LeakA}_q^m \approx_{(t-1)\delta_q} \mathsf{LeakB}_q^m$*

*Proof.* Observe that the hybrids are identical up to answering the first $(q-1)(t-1)$ leakage queries and differ in answering the $q^{th}$ set of queries. Further, after answering the $q^{th}$ set of leakage queries, the responses to all remaining leakage/reveal queries are answered identically in both hybrids.

**Answering the first $(q-1)$ Sets of Leakage Queries (when $q > 1$):** For any $k \in [q-1]$ and $j \in \mathcal{S}_k$ the leakage response only depends on $\widetilde{sd}_j^k$, $w_j^1, \cdots, w_j^h$, $\{s^i, sd_j^i\}_{1\le i<k}$ and $x_j^k$, where the latter is uniformly chosen. We let

$\mathsf{Pre}$ denote the leakage responses $Z_{[q-1]}$ and the union of these random variables upon which the leakage responses to $j \in \mathcal{S}_{[q-1]}$ depend.

**Answering the $q^{th}$ Set of Leakage Queries:**
Consider $j \in \mathcal{S}_q$ and $f_j$ be the corresponding leakage function. To answer this leakage query, we require computing $f_j(Sh_j)$ where $Sh_j = (w_j^1, \cdots, w_j^h, y_j^1)$. The hybrids only differ in computation of $y_j^1$ (particularly in computation of $x_j^q$, which is used to compute $y_j^1$) and the distribution of extractor sources is identical in both. The hybrids iteratively computes $y_j^q, \cdots, y_j^1$ as follows.

- $x_j^q$ is chosen uniformly from $\{0,1\}^{l_q}$ in $\mathsf{LeakB}_q^m$. In contrast, $x_j^q$ of $\mathsf{LeakA}_q^m$ depended on $\mathsf{Ext}^q(w_j^q; s^q)$ and $y_j^{q+1}$.
- $(y_j^q, \cdots, y_j^1)$ is determined given $x_j^q, \widetilde{sd}_j^q$ and $\{sd_j^i, w_j^i, s^i\}_{i\in[q-1]}$ in both the hybrids.
- Both hybrids define $Sh_j = (w_j^1, \cdots, w_j^h, y_j^1)$

Let $\mathsf{Pre}' = \{w_j^i, sd_j^i, s^i, y_j^{h+1}, \widetilde{sd}_j^q\}_{i\in[h]\setminus\{q\}, j\in[n]\setminus\mathcal{S}_{[q-1]}}$.We capture $\mathsf{Pre}'$ as the information which along with $\{w_j^q, s^q\}_{j\in\mathcal{S}_q}$ is sufficient to answer any leakage queries on $j \in \mathcal{S}_q$. Also, $\mathsf{Pre}'$ is identical in both hybrids.
Let $j_1, \cdots, j_{t-1}$ be the order of indices in which leakage queries are made in $\mathcal{S}_q$. Firstly, we prove that $(\mathsf{Pre}, \mathsf{Pre}', f_{j_1}(Sh_{j_1}))$ of both hybrids are statistically close. After that we proceed to show that $(\mathsf{Pre}, \mathsf{Pre}', f_{j_1}(Sh_{j_1}), \cdots, f_{j_{(t-1)}}(Sh_{j_{(t-1)}}))$ of both the hybrids are statistically close, which implies that the hybrids are statistically close up to answering first $q$ sets of queries. For convenience, in this proof we distinguish (whenever necessary) the random variables that have same literal in the hybrids but are distributionally different with subscripts A and B respectively. For example, $x_{j,A}^q$ and $x_{j,B}^q$ denote the distributions of $x_j^q$ in $\mathsf{LeakA}_q^m$ and $\mathsf{LeakB}_q^m$ respectively.

Firstly, in both hybrids the distribution of $(j_1, f_{j_1})$ only depends on $Z_{[q-1]}$ (and any internal randomness of $\mathcal{D}$) and hence are identical. Note that given $\mathsf{Pre}'$, $f_{j_1}(Sh_{j_1})$ in $\mathsf{LeakA}_q^m$, can be captured as $\mathsf{Full}_\tau$-adaptive leakage on the extractor source $w_{j_1}^q$ and $(x_{j_1,A}^q =)\ \mathsf{Ext}^q(w_{j_1}^q; s^q) \oplus y_{j_1}^{q+1}$. This is because $(y_{j_1}^{q+1}, \mathsf{Pre}')$ are independent of $(w_{j_1}^q, s^q)$. Let $g_1$ be a function that takes $\mathsf{Pre}', w_{j_1}^q$ and $x_{j_1,A}^q$(or $x_{j_1,B}^q$) as input, computes $y_{j_1,A}^1$ (or $y_{j_1,B}^1$) and outputs $f_j(w_{j_1}^1, \cdots, w_{j_1}^h, y_{j_1,A}^1)$ (or $f_j(w_{j_1}^1, \cdots, w_{j_1}^h, y_{j_1,B}^1)$). With a reduction to adaptive security of $\mathsf{Ext}^q$ we have

$$\begin{aligned}
&\mathsf{Pre}, \mathsf{Pre}', s^q, g_1(\mathsf{Pre}', w_{j_1}^q, \mathsf{Ext}^q(w_{j_1}^q; s^q) \oplus y_j^{q+1}) \\
&\approx_{\delta_q} \mathsf{Pre}, \mathsf{Pre}', s^q, g_1(\mathsf{Pre}', w_{j_1}^q, U_{l_q} \oplus y_j^{q+1}) \\
&\equiv \mathsf{Pre}, \mathsf{Pre}', s^q, g_1(\mathsf{Pre}', w_{j_1}^q, x_{j_1,B}^q)
\end{aligned}$$

Therefore

$$\mathsf{Pre}, \mathsf{Pre}', s^q, f_{j_1}(Sh_{j_1,A}) \approx_{\delta_q} \mathsf{Pre}, \mathsf{Pre}', s^q, f_{j_1}(Sh_{j_1,B})$$

With this, we showed that the hybrids are statistically close up to responding to the first query in the $q^{th}$ set. Although, superficially, it may seem that all the

leakage responses corresponding to $j \in \mathcal{S}_q$ can be captured as adaptive extractor leakage on the source $w_j^q$, but it's not the case because of the following subtlety. The extractor sources used in each query are independent of each other, but the seed is the same. For example, one cannot directly capture $f_{j_2}(Sh_{j_2})$ as $\mathsf{Full}_\tau$-adaptive leakage (as we did with $f_{j_1}(Sh_{j_1})$). This is because the choice of $j_2, f_{j_2}$ depends on $f_{j_1}(Sh_{j_1})$ which in turn depends on $\mathsf{Ext}^q(w_j^q; s^q)$, and hence is not independent of the seed $s^q$. We observe in Lemma 8 that adaptive extractors allow us to handle even such (stronger) form of adaptive leakages across different sources with same seed.

Proceeding, with a reduction to Lemma 8 with $k = (t-1), \{W_i = W_{j_i}^q : i \in [k]\}$, $S = s^q$ and $\mathsf{Ext} = \mathsf{Ext}^q$ and the $i^{th}$ leakage function being $g_i$ such that $g_i$ (hardwired with $\mathsf{Pre}', y_{j_i}^{q+1}$) takes $w_{j_i}^q$ and $\mathsf{Ext}^q(w_{j_i}^q; s^q)$ (resp. $U_{l_q}$) as input, computes $y_{j_i,A}^1$ (resp. $y_{j_i,B}^1$) and outputs $f_{j_i}(w_{j_i}^1, \cdots, w_{j_i}^h, y_{j_i,A}^1)$ (resp. $f_{j_i}(w_{j_i}^1, \cdots, w_{j_i}^h, y_{j_i,B}^1)$).

$$\mathsf{Pre}, \mathsf{Pre}', s^q, \{f_{j_i}, f_{j_i}(Sh_{j_i,A})\}_{j_i \in \mathcal{S}_{q,A}}, \mathcal{S}_{q,A}$$

$$\approx_{(t-1)\delta_q} \mathsf{Pre}, \mathsf{Pre}', s^q, \{f_{j_i}, f_{j_i}(Sh_{j_i,B})\}_{j_i \in \mathcal{S}_{q,B}}, \mathcal{S}_{q,B}$$

This shows that the hybrids are statistically close up to answering the first $q$ sets of leakage queries.

**Answering the Leakage and Reveal Queries Made After the $q^{th}$ Set of Leakage Queries:** After all the $q^{th}$ set of leakage queries are answered, both hybrids compute $(sd_1^q, \cdots, sd_n^q) \leftarrow \mathsf{SdShare}(s^q | \widetilde{sd}_{\mathcal{S}_q}^q)$. Let $\mathsf{Pre}'' = \{w_j^q, sd_j^q, s^q\}_{j \in [n] \setminus \mathcal{S}_q}$. Note that $\mathsf{Pre}'$ in conjunction with $\mathsf{Pre}''$ completely defines $Sh_j$ for any $j \in [n] \setminus \mathcal{S}_{[q]}$. Since $\mathsf{Pre}''$ corresponding to $\mathsf{LeakA}_q^m$(resp. $\mathsf{LeakB}_q^m$) is only correlated to $\mathcal{S}_q, s^q$ and $\widetilde{sd}_{\mathcal{S}_q}^q$(which is in $\mathsf{Pre}'$) of the respective hybrids, we have

$$\mathsf{Pre}, \mathsf{Pre}', \mathsf{Pre}_A'', s^q, \{f_{j_i}, f_{j_i}(Sh_{j_i,A})\}_{j_i \in \mathcal{S}_{q,A}}, \mathcal{S}_{q,A}$$

$$\approx_{(t-1)\delta_q} \mathsf{Pre}, \mathsf{Pre}', \mathsf{Pre}_B'', s^q, \{f_{j_i}, f_{j_i}(Sh_{j_i,B})\}_{j_i \in \mathcal{S}_{q,B}}, \mathcal{S}_{q,B}$$

Since responses to leakage/reveal queries after the $q^{th}$ set are can be derived from the LHS and RHS respectively depending on the hybrid, we have

$$Z_{[h+1],A} \approx_{(t-1)\delta_q} Z_{[h+1],B}$$

This proves the claim.

Finally, we use the adaptive security of $\mathsf{MShare}$ to show that $\mathsf{LeakC}^m$ is statistically close to $\mathsf{LeakB}_h^m$.

**Claim 3.** *If* $\mathsf{MShare}$ *is $\epsilon$-adaptively private against $(n,t)$-threshold access structures, then* $\mathsf{LeakC}^m \approx_\epsilon \mathsf{LeakB}_h^m$.

*Proof.* The hybrids answer the leakage queries identically and differ only in answering the reveal queries.

**Answering the Leakage Queries:**
For any $k \in [h]$ and $j \in \mathcal{S}_k$ the leakage response only depends on $\widetilde{sd}_j^k, w_j^1, \cdots, w_j^h$, $\{s^i, sd_j^i\}_{1 \le i < k}$ and $x_j^k$, where the latter is uniformly chosen. We let $\mathsf{Pre}$ denote the leakage responses $Z_{[h]}$ and the union of these random variables upon which the leakage responses to $j \in \mathcal{S}_{[h]}$ depend.

**Answering the Reveal Queries:** Let $\mathsf{Pre}' = \{w_j^i, sd_j^i, s^i\}_{i \in [h], j \in [n] \setminus \mathcal{S}_{[h]}}$. Note that given $y_j^{h+1}$ for all $j$ queried in the reveal phase, $(\mathsf{Pre}, \mathsf{Pre}')$ has sufficient information to answer all the reveal queries.

- $\mathsf{LeakB}_h^m$ samples $(m_1, \cdots, m_n) \leftarrow \mathsf{MShare}(m)$ and sets $y_j^{h+1} = m_j$ for all $j$ queried in the reveal phase.
- $\mathsf{LeakC}^m$ samples $(\tilde{m}_0, \cdots, \tilde{m}) \leftarrow \mathsf{MShare}(\tilde{m})$ and sets $y_j^{h+1} = \tilde{m}_j$ for all $j$ queried in the reveal phase.

Let $\mathsf{RevealB}$ and $\mathsf{RevealC}$ denote the sets of indices queried in the reveal phase of $\mathsf{LeakB}_h^m$ and $\mathsf{LeakC}^m$ respectively. As reveal queries are at most $t-1$ in number, we now invoke adaptive privacy of $\mathsf{MShare}$ and get

$$\mathsf{Pre}, \mathsf{Pre}', \tilde{m}, m, \{m_j\}_{j \in \mathsf{RevealB}} \approx_\epsilon \mathsf{Pre}, \mathsf{Pre}', \tilde{m}, m, \{\tilde{m}_j\}_{j \in \mathsf{RevealC}}$$

Note that $(\mathsf{Pre}, \mathsf{Pre}')$ is independent of the randomness used in generating shares of $m$ and $\tilde{m}$, therefore adaptive privacy of $\mathsf{MShare}$ can be invoked even given these random variables.

Since $Sh_j$ for $j$ queried in reveal phase of $\mathsf{LeakB}_h^m$ (resp. $\mathsf{LeakC}^m$) is determined by the above LHS (resp. RHS) we have

$$\underbrace{Z_{[h+1]}}_{of\ \mathsf{LeakB}_q^m} \approx_\epsilon \underbrace{Z_{[h+1]}}_{of\ \mathsf{LeakC}^m}$$

With the above claims and use of triangle inequality we know that for any message $m$, $\mathsf{Leak}_{Share^h}^m \approx_{\epsilon + \sum_{i \in [h]}((t-1)\delta_i + \epsilon_i')} \mathsf{LeakC}^m$. Note that the description of $\mathsf{LeakC}^m$ is independent of $m$. Hence for any message $m \neq m'$, we have $\mathsf{LeakC}^m \equiv \mathsf{LeakC}^{m'}$. Since, $\mathsf{Leak}_{Share^h}^{m'} \approx_{h\epsilon' + h(t-1)\delta + \epsilon} \mathsf{LeakC}^{m'}$ we get

$$\mathsf{Leak}_{Share^h}^m \approx_{2\epsilon + 2\sum_{i \in [h]}((t-1)\delta_i + \epsilon_i')} \mathsf{Leak}_{Share^h}^{m'}$$

### 4.4 Parameters

For $i \in [h]$, we instantiate $\mathsf{SdShare}^i$ on seeds of length $d_i$ with the (adaptively) private Shamir secret sharing scheme, which results in individual seed share length being $d_i$. We instantiate $\mathsf{MShare}$ on messages of length $l_i$ with the (adaptively) private Shamir secret sharing scheme, which results in individual seed share length being $l_i$.

Recall Lemma 7 which states that for any $c > 1$, there exists constants $\alpha, \beta$ such that $d \leq \alpha l$, $\mu \leq \beta l$, $\eta \geq \beta l + \tau$, $\epsilon = 2^{-cl}$ and $\delta = 2^{-(c-1)l+2}$ when $l = \omega(\log \eta)$. Fix any $c > 1$, and constants $\alpha, \beta$ corresponding to this $c$ given by Lemma 7. For each $i \in [h]$, we instantiate $(\eta_i, \mu_i + \tau, d_i, l_i, \delta_i')$-extractor $\mathsf{Ext}^i$ that is $(\mathsf{Full}_\tau, \delta_i)$-adaptive as per this lemma as follows.

- We set $l_1 = l$, $\delta_1' = 2^{-cl}$, $\delta_1 = 2^{-\Omega(l)}$, $d_1 \leq \alpha l_1$, $\mu_1 \leq \beta l_1$ and $\eta_1 = \beta l_1 + \tau$.
- For $i > 1$, we set $l_i = l_{i-1} + d_{i-1}$, $\delta_i' = 2^{-cl_i}$, $\delta_i = 2^{-\Omega(l_i)}$, $d_i \leq \alpha l_i$, $\mu_i \leq \beta l_i$ and $\eta_i = \beta l_i + \tau$.

With this setting, individual share length of $\mathsf{Share}^h$ is $l_h + d_h + \sum_{i \in [h]} \eta_i = h\tau + \Theta((1+\alpha)^h l)$. Therefore, $\mathsf{Share}^h$ acheives constant rate and constant leakage rate whenever $\tau = \mathcal{O}(l)$ and either $n = \Theta(t)$ or $h$ is a constant.

As our instantiations of $\mathsf{SdShare}^i$'s and $\mathsf{MShare}$ are perfectly adaptively private, we have $\mathsf{Share}^h$ to be a perfectly adaptively private secret sharing scheme which is $t \cdot 2^{-\Omega(l)}$-leakage resilient against the adaptive leakage and reveal model.

### 4.5 LRSS for Joint Leakage and Reveal Model

#### 4.5.1 Joint Leakage and Reveal Model $\mathcal{J}^{X,\psi,\tau}$

The model allows for $\psi$ number of joint leakage queries on disjoint sets where each query depends on $X$ number of shares and additionally also reveals $t-1$ of the remaining shares (on which leakage isn't queried) in clear. The parameter $\tau$ captures the amount of leakage provided in each leakage query.

Let $(Share, Rec)$ (where $Share : \{0,1\}^l \to (\{0,1\}^\gamma)^n$) be a secret sharing scheme for an $(n, t)$- threshold access structure. We formalize leakage obtained in this model on shares of a message $m$ as $\mathsf{JLeak}^m_{Share}$ in Fig. 7, where an arbitrary stateful distinguisher $\mathcal{D}$ makes the queries. For any two messages $m$ and $m'$, we require $\mathsf{JLeak}^m_{Share} \approx_{\epsilon_{lr}} \mathsf{JLeak}^{m'}_{Share}$, for $(Share, Rec)$ to be $\epsilon_{lr}$ leakage resilient against this model.

$\mathsf{JLeak}^m_{Share}$:

- Initialize $Z$ be a null string and $\mathcal{S}$ to be a null set.
- $(Sh_1, \cdots, Sh_n) \leftarrow Share(m)$
- **Leakage Phase:**
  For upto $\psi$ times
  - $(Q_j, f_j) \leftarrow \mathcal{D}(Z)$ where $Q_j \subseteq [n]$ and $f_j : \{0,1\}^{|Q_j|\gamma} \to \{0,1\}^\tau$
  - If $Q_j \in [n] \backslash \mathcal{S}$ and $|Q_j| \leq X$,
    add elements of $Q_j$ to $\mathcal{S}$ and append $(Q_j, f_j, f_j(Sh_{Q_j}))$ to $Z$
- **Reveal phase**
  For upto $t-1$ times
  - $j \leftarrow \mathcal{D}(Z)$
  - If $j \in [n] \backslash \mathcal{S}$, append $(j, Sh_j)$ to $Z$
- $\mathcal{D}$ updates $Z$ to include any relevant state information.
- Output $Z$

**Fig. 7.** Joint LRSS definition- $\mathsf{JLeak}^m_{Share}$ distribution

### 4.5.2 Leakage Resilience of $(\mathsf{Share}^h, \mathsf{Rec}^h)$ in $\mathcal{J}^{X,\psi,\tau}$ Model

**Theorem 3.** *For any $\psi, X > 0$ such that $\psi \cdot X \leq n - t + 1$ and $l, \tau > 0$, $(\mathsf{Share}^h, \mathsf{Rec}^h)$ is an $((n,t),\varepsilon)$-secret sharing scheme for $l$ bit messages and is $\epsilon_{lr}$-leakage resilient in the joint leakage and reveal model $\mathcal{J}^{X,\psi,\tau}$ where $h = \lceil \frac{\psi}{\lfloor (t-1)/X \rfloor} \rceil$ and $\epsilon_{lr} = 2(\epsilon + h\epsilon' + (t-1)\sum_{i\in[h]} 2^{Xl_i}\delta'_i))$.*

*Further, there exists an instantiation of the scheme with rate is $(X^{\Theta(h)} + h\tau/l)^{-1}$. When $\tau = \Theta(l)$, $X$ is a constant and when either $n = \Theta(t)$ or $h$ is a constant, the scheme achieves constant rate and leakage rate asymptotically.*

The proof for the joint leakage setting is very similar to the proof of Theorem 2 for the adaptive setting (on single shares). We give a complete proof of this in our full version.

Further, we can also extend our construction to get LRSS for general access structures as well, the details of which are given in the full version of our paper.

**Acknowledgement.** We thank all the anonymous reviewers who provided their valuable comments on an earlier version of this manuscript.

## References

1. Aggarwal, D., Damgård, I., Nielsen, J.B., Obremski, M., Purwanto, E., Ribeiro, J., Simkin, M.: Stronger leakage-resilient and non-malleable secret sharing schemes for general access structures. In: Boldyreva, A., Micciancio, D. (eds.) CRYPTO 2019. LNCS, vol. 11693, pp. 510–539. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-26951-7_18
2. Aggarwal, D., Dodis, Y., Jafargholi, Z., Miles, E., Reyzin, L.: Amplifying privacy in privacy amplification. In: Garay, J.A., Gennaro, R. (eds.) CRYPTO 2014. LNCS, vol. 8617, pp. 183–198. Springer, Heidelberg (2014). https://doi.org/10.1007/978-3-662-44381-1_11
3. Aggarwal, D., Dodis, Y., Kazana, T., Obremski, M.: Non-malleable reductions and applications. In: Proceedings of the Forty-Seventh Annual ACM on Symposium on Theory of Computing, STOC 2015 (2015). https://doi.org/10.1145/2746539.2746544
4. Aggarwal, D., Dodis, Y., Lovett, S.: Non-malleable codes from additive combinatorics. In: Symposium on Theory of Computing, STOC 2014 (2014). https://doi.org/10.1145/2591796.2591804
5. Badrinarayanan, S., Srinivasan, A.: Revisiting non-malleable secret sharing. In: Ishai, Y., Rijmen, V. (eds.) EUROCRYPT 2019. LNCS, vol. 11476, pp. 593–622. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-17653-2_20
6. Bellare, M., Rogaway, P.: Robust computational secret sharing and a unified account of classical secret-sharing goals. In: Proceedings of the 14th ACM Conference on Computer and Communications Security. CCS 2007, Association for Computing Machinery (2007). https://doi.org/10.1145/1315245.1315268
7. Ben-Or, M., Goldwasser, S., Wigderson, A.: Completeness theorems for non-cryptographic fault-tolerant distributed computation (extended abstract). In: Simon, J. (ed.) Proceedings of the 20th Annual ACM Symposium on Theory of Computing, Chicago, Illinois, USA, 2–4 May 1988, pp. 1–10. ACM (1988). https://doi.org/10.1145/62212.62213

8. Benaloh, J., Leichter, J.: Generalized secret sharing and monotone functions. In: Goldwasser, S. (ed.) CRYPTO 1988. LNCS, vol. 403, pp. 27–35. Springer, New York (1990). https://doi.org/10.1007/0-387-34799-2_3
9. Benhamouda, F., Degwekar, A., Ishai, Y., Rabin, T.: On the local leakage resilience of linear secret sharing schemes. In: Shacham, H., Boldyreva, A. (eds.) CRYPTO 2018. LNCS, vol. 10991, pp. 531–561. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-96884-1_18
10. Blakley, G.: Safeguarding cryptographic keys. In: Proceedings of the 1979 AFIPS National Computer Conference. AFIPS Press (1979)
11. Brian, G., Faonio, A., Obremski, M., Simkin, M., Venturi, D.: Non-malleable secret sharing against bounded joint-tampering attacks in the plain model. In: Micciancio, D., Ristenpart, T. (eds.) CRYPTO 2020. LNCS, vol. 12172, pp. 127–155. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-56877-1_5
12. Brian, G., Faonio, A., Venturi, D.: Continuously non-malleable secret sharing for general access structures. In: Hofheinz, D., Rosen, A. (eds.) TCC 2019. LNCS, vol. 11892, pp. 211–232. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-36033-7_8
13. Chattopadhyay, E., et al.: Extractors and secret sharing against bounded collusion protocols. In: 61st IEEE Annual Symposium on Foundations of Computer Science, FOCS 2020 (2020). https://doi.org/10.1109/FOCS46700.2020.00117
14. Chaum, D., Crépeau, C., Damgård, I.: Multiparty unconditionally secure protocols (extended abstract). In: Simon, J. (ed.) Proceedings of the 20th Annual ACM Symposium on Theory of Computing, Chicago, Illinois, USA, 2–4 May 1988, pp. 11–19. ACM (1988). https://doi.org/10.1145/62212.62214
15. Davì, F., Dziembowski, S., Venturi, D.: Leakage-resilient storage. In: 7th International Conference on Security and Cryptography for Networks, SCN 2010 (2010). https://doi.org/10.1007/978-3-642-15317-4_9
16. Dodis, Y., Ostrovsky, R., Reyzin, L., Smith, A.: Fuzzy extractors: how to generate strong keys from biometrics and other noisy data. SIAM J. Comput. **38**(1), 97–139 (2008), arXiv:cs/0602007
17. Dziembowski, S., Pietrzak, K.: Intrusion-resilient secret sharing. In: Proceedings of the 48th Annual IEEE Symposium on Foundations of Computer Science, FOCS 2007 (2007). https://doi.org/10.1109/FOCS.2007.35
18. Faonio, A., Venturi, D.: Non-malleable secret sharing in the computational setting: adaptive tampering, noisy-leakage resilience, and improved rate. In: Boldyreva, A., Micciancio, D. (eds.) CRYPTO 2019. LNCS, vol. 11693, pp. 448–479. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-26951-7_16
19. Faust, S., Rabin, T., Reyzin, L., Tromer, E., Vaikuntanathan, V.: Protecting circuits from leakage: the computationally-bounded and noisy cases. In: Gilbert, H. (ed.) EUROCRYPT 2010. LNCS, vol. 6110, pp. 135–156. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-13190-5_7
20. Goyal, V., Kumar, A.: Non-malleable secret sharing. In: Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2018 (2018). https://doi.org/10.1145/3188745.3188872
21. Guruswami, V., Umans, C., Vadhan, S.P.: Unbalanced expanders and randomness extractors from Parvaresh-Vardy codes. In: IEEE Conference on Computational Complexity, pp. 96–108 (2007)
22. Guruswami, V., Wootters, M.: Repairing reed-solomon codes. In: Proceedings of the Forty-eighth Annual ACM Symposium on Theory of Computing. STOC 2016. ACM, New York (2016). https://doi.org/10.1145/2897518.2897525

23. Ishai, Y., Sahai, A., Wagner, D.: Private circuits: securing hardware against probing attacks. In: Boneh, D. (ed.) CRYPTO 2003. LNCS, vol. 2729, pp. 463–481. Springer, Heidelberg (2003). https://doi.org/10.1007/978-3-540-45146-4_27
24. Kocher, P.C.: Timing attacks on implementations of Diffie-Hellman, RSA, DSS, and other systems. In: Koblitz, N. (ed.) CRYPTO 1996. LNCS, vol. 1109, pp. 104–113. Springer, Heidelberg (1996). https://doi.org/10.1007/3-540-68697-5_9
25. Kumar, A., Meka, R., Sahai, A.: Leakage-resilient secret sharing against colluding parties. In: 60th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2019 (2019). https://doi.org/10.1109/FOCS.2019.00045
26. Lin, F., Cheraghchi, M., Guruswami, V., Safavi-Naini, R., Wang, H.: Non-malleable secret sharing against affine tampering. CoRR abs/1902.06195 (2019). http://arxiv.org/abs/1902.06195
27. Liu, F.-H., Lysyanskaya, A.: Tamper and leakage resilience in the split-state model. In: Safavi-Naini, R., Canetti, R. (eds.) CRYPTO 2012. LNCS, vol. 7417, pp. 517–532. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-32009-5_30
28. Nisan, N., Zuckerman, D.: Randomness is linear in space. J. Comput. Syst. Sci. **52**(1), 43–53 (1996)
29. Rothblum, G.N.: How to compute under $\mathcal{AC}^0$ leakage without secure hardware. In: Safavi-Naini, R., Canetti, R. (eds.) CRYPTO 2012. LNCS, vol. 7417, pp. 552–569. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-32009-5_32
30. Shamir, A.: How to share a secret. Commun. ACM **22**(11), 612–613 (1979)
31. Srinivasan, A., Vasudevan, P.N.: Leakage resilient secret sharing and applications. In: Boldyreva, A., Micciancio, D. (eds.) CRYPTO 2019. LNCS, vol. 11693, pp. 480–509. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-26951-7_17
32. Vadhan, S.: Pseudorandomness. Foundations and Trends in Theoretical Computer Science. Now Publishers (2012). http://people.seas.harvard.edu/~salil/pseudorandomness/
33. Zimand, M.: Exposure-resilient extractors. In: 21st Annual IEEE Conference on Computational Complexity (CCC 2006). IEEE Computer Society (2006). https://doi.org/10.1109/CCC.2006.19