An abridged version of this paper appears in the proceedings of the 40th Annual International Cryptology Conference (CRYPTO 2020). This is the full version.

Non-Malleable Secret Sharing against Bounded Joint-Tampering Attacks in the Plain Model

Gianluca Brian
Sapienza University of Rome

Antonio Faonio[*]
IMDEA Software Institute

Maciej Obremski[†]
National University of Singapore

Mark Simkin[‡]
Aarhus University

Daniele Venturi
Sapienza University of Rome

June 21, 2020

Abstract

Secret sharing enables a dealer to split a secret into a set of shares, in such a way that certain authorized subsets of share holders can reconstruct the secret, whereas all unauthorized subsets cannot. Non-malleable secret sharing (Goyal and Kumar, STOC 2018) additionally requires that, even if the shares have been tampered with, the reconstructed secret is either the original or a completely unrelated one.

In this work, we construct non-malleable secret sharing tolerating p-time *joint-tampering* attacks in the plain model (in the computational setting), where the latter means that, for any $p > 0$ fixed *a priori*, the attacker can tamper with the same target secret sharing up to p times. In particular, assuming one-to-one one-way functions, we obtain:

- A secret sharing scheme for threshold access structures which tolerates joint p-time tampering with subsets of the shares of maximal size (*i.e.*, matching the privacy threshold of the scheme). This holds in a model where the attacker commits to a partition of the shares into non-overlapping subsets, and keeps tampering jointly with the shares within such a partition (so-called *selective partitioning*).
- A secret sharing scheme for general access structures which tolerates joint p-time tampering with subsets of the shares of size $O(\sqrt{\log n})$, where n is the number of parties. This holds in a stronger model where the attacker is allowed to adaptively change the partition within each tampering query, under the restriction that once a subset of the shares has been tampered with jointly, that subset is always either tampered jointly or not modified by other tampering queries (so-called *semi-adaptive partitioning*).

At the heart of our result for selective partitioning lies a new technique showing that every one-time *statistically* non-malleable secret sharing against joint tampering is in fact *leakage-resilient* non-malleable (*i.e.*, the attacker can leak jointly from the shares prior to tampering).

[*] Research leading to these results has been supported by the Spanish Government under projects SCUM (ref. RTI2018-102043-B-I00), CRYPTOEPIC (ref. EUR2019-103816), and SECURITAS (ref. RED2018-102321-T), by the Madrid Regional Government under project BLOQUES (ref. S2018/TCS-4339).

[†] Supported by MOE2019-T2-1-145 Foundations of quantum-safe cryptography.

[‡] Supported by the European Research Council (ERC) under the European Unions's Horizon 2020 research and innovation programme under grant agreement No 669255 (MPCPRO), grant agreement No 803096 (SPEC), Danish Independent Research Council under Grant-ID DFF-6108-00169 (FoCC), and the Concordium Blockhain Research Center.

We believe this may be of independent interest, and in fact we show it implies lower bounds on the share size and randomness complexity of statistically non-malleable secret sharing against *independent* tampering.

Keywords: secret sharing – non-malleability – joint tampering.

Contents

1 Introduction

In the past 40 years, secret sharing [Sha79, Bla79] became one of the most fundamental cryptographic primitives. Secret sharing schemes allow a trusted dealer to split a message m into shares $s_1, \ldots, s_n$ and distribute them among n participants, such that only certain authorized subsets of share holders are allowed to recover m. The collection $\mathcal{A}$ of authorized subsets is called the *access structure*. The most basic security guarantee is that any *unauthorized* subset outside $\mathcal{A}$ collectively has no information about the shared message. Shamir [Sha79] and Blakley [Bla79] showed how to construct secret sharing schemes with information-theoretic security, and Krawczyk [Kra94] presented the first computationally-secure construction with improved efficiency parameters.

Non-malleable secret sharing. A long line of research [RB89, CDV94, GK18a, GK18b, BS19, ADN$^+$19a, FV19, SV19, KMS19, BFV19, CL18] has focused on different settings with active adversaries that were allowed to tamper with the shares in one or another way. In verifiable secret sharing [RB89] the dealer is considered to be untrusted and the share holders want to ensure they hold shares of a consistent secret. In robust secret sharing [CDV94] some parties may act maliciously and try to prevent the correct reconstruction of the shared secret by providing incorrect shares. It is well known that robust secret sharing is impossible when more than half of the parties are malicious.

A recent line of works considers an adversary that has some form of restricted access to *all* shares. In non-malleable secret sharing [GK18a] the adversary can partition the shares in disjoint sets and can then independently tamper with each set of shares. Security guarantees that whatever is reconstructed from the tampered shares is either the original secret, or a completely unrelated value. Most previous works have focused on the setting of independent tampering [GK18a, GK18b, BS19, ADN$^+$19a, FV19, SV19, KMS19, BFV19], where the adversary is only allowed to tamper with each share *independently*. Only a few papers [GK18a, GK18b, CL18, BFV19] have considered the stronger setting where the adversary is allowed to tamper with subsets of shares *jointly*.

Continuous non-malleability. The first notions of non-malleability only focused on security against a *single* round of tampering. A natural extension of this setting is to consider adversaries that may perform several rounds of tampering attacks on a secret sharing scheme. Badrinarayanan and Srinivasan [BS19] and Aggarwal *et al.* [ADN$^+$19a] considered p-time tampering attacks in the information-theoretic setting, where p must be *a-priori* bounded. The works of Faonio and Venturi [FV19] and Brian, Faonio and Venturi [BFV19] considered *continuous*, i.e., poly-many tampering attacks in the computational setting. It is well known that cryptographic assumptions are inherent in the latter case [FMNV14, BS19, FV19].

An important limitation of all works mentioned above is that, with the exception of [BFV19], they only consider the setting of independent tampering. Brian Faonio, and Venturi [BFV19] achieve continuous non-malleability against joint tampering, where each tampering function can tamper with $O(\log n)$-large sets of shares assuming a trusted setup in the form of a common reference string. This leads to the following question:

Can we obtain continuously non-malleable secret sharing against joint tampering in the plain model?

1.1 Our Contributions

In this work, we make progress towards answering the above question. Our main contribution is a general framework for reducing *computational* p-time non-malleability against joint tampering to *statistical* one-time non-malleability against joint tampering. Our framework encompasses the following models:

- **Selective partitioning.** Here, the adversary has to initially fix any k-sized partition[1] of the n shares, at the beginning of the experiment. Afterwards, the adversary can tamper p times with the shares within each subset in a joint manner. We call this notion k-joint p-time non-malleability under *selective partitioning.*

- **Semi-adaptive partitioning.** In this setting, the adversary can adaptively choose different k-sized partitions for each tampering query. However, once a subset of the shares has been tampered with jointly, that subset is always either tampered jointly or not modified by other tampering queries. We call this notion k-joint p-time non-malleability under *semi-adaptive partitioning.*

Combining known constructions of one-time statistically non-malleable secret sharing schemes against joint tampering [GK18a, GK18b, CL18] with a new secret sharing scheme that we present in this work, we obtain the following result:

Theorem 1 (Main Theorem, Informal). *Assuming the existence of one-to-one one-way functions, there exist:*

(i) A τ-out-of-n secret sharing scheme satisfying k-joint p-time non-malleability under selective partitioning,[2] *for any $\tau \leq n$, $k \leq \tau - 1$, and $p > 0$.*

(ii) An (n, τ)-ramp[3] *secret sharing scheme* with binary shares *satisfying k-joint p-time non-malleability under selective partitioning, for $\tau = n - n^{\beta}$, $k \leq \tau - 1$, $\beta < 1$, and $p \in O(\sqrt{n})$.*

(iii) A secret sharing scheme satisfying k-joint p-time non-malleability under semi-adaptive partitioning, for $k \in O(\sqrt{\log n})$ and $p > 0$, and for any access structure that can be described by a polynomial-size monotone span program for which authorized sets have size greater than k.

1.2 Technical Overview

Our initial observation is that a slight variant of a transformation by Ostrovsky *et al.* [OPVV18] allows to turn a *bounded leakage-resilient*, statistically one-time non-malleable secret sharing Σ into a *bounded-time* non-malleable secret sharing Σ^* against joint tampering. Bounded leakage resilience here means that, prior to tampering, the attacker may also repeatedly leak information jointly from the shares of Σ, as long as the overall leakage is bounded.

In the setting of joint tampering under selective partitioning, the leakage resilience property of Σ has to hold w.r.t. the same partition used for tampering. For joint tampering under semi-adaptive partitioning, we need Σ to be leakage-resilient under a semi-adaptive choice of the partitions too. A nice feature of this transformation is that it only requires perfectly binding

[1] This a sequence of non-overlapping subsets $\mathcal{B}_1, \ldots, \mathcal{B}_t$ covering $[n]$, such that each $\mathcal{B}_i$ has size at most k.

[2] Here, we inherit a few restrictions from [GK18a]. Namely, the attacker is allowed to tamper jointly using a partition of a minimal reconstruction set in subsets of different sizes. We can remove these restrictions relying on the scheme from [GK18b], which however only works for the n-out-of-n access structure.

[3] This means privacy holds with threshold τ, but all of the n shares are required to reconstruct the message.

commitments, which can be built from injective one-way functions. Moreover, it preserves the access structure of the underlying secret sharing scheme Σ.

Given the above result, we can focus on the simpler task of constructing bounded leakage-resilient, statistically one-time non-malleable secret sharing, instead of directly attempting to construct their multi-time counterparts. We show different ways of doing that for both settings of selective and semi-adaptive partitioning.

Selective partitioning. First, we show that every statistically one-time non-malleable secret sharing scheme Σ is also resilient to *bounded leakage* under selective partitioning. Let ℓ be an upper bound on the total bit-length of the leakage over all shares. We use an argument reminiscent to standard complexity leveraging to prove that every one-time non-malleable secret sharing scheme with statistical security $\epsilon \in [0,1)$ is also ℓ-bounded leakage-resilient one-time non-malleable under selective partitioning with statistical security $\epsilon/2^{\ell}$. The proof roughly works as follows. Given an unbounded attacker A breaking the leakage-resilient one-time non-malleability of Σ, we construct an unbounded attacker $\hat{\mathsf{A}}$ against one-time non-malleability of Σ (without leakage). The challenge is how $\hat{\mathsf{A}}$ can answer the leakage queries done by A. Our strategy is to simply guess the overall leakage Λ by sampling it uniformly at random, and use this guess to answer all of A's leakage queries.

The problem with this approach is that, whenever our guess was incorrect, the attacker A may notice that it is being used in a simulation and start behaving arbitrarily. We solve this issue with the help of $\hat{\mathsf{A}}$'s final tampering query. Recall that in the model of selective partitioning, all leakage queries and the tampering query, act on the same arbitrary but fixed subsets $\mathcal{B}_1, \ldots, \mathcal{B}_t$ of a k-sized partition of the shares. Hence, when A outputs its tampering query $(f_1, \ldots, f_t)$, the reduction $\hat{\mathsf{A}}$ defines a modified tampering query $(\hat{f}_1, \ldots, \hat{f}_t)$ that first checks whether the guessed leakage from each subset $\mathcal{B}_i$ was correct; if not, the tampering function sets[4] the modified shares within $\mathcal{B}_i$ to $\perp$, else it acts identically to f_i. This strategy ensures that our reduction either performs a correct simulation or destroys the secret. In turn, destroying the secret whenever we guessed incorrectly implies that the success probability of $\hat{\mathsf{A}}$ is exactly that of A times the probability of guessing the leakage correctly, which is $2^{-\ell}$.

By plugging the schemes from [GK18a, Thm. 2], [GK18b, Thm. 6], and [CL18, Thm. 3], together with our refined analysis of the transformation by Ostrovsky *et al.* [OPVV18], the above insights directly imply items i and ii of Thm. 1.

Semi-adaptive partitioning. Unfortunately, the argument for showing that one-time non-malleability implies bounded leakage resilience breaks in the setting of adaptive (or even semi-adaptive) partitioning. Intuitively, the problem is that the adversary can leak jointly from adaptively chosen partitions, and thus it is unclear how the reduction can check whether the simulated leakage was correct using a single tampering query.

Hence, we take a different approach. We directly construct a bounded leakage-resilient, statistically one-time non-malleable secret sharing scheme for general access structures. Our construction Σ combines a 2-out-of-2 non-malleable secret sharing scheme Σ_2 with two auxiliary leakage-resilient secret sharing schemes Σ_0 and Σ_1 realizing different access structures. When taking Σ_0 to be the secret sharing scheme from [KMS19, Thm. 1], our construction achieves k-joint bounded leakage-resilient statistical one-time non-malleability under semi-adaptive partitioning for $k \in O(\sqrt{\log n})$. This implies item iii of Thm. 1. We refer the reader directly to §5 for a thorough description of our new secret sharing scheme and its security analysis.

[4] We assume that the reconstruction algorithm outputs $\perp$ whenever one of the input shares is set to $\perp$. As we will see later, this is without loss of generality.

Lower bounds. Our complexity leveraging argument implies that every statistically one-time non-malleable secret sharing scheme against *independent* tampering with the shares is also statistically bounded leakage resilient against independent leakage (and no tampering).

By invoking a recent result of Nielsen and Simkin [NS20], we immediately obtain lower bounds on the share size and randomness complexity of any statistically one-time non-malleable secret sharing scheme against *independent* tampering.

1.3 Related Works

Non-malleable secret sharing is intimately related to non-malleable codes [DPW10]. The difference between the two lies in the privacy property: While any non-malleable code in the split-state model [DPW10, LL12, DKO13, CG14, FMNV14, ADKO15a, ADKO15b, AAG+16, CGL16, Li17, AKO17, OPVV18, FNSV18, AO19, CFV19] is also a 2-out-of-2 secret sharing [DKO13], for any $n \geq 3$ there are n-split-state non-malleable codes that are not private.

Continuously non-malleable codes in the n-split-state model are currently known for $n = 8$ [ADN+19b] (with statistical security), and for $n = 2$ [FMNV14, FNSV18, OPVV18, CFV19] (with computational security).

Non-malleable secret sharing schemes have useful cryptographic applications, such as non-malleable message transmission [GK18a] and continuously non-malleable threshold signatures [ADN+19a, FV19].

1.4 Paper Organization

The rest of this paper is organized as follows. In §2, we recall a few standard definitions. In §3, we define our model of k-joint non-malleability under selective and semi-adaptive partitioning.

In §4 and §5, we describe our constructions of bounded leakage-resilient statistically one-time non-malleable secret sharing schemes under selective and semi-adaptive partitioning. The lower bounds for non-malleable secret sharing, and the compiler for achieving p-time non-malleability against joint tampering are presented in §6. Finally, in §7, we conclude the paper with a list of open problems for further research.

2 Preliminaries

2.1 Standard Notation

For a string $x \in \{0,1\}^*$, we denote its length by $|x|$; if $\mathcal{X}$ is a set, $|\mathcal{X}|$ represents the number of elements in $\mathcal{X}$. We denote by $[n]$ the set $\{1,\ldots,n\}$. For a set of indices $\mathcal{I} = (i_1,\ldots,i_t)$ and a vector $x = (x_1,\ldots,x_n)$, we write $x_{\mathcal{I}}$ to denote the vector $(x_{i_1},\ldots,x_{i_t})$. When x is chosen randomly in $\mathcal{X}$, we write $x \leftarrow\!\!\$\, \mathcal{X}$. When A is a randomized algorithm, we write $y \leftarrow\!\!\$\, \mathsf{A}(x)$ to denote a run of A on input x (and implicit random coins r) and output y; the value y is a random variable and $\mathsf{A}(x;r)$ denotes a run of A on input x and randomness r. An algorithm A is *probabilistic polynomial-time* (PPT for short) if A is randomized and for any input $x, r \in \{0,1\}^*$, the computation of $\mathsf{A}(x;r)$ terminates in a polynomial number of steps (in the size of the input).

Negligible functions. We denote with $\lambda \in \mathbb{N}$ the security parameter. A function p is *polynomial* (in the security parameter), denoted $p \in \mathsf{poly}(\lambda)$, if $p(\lambda) \in O(\lambda^c)$ for some constant $c > 0$. A function $\nu : \mathbb{N} \to [0,1]$ is *negligible* (in the security parameter) if it vanishes faster than the inverse of any polynomial in λ, *i.e.* $\nu(\lambda) \in O(1/p(\lambda))$ for all positive polynomials $p(\lambda)$. We often write $\nu(\lambda) \in \mathsf{negl}(\lambda)$ to denote that $\nu(\lambda)$ is negligible. Unless stated otherwise, throughout the paper, we implicitly assume that the security parameter is given as input (in unary) to all algorithms.

Random variables. For a random variable $\mathbf{X}$, we write $\mathbb{P}[\mathbf{X} = x]$ for the probability that $\mathbf{X}$ takes on a particular value $x \in \mathcal{X}$, with $\mathcal{X}$ being the set where $\mathbf{X}$ is defined. The statistical distance between two random variables $\mathbf{X}$ and $\mathbf{Y}$ over the same set $\mathcal{X}$ is defined as

$$\Delta(\mathbf{X}, \mathbf{Y}) := \frac{1}{2} \sum_{x \in \mathcal{X}} |\mathbb{P}[\mathbf{X} = x] - \mathbb{P}[\mathbf{Y} = x]|\,.$$

Given two ensembles $\mathbf{X} = \{\mathbf{X}_\lambda\}_{\lambda \in \mathbb{N}}$ and $\mathbf{Y} = \{\mathbf{Y}_\lambda\}_{\lambda \in \mathbb{N}}$, we write $\mathbf{X} \equiv \mathbf{Y}$ to denote that they are identically distributed, $\mathbf{X} \overset{\mathsf{s}}{\approx} \mathbf{Y}$ to denote that they are *statistically close*, i.e. $\Delta(\mathbf{X}_\lambda, \mathbf{Y}_\lambda) \in \mathsf{negl}(\lambda)$, and $\mathbf{X} \overset{\mathsf{c}}{\approx} \mathbf{Y}$ to denote that they are *computationally indistinguishable*, *i.e.* for all PPT distinguishers D:

$$|\mathbb{P}[\mathsf{D}(\mathbf{X}_\lambda) = 1] - \mathbb{P}[\mathsf{D}(\mathbf{Y}_\lambda) = 1]| \in \mathsf{negl}(\lambda)\,.$$

Sometimes we explicitly denote by $\mathbf{X} \overset{\mathsf{s}}{\approx}_\epsilon \mathbf{Y}$ the fact that $\Delta(\mathbf{X}_\lambda, \mathbf{Y}_\lambda) \le \epsilon$ for a parameter $\epsilon = \epsilon(\lambda)$. We also extend the notion of computational indistinguishability to the case of interactive experiments (a.k.a. games) featuring an adversary A. In particular, let $\mathbf{G}_\mathsf{A}(\lambda)$ be the random variable corresponding to the output of A at the end of the experiment, where wlog. we may assume A outputs a decision bit. Given two experiments $\mathbf{G}_\mathsf{A}(\lambda, 0)$ and $\mathbf{G}_\mathsf{A}(\lambda, 1)$, we write $\{\mathbf{G}_\mathsf{A}(\lambda, 0)\}_{\lambda \in \mathbb{N}} \overset{\mathsf{c}}{\approx} \{\mathbf{G}_\mathsf{A}(\lambda, 1)\}_{\lambda \in \mathbb{N}}$ as a shorthand for

$$|\mathbb{P}[\mathbf{G}_\mathsf{A}(\lambda, 0) = 1] - \mathbb{P}[\mathbf{G}_\mathsf{A}(\lambda, 1) = 1]| \in \mathsf{negl}(\lambda)\,.$$

The above naturally generalizes to statistical distance, which we denote by $\Delta(\mathbf{G}_\mathsf{A}(\lambda, 0), \mathbf{G}_\mathsf{A}(\lambda, 1))$, in case of *unbounded* adversaries.

We recall a lemma from Dziembowski and Pietrzak [DP07]:

Lemma 1. *Let $\mathbf{X}$ and $\mathbf{Y}$ be two independent random variables, and $\mathcal{O}_{\mathsf{leak}}(\cdot, \cdot)$ be an oracle that upon input arbitrary functions (g_0, g_1) returns $(g_0(\mathbf{X}), g_1(\mathbf{Y}))$. Then, for any adversary A outputting $\mathbf{Z} \leftarrow\!\!\$\ \mathsf{A}^{\mathcal{O}_{\mathsf{leak}}(\cdot,\cdot)}$, it holds that the random variables $\mathbf{X}|\mathbf{Z}$ and $\mathbf{Y}|\mathbf{Z}$ are independent.*

2.2 Secret Sharing Schemes

An n-party secret sharing scheme Σ consists of polynomial-time algorithms $(\mathsf{Share}, \mathsf{Rec})$ specified as follows. The randomized sharing algorithm Share takes a message $m \in \mathcal{M}$ as input and outputs n shares $s_1, \ldots, s_n$, where each $s_i \in \mathcal{S}_i$. The deterministic algorithm Rec takes some number of shares as input and outputs a value in $\mathcal{M} \cup \{\perp\}$. We define $\mu := \log|\mathcal{M}|$ and $\sigma_i := \log|\mathcal{S}_i|$ respectively, to be the bit length of the message and of the ith share.

Which subsets of shares are authorized to reconstruct the secret and which are not is defined via an *access structure*, which is the set of all authorized subsets.

Definition 1 (Access structure)**.** We say that $\mathcal{A}$ is an access structure for n parties if $\mathcal{A}$ is a monotone class of subsets of $[n]$, *i.e.*, if $\mathcal{I}_1 \in \mathcal{A}$ and $\mathcal{I}_1 \subseteq \mathcal{I}_2$, then $\mathcal{I}_2 \in \mathcal{A}$. We call authorized or qualified any set $\mathcal{I} \in \mathcal{A}$, and unauthorized or unqualified any other set. We say that an authorized set $\mathcal{I} \in \mathcal{A}$ is minimal if any proper subset of $\mathcal{I}$ is unauthorized, *i.e.*, if $\mathcal{U} \subsetneq \mathcal{I}$, then $\mathcal{U} \notin \mathcal{A}$.

Intuitively, a perfectly secure secret sharing scheme must be such that all qualified subsets of players can efficiently reconstruct the secret, whereas all unqualified subsets have no information (possibly in a computational sense) about the secret.

Definition 2 (Secret sharing scheme)**.** Let $n \in \mathbb{N}$ and $\mathcal{A}$ be an access structure for n parties. We say that $\Sigma = (\mathsf{Share}, \mathsf{Rec})$ is a secret sharing scheme realizing access structure $\mathcal{A}$ with message space $\mathcal{M}$ and share space $\mathcal{S} = \mathcal{S}_1 \times \ldots \times \mathcal{S}_n$ if it is an n-party secret sharing with the following properties.

(i) **Correctness:** For all $\lambda \in \mathbb{N}$, all messages $m \in \mathcal{M}$ and all authorized subsets $\mathcal{I} \in \mathcal{A}$, we have that $\mathsf{Rec}((\mathsf{Share}(m))_{\mathcal{I}}) = m$ with overwhelming probability over the randomness of the sharing algorithm.

(ii) **Privacy:** For all PPT adversaries A, all pairs of messages $m_0, m_1 \in \mathcal{M}$ and all unauthorized subsets $\mathcal{U} \notin \mathcal{A}$, we have that

$$\{(\mathsf{Share}(1^\lambda, m_0))_{\mathcal{U}}\}_{\lambda \in \mathbb{N}} \overset{\mathsf{c}}{\approx} \{(\mathsf{Share}(1^\lambda, m_1))_{\mathcal{U}}\}_{\lambda \in \mathbb{N}}.$$

If the above ensembles are statistically close (resp. identically distributed), we speak of *statistical* (resp. *perfect*) privacy.

2.3 Non-Interactive Commitments

A non-interactive commitment scheme Commit is a randomized algorithm taking as input a message $m \in \mathcal{M}$ and outputting a value $c = \mathsf{Commit}(m; r)$ called commitment, using random coins $r \in \mathcal{R}$. The pair (m, r) is called the opening.

Intuitively, a secure commitment satisfies two properties called binding and hiding. The first property says that it is hard to open a commitment in two different ways. The second property says that a commitment hides the underlying message. The formal definition follows.

Definition 3 (Binding)**.** We say that a non-interactive commitment scheme Commit is *computationally binding* if for all PPT adversaries A, all messages $m \in \mathcal{M}$, and all random coins $r \in \mathcal{R}$, the following probability is negligible:

$$\mathbb{P}\left[m' \neq m \wedge \mathsf{Commit}(m'; r') = \mathsf{Commit}(m; r) : \ (m', r') \leftarrow\!\!\$\ \mathsf{A}(m, r)\right].$$

If the above holds even in the case of unbounded adversaries, we say that Commit is *statistically binding*. Finally, if the above probability is exactly 0 for all adversaries (*i.e.*, each commitment can be opened to at most a single message), then we say that Commit is *perfectly binding*.

Definition 4 (Hiding)**.** We say that a non-interactive commitment scheme Commit is *computationally hiding* if, for all $m_0, m_1 \in \mathcal{M}$, it holds that

$$\left\{\mathsf{Commit}(1^\lambda; m_0)\right\}_{\lambda \in \mathbb{N}} \overset{\mathsf{c}}{\approx} \left\{\mathsf{Commit}(1^\lambda; m_1)\right\}_{\lambda \in \mathbb{N}}.$$

In case the above ensembles are statistically close (resp. identically distributed), we say that Commit is *statistically* (resp. *perfectly*) hiding.

3 Our Leakage and Tampering Model

In this section we define various notions of non-malleability against joint tampering and leakage for secret sharing. Very roughly, in our model the attacker is allowed to partition the set of share holders into t (non-overlapping) blocks with size at most k, covering the entire set $[n]$. This is formalized through the notion of a k-sized partition.

Definition 5 (k-sized partition)**.** Let $n, k, t \in \mathbb{N}$. We call $\mathcal{B} = (\mathcal{B}_1, \ldots, \mathcal{B}_t)$ a k-sized partition of $[n]$ when:

(i) $\bigcup_{i=1}^{t} \mathcal{B}_i = [n]$;

(ii) $\forall i_1, i_2 \in [t]$ such that $i_1 \neq i_2$, $\mathcal{B}_{i_1} \cap \mathcal{B}_{i_2} = \emptyset$.

(iii) $\forall i \in [t] : \ |\mathcal{B}_i| \leq k$.

Let $\mathcal{B} = (\mathcal{B}_1, \ldots, \mathcal{B}_t)$ be a k-sized partition of $[n]$. To define non-malleability, we consider an adversary A interacting with a target secret sharing $s = (s_1, \ldots, s_n)$ via the following queries:

- **Leakage queries.** For each $i \in [t]$, the attacker can leak jointly from the shares $s_{\mathcal{B}_i}$. This can be done repeatedly and in an adaptive[5] fashion, as long as the total number of bits that the adversary leaks from each share does not exceed $\ell \in \mathbb{N}$.

- **Tampering queries.** For each $i \in [t]$, the attacker can tamper jointly with the shares $s_{\mathcal{B}_i}$. Each such query yields mauled shares $(\tilde{s}_1, \ldots, \tilde{s}_n)$, for which the adversary is allowed to see the corresponding reconstructed message w.r.t. a reconstruction set $\mathcal{T} \in \mathcal{A}$ of his choice. This can be done for at most $p \in \mathbb{N}$ times, and in an adaptive fashion.

Depending on the partition $\mathcal{B}$ being fixed, or chosen adaptively with each leakage/tampering query, we obtain two different flavors of non-malleability, as defined in the following subsections.

3.1 Selective Partitioning

Here, we restrict the adversary to jointly leak from and tamper with subsets of shares belonging to a fixed partition of $[n]$.

Definition 6 (Selective bounded-leakage and tampering admissible adversary)**.** Let $n, k, t, \ell, p \in \mathbb{N}$, and fix an arbitrary message space $\mathcal{M}$, sharing space $\mathcal{S} = \mathcal{S}_1 \times \cdots \times \mathcal{S}_n$, and access structure $\mathcal{A}$ for n parties. We say that a (possibly unbounded) adversary A is selective k-joint ℓ-bounded leakage p-tampering admissible (selective (k, ℓ, p)-BLTA for short) if, for every fixed k-sized partition $(\mathcal{B}_1, \ldots, \mathcal{B}_t)$ of $[n]$, A satisfies the following conditions:

- A outputs a sequence of poly-many leakage queries $(g_1^{(q)}, \ldots, g_t^{(q)})$, such that for all $q \in \mathsf{poly}(\lambda)$ and all $i \in [t]$,
$$g_i^{(q)} : \bigtimes_{j \in \mathcal{B}_i} \mathcal{S}_j \to \{0,1\}^{\ell_i^{(q)}},$$
where $\ell_i^{(q)}$ is the length of the output $\Lambda_i^{(q)}$ of $g_i^{(q)}$. The only restriction is that $|\Lambda| \leq \ell$, where Λ is the string containing the total leakage performed (over all queries).

- A outputs a sequence of tampering queries $(\mathcal{T}^{(q)}, (f_1^{(q)}, \ldots, f_t^{(q)}))$, such that, for all $q \in [p]$, and for all $i \in [t]$, it holds that
$$f_i^{(q)} : \bigtimes_{j \in \mathcal{B}_i} \mathcal{S}_j \to \bigtimes_{j \in \mathcal{B}_i} \mathcal{S}_j \qquad \text{and} \qquad \mathcal{T}^{(q)} \cap \mathcal{B}_i \neq \emptyset,$$
and moreover $\mathcal{T}^{(q)} \in \mathcal{A}$ is a minimal authorized subset.

- All queries performed by A are chosen adaptively, *i.e.* each query may depend on the information obtained from all the previous queries.

- If $p > 0$, the last query performed by A is a tampering query.

[5] This means that the choice of the next leakage query depends on the overall leakage so far.

Note that A can choose a different reconstruction set $\mathcal{T}^{(q)}$ with each tampering query, in a fully adaptive manner. This feature is known as *adaptive reconstruction* [FV19]. However, we consider the following two restrictions (that were not present in previous works): (i) Each set $\mathcal{T}^{(q)}$ must be minimal and contain at least one mauled share from each subset $\mathcal{B}_i$; (ii) The last query asked by A is a tampering query. Looking ahead, these technical conditions are needed for the complexity leveraging argument used in Thm. 3. Note that the above restrictions are still meaningful, as they allow, *e.g.*, to capture the setting in which the attacker first leaks from all the shares and then tampers with the shares in a minimal authorized subset.

3.2 Semi-Adaptive Partitioning

Next, we generalize the above definition to the stronger setting in which the adversary is allowed to change the k-sized partition with each leakage and tampering query. Here, we do not consider the restrictions (i) and (ii) mentioned above as they are not needed for the analysis of our secret sharing scheme in §5; yet, we will need to restrict the way in which the attacker specifies the partitions corresponding to each leakage and tampering query. For this reason, we refer to our model as *semi-adaptive* partitioning.

Definition 7 (Semi-adaptive bounded-leakage and tampering admissible adversary)**.** Let $n, k, \ell, p \in \mathbb{N}$ and $\mathcal{M}, \mathcal{S}, \mathcal{A}$ as in Def. 6. We say that a (possibly unbounded) adversary A is semi-adaptive k-joint ℓ-bounded leakage p-tampering admissible (semi-adaptive (k, ℓ, p)-BLTA for short) if it satisfies the following conditions:

- A outputs a sequence of poly-many leakage queries $(\mathcal{B}^{(q)}, (g_1^{(q)}, \ldots, g_{t^{(q)}}^{(q)}))$, chosen adaptively, such that, for all $q \in \mathsf{poly}(\lambda)$, and for all $i \in [t^{(q)}]$, it holds that $\mathcal{B}^{(q)} = (\mathcal{B}_1^{(q)}, \ldots, \mathcal{B}_{t^{(q)}}^{(q)})$ is a k-sized partition of $[n]$ and
$$g_i^{(q)} : \bigtimes_{j \in \mathcal{B}_i^{(q)}} \mathcal{S}_j \to \{0,1\}^{\ell_i^{(q)}},$$
where $\ell_i^{(q)}$ is the length of the output. The only restriction is that $|\Lambda| \le \ell$, where $\Lambda = (\Lambda^{(1)}, \Lambda^{(2)}, \ldots)$ is the total leakage (over all queries).

- A outputs a sequence of p tampering queries $(\mathcal{B}^{(q)}, \mathcal{T}^{(q)}, (f_1^{(q)}, \ldots, f_t^{(q)}))$, chosen adaptively, such that, for all $q \in [p]$, and for all $i \in [t^{(q)}]$, it holds that $\mathcal{B}^{(q)}$ is a k-sized partition of $[n]$ and
$$f_i^{(q)} : \bigtimes_{j \in \mathcal{B}_i^{(q)}} \mathcal{S}_j \to \bigtimes_{j \in \mathcal{B}_i^{(q)}} \mathcal{S}_j.$$

- Given a tampering query $(\mathcal{B}, \mathcal{T}, f)$, let $\mathcal{T} = \{\beta_1, \ldots, \beta_\tau\}$ for $\tau \in \mathbb{N}$. We write $\xi(i)$ for the index such that $\beta_i \in \mathcal{B}_{\xi(i)}$; namely, the i-th share used in the reconstruction is tampered by the $\xi(i)$-th tampering function. Then:
  (i) For all leakage queries $(\mathcal{B}, g)$ and all tampering queries $(\mathcal{B}', \mathcal{T}', f')$, where $\mathcal{B} = (\mathcal{B}_1, \ldots, \mathcal{B}_t)$ and $\mathcal{B}' = (\mathcal{B}'_1, \ldots, \mathcal{B}'_{t'})$, the following holds: for all indices $i \in [t]$, either there exists $j \in \mathcal{T}'$ such that $\mathcal{B}_i \subseteq \mathcal{B}'_{\xi(j)}$, or for all $j \in \mathcal{T}'$ we have $\mathcal{B}_i \cap \mathcal{B}'_{\xi(j)} = \emptyset$.
  (ii) For any pair of tampering queries $(\mathcal{B}', \mathcal{T}', f')$ and $(\mathcal{B}'', \mathcal{T}'', f'')$, where $\mathcal{B}' = \{\mathcal{B}'_1, \ldots, \mathcal{B}'_{t'}\}$ and $\mathcal{B}'' = \{\mathcal{B}''_1, \ldots, \mathcal{B}''_{t''}\}$, the following holds: for all $i \in \mathcal{T}'$, either there exists $j \in \mathcal{T}''$ such that $\mathcal{B}'_{\xi(i)} \subseteq \mathcal{B}''_{\xi(j)}$, or for all $j \in \mathcal{T}''$ we have $\mathcal{B}'_{\xi(i)} \cap \mathcal{B}''_{\xi(j)} = \emptyset$.

```
$\mathbf{JSTamper}^{\mathcal{B},m_0,m_1}_{\Sigma,\mathsf{A}}(\lambda, b)$:
$s := (s_1, \ldots, s_n) \leftarrow\!\!\$\ \mathsf{Share}(m_b)$
$\mathtt{stop} \leftarrow \mathsf{false}$
Return $\mathsf{A}^{\mathcal{O}_{\mathsf{nmss}}(s,\mathcal{B},\cdot,\cdot),\mathcal{O}_{\mathsf{leak}}(s,\mathcal{B},\cdot)}(1^\lambda)$

$\mathbf{JATamper}^{m_0,m_1}_{\Sigma,\mathsf{A}}(\lambda, b)$:
$s := (s_1, \ldots, s_n) \leftarrow\!\!\$\ \mathsf{Share}(m_b)$
$\mathtt{stop} \leftarrow \mathsf{false}$
Return $\mathsf{A}^{\mathcal{O}_{\mathsf{nmss}}(s,\cdot,\cdot,\cdot),\mathcal{O}_{\mathsf{leak}}(s,\cdot,\cdot)}(1^\lambda)$

Oracle $\mathcal{O}_{\mathsf{leak}}(s, \mathcal{B}, (g_1, \ldots, g_t))$:
Return $g_1(s_{\mathcal{B}_1}), \ldots, g_t(s_{\mathcal{B}_t})$
```

```
Oracle $\mathcal{O}_{\mathsf{nmss}}(s, \mathcal{B}, \mathcal{T}, (f_1, \ldots, f_t))$:
If $\mathtt{stop} = \mathsf{true}$
    Return $\perp$
Else
    $\forall i \in [t] : \tilde{s}_{\mathcal{B}_i} := f_i(s_{\mathcal{B}_i})$
    $\tilde{s} = (\tilde{s}_1, \ldots, \tilde{s}_n)$
    $\tilde{m} = \mathsf{Rec}(\tilde{s}_{\mathcal{T}})$
    If $\tilde{m} \in \{m_0, m_1\}$
        Return $\diamond$
    If $\tilde{m} = \perp$
        Return $\perp$
        $\mathtt{stop} \leftarrow \mathsf{true}$
    Else return $\tilde{m}$
```

Figure 1: Experiments defining selective (**JSTamper**) and adaptive (**JATamper**) joint leakage-resilient (continuously) non-malleable secret sharing. The oracle $\mathcal{O}_{\mathsf{nmss}}$ is implicitly parameterized by the flag $\mathtt{stop}$.

Intuitively, condition (i) means that whenever the attacker leaks jointly from the shares within a subset $\mathcal{B}_i$, then for any tampering query the adversary must either tamper jointly with the shares within $\mathcal{B}_i$, or do not modify those shares at all. Condition (ii) is the same translated to the partitions corresponding to different tampering queries. Looking ahead, condition (i) is needed for the proof in §5.3, whereas condition (ii) is needed for the proof in §6.2. Note that the above restrictions are still meaningful, as they allow, *e.g.*, to capture the setting in which the attacker defines two non-overlapping[6] subsets of $[n]$ and then performs joint leakage under adaptive partitioning within the first subset and joint leakage/tampering under selective partitioning within the second subset.

3.3 The Definition

Very roughly, leakage-resilient non-malleability states that no admissible adversary, as defined above, can distinguish whether it is interacting with a secret sharing of m_0 or of m_1.

Definition 8 (Leakage-resilient non-malleability)**.** Let $n, k, \ell, p \in \mathbb{N}$ and $\epsilon \in [0, 1]$ be parameters, and $\mathcal{A}$ be an access structure for n parties. We say that $\Sigma = (\mathsf{Share}, \mathsf{Rec})$ is a k-joint ℓ-bounded leakage-resilient p-time ϵ-non-malleable secret sharing scheme realizing $\mathcal{A}$, shortened (k, ℓ, p, ϵ)-BLR-NMSS, if it is an n-party secret sharing scheme realizing $\mathcal{A}$, and additionally, for all pairs of messages $m_0, m_1 \in \mathcal{M}$, we have one of the following:

- For all *selective* (k, ℓ, p)-BLTA adversaries A, and for all k-sized partitions $\mathcal{B}$ of $[n]$,

$$\left\{\mathbf{JSTamper}^{\mathcal{B},m_0,m_1}_{\Sigma,\mathsf{A}}(\lambda, 0)\right\}_{\lambda\in\mathbb{N}} \overset{\mathsf{s}}{\approx}_{\epsilon} \left\{\mathbf{JSTamper}^{\mathcal{B},m_0,m_1}_{\Sigma,\mathsf{A}}(\lambda, 1)\right\}_{\lambda\in\mathbb{N}}. \tag{1}$$

  In this case, we speak of (k, ℓ, p, ϵ)-BLR-NMSS under *selective* partitioning.

- For all *semi-adaptive* (k, ℓ, p)-BLTA adversaries A,

$$\left\{\mathbf{JATamper}^{m_0,m_1}_{\Sigma,\mathsf{A}}(\lambda, 0)\right\}_{\lambda\in\mathbb{N}} \overset{\mathsf{s}}{\approx}_{\epsilon} \left\{\mathbf{JATamper}^{m_0,m_1}_{\Sigma,\mathsf{A}}(\lambda, 1)\right\}_{\lambda\in\mathbb{N}}. \tag{2}$$

  In this case, we speak of (k, ℓ, p, ϵ)-BLR-NMSS under *semi-adaptive* partitioning.

[6] In fact, the two subsets do not need to be fixed a priori.

Experiments $\mathbf{JSTamper}_{\Sigma,\mathsf{A}}^{\mathcal{B},m_0,m_1}(\lambda, b)$ and $\mathbf{JATamper}_{\Sigma,\mathsf{A}}^{m_0,m_1}(\lambda, b)$, for $b \in \{0,1\}$, are depicted in Fig. 1.

In case there exists $\epsilon = \epsilon(\lambda) \in \mathsf{negl}(\lambda)$ such that indistinguishability still holds computationally in the above definitions for any $p = p(\lambda) \in \mathsf{poly}(\lambda)$, and any PPT adversaries A, we call Σ bounded leakage-resilient continuously non-malleable, shortened (k, ℓ)-BLR-CNMSS, under selective/semi-adaptive partitioning.

Non-malleable secret sharing. When no leakage is allowed (*i.e.*, $\ell = 0$), we obtain the notion of non-malleable secret sharing as a special case. In particular, an adversary is k-joint p-time tampering admissible, shortened (k, p)-TA, if it is $(k, 0, p)$-BLTA. Furthermore, we say that Σ is a k-joint p-time ϵ-non-malleable secret sharing, shortened (k, p, ϵ)-NMSS, if Σ is a $(k, 0, p, \epsilon)$-BLR-NMSS scheme.

Leakage-resilient secret sharing. When no tampering is allowed (*i.e.*, $p = 0$), we obtain the notion of leakage-resilient secret sharing as a special case. In particular, an adversary is k-joint ℓ-bounded leakage admissible, shortened (k, ℓ)-BLA, if it is $(k, \ell, 0)$-BLTA. Furthermore, we say that Σ is a k-joint ℓ-bounded ϵ-leakage-resilient secret sharing, shortened (k, ℓ, ϵ)-BLRSS, if Σ is a $(k, \ell, 0, \epsilon)$-BLR-NMSS scheme.

Finally, we denote by $\mathbf{JSLeak}_{\Sigma,\mathsf{A}}^{\mathcal{B},m_0,m_1}(\lambda, b)$ and $\mathbf{JALeak}_{\Sigma,\mathsf{A}}^{m_0,m_1}(\lambda, b)$ the experiments in Def. 8 defining leakage resilience against selective and semi-adaptive partitioning respectively. However, note that when no tampering happens the conditions (i) and (ii) of Def. 7 are irrelevant, and thus we simply speak of (k, ℓ, ϵ)-BLRSS under adaptive partitioning.

Augmented leakage resilience. We also define a seemingly stronger variant of leakage-resilient secret sharing, in which A is allowed to obtain the shares within a subset of the partition $\mathcal{B}$ (in the case of selective partitioning, or any unauthorized subset of at most k shares in the case of adaptive partitioning) at the end of the experiment. In particular, in the case of selective partitioning, an *augmented* admissible adversary is an attacker $\mathsf{A}^+ = (\mathsf{A}_1^+, \mathsf{A}_2^+)$ such that:

- A_1^+ is an admissible adversary in the sense of Def. 6, the only difference being that A_1^+ outputs a tuple (α, i^*), where α is an auxiliary state, and $i^* \in [t]$;
- A_2^+ takes as input α and all the shares $s_{\mathcal{B}_{i^*}}$, and outputs a decision bit.

In case of adaptive partitioning, the definition changes as follows: the adversary A_1^+ is admissible in the sense of Def. 7 and outputs an unauthorized subset $\mathcal{U} \notin \mathcal{A}$ of size at most k instead of the index i^*, and A_2^+ takes as input the shares $s_{\mathcal{U}}$ instead of the shares $s_{\mathcal{B}_{i^*}}$.

This flavor of security is called *augmented leakage resilience*. The theorem below, which was established by [BFV19, KMS19] for the case of independent leakage, shows that any joint LRSS is also an augmented LRSS at the cost of an extra bit of leakage.

Theorem 2. *Let Σ be a $(k, \ell+1, \epsilon)$-BLRSS realizing access structure $\mathcal{A}$ under selective/adaptive partitioning. Then, Σ is an* augmented *(k, ℓ, ϵ)-BLRSS realizing $\mathcal{A}$ under selective/adaptive partitioning.*

Proof. By reduction to non-augmented leakage resilience. Let $\mathsf{A}^+ = (\mathsf{A}_1^+, \mathsf{A}_2^+)$ be a (k, ℓ, ϵ)-BLA adversary violating augmented leakage-resilience; we construct an adversary A breaking the non-augmented variant of leakage resilience. Fix $m_0, m_1 \in \mathcal{M}$ and a k-sized partition $\mathcal{B} = (\mathcal{B}_1, \ldots, \mathcal{B}_t)$. Attacker A works as follows.

- Run A_1^+ and, upon input a leakage query $(g_1, \ldots, g_t)$, forward the same query to the target leakage oracle and return the answer to A_1^+.
- Let (α, i^*) be the final output of A_1^+. Define the leakage function $\hat{g}_{i^*}^{\alpha,\mathsf{A}_2^+}$ which hard-wires α and a description of A_2^+, takes as input the shares $s_{\mathcal{B}_{i^*}}$ and returns the decision bit $b' \leftarrow\!\!\$\ \mathsf{A}_2^+(\alpha, s_{\mathcal{B}_{i^*}})$.
- Forward $(\varepsilon, \ldots, \varepsilon, \hat{g}_{i^*}^{\alpha,\mathsf{A}_2^+}, \varepsilon, \ldots, \varepsilon)$ to the target leakage oracle, obtaining a bit b'.
- Output b'.

The statement follows by observing that A's simulation to A^+'s leakage queries is perfect, thus A and A^+ have the same advantage, and moreover A leaks a total of at most $\ell + 1$ bits. □

4 Selective Partitioning

In this section, we construct bounded leakage-resilient, statistically one-time non-malleable secret sharing under selective partitioning. We achieve this in two steps. First, in §4.1, we prove that every statistically one-time non-malleable secret sharing is in fact bounded leakage-resilient, statistically one-time non-malleable under selective partitioning at the price of a security loss exponential in the size of the leakage. Then, in §4.2, we provide concrete instantiations using known results from the literature.

4.1 Non-Malleability Implies Bounded Leakage Resilience

Theorem 3. *Let $\Sigma = (\mathsf{Share}, \mathsf{Rec})$ be a $(k, 1, \epsilon/2^\ell)$-NMSS realizing $\mathcal{A}$. Then, Σ is also a $(k, \ell, 1, \epsilon)$-BLR-NMSS realizing $\mathcal{A}$ under selective partitioning.*

Proof. By contradiction, assume that there exist a pair of messages $m_0, m_1 \in \mathcal{M}$, a k-partition $\mathcal{B} = (\mathcal{B}_1, \ldots, \mathcal{B}_t)$ of $[n]$, and a $(k, \ell, 1)$-BLTA unbounded adversary A such that

$$\left|\mathbb{P}\left[\mathbf{JSTamper}_{\Sigma,\mathsf{A}}^{\mathcal{B},m_0,m_1}(\lambda, 0) = 1\right] - \mathbb{P}\left[\mathbf{JSTamper}_{\Sigma,\mathsf{A}}^{\mathcal{B},m_0,m_1}(\lambda, 1) = 1\right]\right| > \epsilon.$$

Consider the following unbounded reduction $\hat{\mathsf{A}}$ trying to break $(k, 0, 1, \epsilon/2^\ell)$-non-malleability using the same partition $\mathcal{B}$, and the same messages m_0, m_1.

1. Run $\mathsf{A}(1^\lambda)$.
2. Upon input the q-th leakage query $g^{(q)} = (g_1^{(q)}, \ldots, g_t^{(q)})$, generate a uniformly random string $\Lambda^{(q)} = (\Lambda_1^{(q)}, \ldots, \Lambda_t^{(q)})$ compatible with the range of $g^{(q)}$, and output $\Lambda^{(q)}$ to A.
3. Upon input the final tampering query $f = (f_1, \ldots, f_t)$, construct the following tampering function $\hat{f} = (\hat{f}_1, \ldots, \hat{f}_t)$:
    - The function hard-wires (a description of) all the leakage functions $g^{(q)}$, the tampering query f, and the guess on the leakage $\Lambda = \Lambda^{(1)} || \Lambda^{(2)} || \cdots$.
    - Upon input the shares $(s_j)_{j \in \mathcal{B}_i}$, the function $\hat{f}_i$ checks that the guess on the leakage was correct, *i.e.* $g_i^{(q)}((s_j)_{j \in \mathcal{B}_i}) = \Lambda_i^{(q)}$ for all q. If the guess was correct, compute and output $f_i((s_j)_{j \in \mathcal{B}_i})$; else, output $\perp$.
4. Send $\hat{f}$ to the tampering oracle and pass the answer $\tilde{m} \in \mathcal{M} \cup \{\diamond, \perp\}$ to A.

5. Output the same guessing bit as A.

For the analysis, we now compute the distinguishing advantage of $\hat{\mathsf{A}}$. In particular, call $\mathbf{Miss}_b$ the event in which the guess on the leakage was wrong in experiment $\mathbf{JSTamper}_{\Sigma,\hat{\mathsf{A}}}^{\mathcal{B},m_0,m_1}(\lambda, b)$, *i.e.* there exists $i \in [t]$ such that $\hat{f}_i$ outputs $\perp$ in step 3, and call $\mathbf{Hit}_b$ its complementary event. We notice that the probability of $\mathbf{Hit}_0$ is equal to the probability of $\mathbf{Hit}_1$, since the strings $\Lambda^{(q)}$ are sampled uniformly at random:

$$\mathbb{P}[\mathbf{Hit}_b] = \sum_{\Lambda \in \{0,1\}^\ell} \mathbb{P}[\mathbf{U}_\ell = \Lambda \wedge g(\mathbf{S}^b) = \Lambda] = 2^{-\ell} \sum_{\Lambda \in \{0,1\}^\ell} \mathbb{P}[g(\mathbf{S}^b) = \Lambda] = 2^{-\ell},$$

where $\mathbf{S}^b$ is the random variable corresponding to $\mathsf{Share}(m_b)$, $\mathbf{U}_\ell$ is the uniform distribution over $\{0,1\}^\ell$, and g is the concatenation of all the leakage functions. Then, we can write:

$$\begin{aligned}
&\left|\mathbb{P}\left[\mathbf{JSTamper}_{\Sigma,\hat{\mathsf{A}}}^{\mathcal{B},m_0,m_1}(\lambda,0)=1\right] - \mathbb{P}\left[\mathbf{JSTamper}_{\Sigma,\hat{\mathsf{A}}}^{\mathcal{B},m_0,m_1}(\lambda,1)=1\right]\right| \\
&= \left|\mathbb{P}\left[\mathbf{Hit}_0\right]\mathbb{P}\left[\mathbf{JSTamper}_{\Sigma,\hat{\mathsf{A}}}^{\mathcal{B},m_0,m_1}(\lambda,0)=1\middle|\mathbf{Hit}_0\right]\right. && (3)\\
&\qquad - \mathbb{P}\left[\mathbf{Hit}_1\right]\mathbb{P}\left[\mathbf{JSTamper}_{\Sigma,\hat{\mathsf{A}}}^{\mathcal{B},m_0,m_1}(\lambda,1)=1\middle|\mathbf{Hit}_1\right] \\
&\qquad + \mathbb{P}\left[\mathbf{Miss}_0\right]\mathbb{P}\left[\mathbf{JSTamper}_{\Sigma,\hat{\mathsf{A}}}^{\mathcal{B},m_0,m_1}(\lambda,0)=1\middle|\mathbf{Miss}_0\right] \\
&\qquad \left. - \mathbb{P}\left[\mathbf{Miss}_1\right]\mathbb{P}\left[\mathbf{JSTamper}_{\Sigma,\hat{\mathsf{A}}}^{\mathcal{B},m_0,m_1}(\lambda,1)=1\middle|\mathbf{Miss}_1\right]\right| \\
&= 2^{-\ell}\left|\mathbb{P}\left[\mathbf{JSTamper}_{\Sigma,\hat{\mathsf{A}}}^{\mathcal{B},m_0,m_1}(\lambda,0)=1\middle|\mathbf{Hit}_0\right]\right. && (4)\\
&\qquad \left. - \mathbb{P}\left[\mathbf{JSTamper}_{\Sigma,\hat{\mathsf{A}}}^{\mathcal{B},m_0,m_1}(\lambda,1)=1\middle|\mathbf{Hit}_1\right]\right| \\
&= 2^{-\ell}\left|\mathbb{P}\left[\mathbf{JSTamper}_{\Sigma,\mathsf{A}}^{\mathcal{B},m_0,m_1}(\lambda,0)=1\right]\right. && (5)\\
&\qquad \left. -\mathbb{P}\left[\mathbf{JSTamper}_{\Sigma,\mathsf{A}}^{\mathcal{B},m_0,m_1}(\lambda,1)=1\right]\right| > \frac{\epsilon}{2^\ell}, && (6)
\end{aligned}$$

In the above derivation, Eq. (3) follows from the law of total probability, Eq. (4) comes from the fact that, when **Miss** happens, the view of A (*i.e.* the leakage Λ and the output of the tampering query) is independent[7] of the target secret sharing, and thus its distinguishing advantage is zero, and Eq. (5) follows because $\mathbb{P}[\mathbf{Hit}] = 2^{-\ell}$ and moreover, when **Hit** happens, the view of A is perfectly simulated and thus $\hat{\mathsf{A}}$ has the same distinguishing advantage of A, which is at least ϵ by assumption.

Therefore, $\hat{\mathsf{A}}$ has a distinguishing advantage of at least $\epsilon/2^\ell$. Finally, note that $\hat{\mathsf{A}}$ performs no leakage and uses only one tampering query, and thus $\hat{\mathsf{A}}$ is $(k,1)$-TA. The lemma follows. □

4.2 Instantiations

Using known constructions of one-time non-malleable secret sharing schemes against joint tampering, we obtain the following:

Corollary 1. *For every $\lambda, \ell, n \geq 0$, and every $k, \tau \geq 0$ such that $k < \tau \leq n$, there exists a τ-out-of-n secret sharing Σ that is a $(k, \ell, 1, 2^{-\lambda})$-BLR-NMSS under selective partitioning.*

[7] Here is where we use the restriction that the reconstruction set $\mathcal{T}$ must be minimal and contain at least one share from each subset $\mathcal{B}_i$; otherwise, we cannot argue that the output of the tampering query is $\perp$, and thus independent of the target.

Proof. Follows by combining Thm. 3 with the secret sharing scheme[8] of [GK18a, Thm. 4], using security parameter $\lambda' + \ell$ and choosing $\lambda \geq (\lambda' + \ell)^{\Omega(1)} - \ell$ in order to obtain

$$\epsilon = 2^{\ell} \cdot 2^{-(\lambda'+\ell)^{\Omega(1)}} \leq 2^{-\lambda}.$$

□

Corollary 2. *For every $\ell, n \geq 0$, any $\beta < 1$, and every $k, \tau \geq 0$ such that $k < \tau \leq n$, there exists an (n, τ)-ramp secret sharing Σ that is a $(k, \ell, 1, 2^{\ell} \cdot 2^{-n^{\Omega(1)}})$-BLR-NMSS under selective partitioning with binary shares.*

Proof. Follows by combining Thm. 3 with the secret sharing scheme of [CL18, Thm. 4.1]. □

5 Semi-Adaptive Partitioning

As mentioned in the introduction, the proof of Thm. 3 breaks in the setting of semi-adaptive partitioning. To overcome this issue, in §5.1, we give a direct construction of a bounded leakage-resilient, one-time statistically non-malleable secret sharing (for general access structures) under semi-adaptive partitioning. We explain the main intuition behind our design in §5.2, and formally prove security in §5.3. Finally, in §5.4, we explain how to instantiate our construction using known results from the literature.

5.1 Our New Secret Sharing Scheme

Let Σ_0 be a secret sharing realizing access structure $\mathcal{A}$, let Σ_1 be a k_1-out-of-n secret sharing, and let Σ_2 be a 2-out-of-2 secret sharing. Consider the following scheme $\Sigma = (\mathsf{Share}, \mathsf{Rec})$:

- **Algorithm** Share**:** Upon input m, first compute $(s_0, s_1) \leftarrow\!\!\$\ \mathsf{Share}_2(m)$, $(s_{0,1}, \ldots, s_{0,n}) \leftarrow\!\!\$\ \mathsf{Share}_0(s_0)$, and $(s_{1,1}, \ldots, s_{1,n}) \leftarrow\!\!\$\ \mathsf{Share}_1(s_1)$. Then set $s_i := (s_{0,i}, s_{1,i})$ for all $i \in [n]$, and output $(s_1, \ldots, s_n)$.
- **Algorithm** Rec**:** Upon input $(s_i)_{i \in \mathcal{I}}$, parse $s_i = (s_{0,i}, s_{1,i})$ and $\mathcal{I} = \{i_1, \ldots, i_{|\mathcal{I}|}\}$, and define $\mathcal{I}_{|k_1} := \{i_1, \ldots, i_{k_1}\}$; compute $s_1 = \mathsf{Rec}_1((s_{1,i})_{i \in \mathcal{I}_{|k_1}})$ and $s_0 = \mathsf{Rec}_0((s_{0,i})_{i \in \mathcal{I}})$, and finally output $m' = \mathsf{Rec}_2((s_0, s_1))$.

With the above defined scheme, we achieve the following:

Theorem 4. *Let $n, k(\lambda), \ell(\lambda), \sigma_0(\lambda) \in \mathbb{N}$ and $\epsilon_0, \epsilon_1, \epsilon_2 \in [0, 1]$ be parameters, and set $k_1 := \sqrt{k}$, $\ell_0 := \ell + 1$ and $\ell_1 := \ell + n \cdot \sigma_0$. Let $\mathcal{A}$ be an arbitrary access structure for n parties, where for any $\mathcal{I} \in \mathcal{A}$ we have $|\mathcal{I}| > k_1$. Assume that:*

1. *Σ_0 is a (k, ℓ_0, ϵ_0)-BLRSS realizing $\mathcal{A}$ under adaptive partitioning, with share space such that $\log |\mathcal{S}_{0,i}| \leq \sigma_0$ (for any $i \in [n]$);*
2. *Σ_1 is a $(k_1 - 1, \ell_1, \epsilon_1)$-BLRSS realizing the k_1-out-of-n threshold access structure under adaptive partitioning;*
3. *Σ_2 is a one-time ϵ_2-non-malleable 2-out-of-2 secret sharing (*i.e. *a $(1, 1, \epsilon_2)$-NMSS).*

Then, the above defined Σ is a $(k_1 - 1, \ell, 1, 2(\epsilon_0 + \epsilon_1) + \epsilon_2)$-BLR-NMSS realizing $\mathcal{A}$ under semi-adaptive partitioning.

[8] The construction in [GK18a, Thm. 4] actually only achieves security against joint tampering within a partition $\mathcal{B}$ of the reconstruction set $\mathcal{T}$ (rather than the entire set $[n]$). Accordingly, in this case we can only tolerate joint leakage from the shares within the same partition $\mathcal{B}$.

5.2 Proof Overview

In order to prove Thm. 4, we first make some considerations on the tampering query $(\mathcal{T}, \mathcal{B}, f)$. In particular, we construct two disjoint sets $\mathcal{T}_0^*$ and $\mathcal{T}_1^*$ that are the union of subsets from the partition $\mathcal{B}$, in such a way that (i) $\mathcal{T}_0^* \cap \mathcal{T}$ contains at least k_1 elements (so that it can be used as a reconstruction set for Rec_1); and (ii) each subset $\mathcal{B}_i$ of the partition $\mathcal{B}$ intersects at most one of $\mathcal{T}_0^*, \mathcal{T}_1^*$ (so that both leakage and tampering queries can be computed on $\mathcal{T}_0^*$ and on $\mathcal{T}_1^*$ independently). Hence, we define four hybrid experiments as described below.

First Hybrid: In the first hybrid experiment, we change how the tampering query is answered. Namely, after the last leakage query, we replace all the left shares $(s_{0,\beta})_{\beta \in \mathcal{T}_1^*}$ with new shares $(s_{0,\beta}^*)_{\beta \in \mathcal{T}_1^*}$ that are valid shares of s_0 and consistent with the leakage obtained by the adversary and with the shares $(s_{0,\beta})_{\beta \in \mathcal{T}_0^*}$. Here, we note that due to the fact that we only consider *semi-adaptive* partitioning,[9] the shares $(s_{0,\beta})_{\beta \in \mathcal{T}_1^*}$ and $(s_{1,\beta})_{\beta \in \mathcal{T}_0^*}$ are independent even given the leakage. In particular, the above shares are independent before the leakage occurs, and furthermore condition (i) in Def. 7 ensures that the adversary never leaks jointly from shares in $\mathcal{T}_0^*$ and in $\mathcal{T}_1^*$. Thus, since the old and the new shares are sampled from the same distribution, this change does not affect the view of the adversary and does not modify its advantage.

Second Hybrid: In the second hybrid experiment, we change the distribution of the left shares. Namely, we discard the original ones and we replace them with left shares of some unrelated message $\hat{s}_0$, where $(\hat{s}_0, \hat{s}_1) \leftarrow\!\!\$\ \mathsf{Share}_2(0)$. In order to prove that this hybrid experiment is ϵ_0-close to the previous one, we construct an admissible reduction to leakage resilience of Σ_0, thus proving that, if some admissible adversary is able to notice the difference between the old and the new experiment with advantage more than ϵ_0, then our reduction can distinguish between a secret sharing of s_0 and a secret sharing of $\hat{s}_0$ with exactly the same advantage.

The key idea here is to forward leakage queries to the target oracle and, once the adversary outputs its tampering query, obtain all the shares in $\mathcal{T}_0^*$ from the challenger, using the augmented property ensured by Thm. 2; the reduction remains admissible because Σ_0 has security against adaptive k-partitioning and $|\mathcal{T}_0^*| \leq k$. After receiving such shares, the reduction can sample the shares $(s_{0,\beta}^*)_{\mathcal{T}_1^*}$ as in the first hybrid experiment and compute the tampering on both s_0 (using the shares in $\mathcal{T}_0^*$ and the sampled shares in $\mathcal{T}_1^*$) and s_1 (only using the shares in $\mathcal{T}_0^*$), which allows to simulate the tampering query.

Third Hybrid: In the third hybrid experiment, we change how the tampering query is answered. Similarly to the modification introduced in the first hybrid experiment, after the last leakage query, we replace all the right shares $(s_{1,\beta})_{\beta \in \mathcal{T}_0^*}$ with new shares $(s_{1,\beta}^*)_{\beta \in \mathcal{T}_0^*}$ that are valid shares of s_1 and consistent with the leakage obtained by the adversary. However, we now further require that this change does not affect the outcome of the tampering query on the left shares; in particular, if the tampering function applied to $(\hat{s}_{0,\beta}, s_{1,\beta})$ leads to $(\tilde{s}_{0,\beta}, *)$, the same tampering function applied to $(\hat{s}_{0,\beta}, s_{1,\beta}^*)$ must lead to $(\tilde{s}_{0,\beta}, *)$. This is required in order to keep consistency with the modifications introduced in the second hybrid experiment. As before, since the old and the new shares are sampled from the same distribution, this change does not modify the advantage of the adversary.

Fourth Hybrid: In the fourth hybrid experiment, we change the distribution of the right shares. Similarly to the modification introduced in the third hybrid experiment, we discard

[9] We thank Ashutosh Kumar for pointing out to us that independence given the leakage does not necessarily hold in the case of fully adaptive (rather than semi-adaptive) partitioning.

the original shares and replace them with the right shares of the previously computed unrelated message, *i.e.* $\hat{s}_1$. In order to prove that this hybrid experiment is ϵ_1-close to the previous one, we construct an admissible reduction to leakage resilience of Σ_1.

The key idea here is to simulate the tampering query with a leakage query that yields the result of the tampering on all the left shares $(\tilde{s}_{0,\beta})_{\beta\in\mathcal{T}^*}$, where $\mathcal{T}^* = \mathcal{T}_0^* \cup \mathcal{T}_1^*$. This is allowed because of the restriction on the shares of Σ_0 being at most σ_0 bits long, so that the total performed leakage is bounded by $\ell + n\sigma_0$. In particular, after sampling the fake shares $(\hat{s}_{0,1}, \ldots, \hat{s}_{0,n})$, forwarding the leakage queries to the target oracle and receiving the tampering query, the reduction samples the shares $(s^*_{0,\beta})_{\beta\in\mathcal{T}_1^*}$ as in the second hybrid experiment and hard-wires them, along with the shares $(\hat{s}_{0,1}, \ldots, \hat{s}_{0,n})$, inside a leakage function that computes $(\tilde{s}_{0,\beta}, \tilde{s}_{1,\beta})_{\beta\in\mathcal{T}^*}$ and outputs $(\tilde{s}_{0,\beta})_{\beta\in\mathcal{T}^*}$. After receiving the mauled shares, the reduction samples the shares $(s^*_{1,\beta})_{\beta\in\mathcal{T}_0^*}$ as in the third hybrid and computes the corresponding tampered shares $(\tilde{s}_{1,\beta})_{\beta\in\mathcal{T}_0^*}$. Given the mauled shares $(\tilde{s}_{0,\beta})_{\beta\in\mathcal{T}^*}$ and $(\tilde{s}_{1,\beta})_{\beta\in\mathcal{T}_0^*}$, the reduction can then simulate the tampering query correctly.

Since the above defined hybrid experiments are all statistically close, it only remains to show that no adversary can distinguish between the last hybrid experiment with bit $b = 0$ and the same experiment with $b = 1$ with an advantage more than ϵ_2, thus proving the security of our scheme. Here, we once again construct a reduction, this time to one-time ϵ_2-non-malleability, that achieves the same advantage of an adversary distinguishing between the two experiments.

The key idea is to use s_0 to sample the shares $(s^*_{0,\beta})_{\beta\in\mathcal{T}_1^*}$ and s_1 to sample the shares $(s^*_{1,\beta})_{\beta\in\mathcal{T}_0^*}$. In particular, all the missing shares needed for the computation are the one sampled from $(\hat{s}_0, \hat{s}_1)$ and, since $\mathcal{T}_0^* \cap \mathcal{T}_1^* = \emptyset$, there is no overlap and the tampering can be split between two functions f_0, f_1 that hard-wire the sampled values. These two functions take as input s_0 and s_1, respectively, and can thus compute the mauled values $\tilde{s}_0$ and $\tilde{s}_1$, which in turn allows the reduction to simulate the tampering query.

5.3 Security Analysis

Before proceeding with the analysis, we introduce some useful notation. We will define a sequence of hybrid experiments $\mathbf{H}_i(\lambda, b)$ for $i \in \mathbb{N}$ and $b \in \{0, 1\}$, starting with $\mathbf{H}_0(\lambda, b)$ which is identical to the $\mathbf{JATamper}_{\Sigma,\mathsf{A}}(\lambda, b)$ experiment. Recall that, after the leakage phase, the adversary sends a single tampering query $(\mathcal{T}, \mathcal{B}, f)$.

- Let $\tau \in \mathbb{N}$ and $\mathcal{T} = \{\beta_1, \ldots, \beta_\tau\}$, and write $\xi(i)$ for the index such that $\beta_i \in \mathcal{B}_{\xi(i)}$ (*i.e.*, the i-th share of the reconstruction is tampered by the $\xi(i)$-th tampering function).
- We define some subsets starting from $\mathcal{T}$. Call

$$\mathcal{T}_0^* = \bigcup_{\beta\in\mathcal{T}_{|k_1}} \mathcal{B}_{\xi(\beta)} \qquad \text{and} \qquad \mathcal{T}_0 = \mathcal{T}_0^* \cap \mathcal{T}.$$

  Then, use the above to define

$$\mathcal{T}_1 = \mathcal{T} \setminus \mathcal{T}_0 \qquad \text{and} \qquad \mathcal{T}_1^* = \bigcup_{\beta\in\mathcal{T}_1} \mathcal{B}_{\xi(\beta)}.$$

  and let $\mathcal{T}^* = \mathcal{T}_0^* \cup \mathcal{T}_1^*$.

Note that, with the above notation, we can write:

$$\bigcup_{\beta\in\mathcal{T}_{|k_1}} \mathcal{B}_{\xi(\beta)} = \bigcup_{\beta\in\mathcal{T}_0} \mathcal{B}_{\xi(\beta)}.$$

Moreover, $\mathcal{T}_0$ and $\mathcal{T}_1$ are defined in such a way that $|\mathcal{T}_0| \geq k_1$ and, if $\mathcal{B}_i \cap \mathcal{T} \neq \emptyset$, then either $\mathcal{B}_i \cap \mathcal{T}_0 \neq \emptyset$ or $\mathcal{B}_i \cap \mathcal{T}_1 \neq \emptyset$, but not both. In this way, we also obtain that $\mathcal{T}_0^* \cap \mathcal{T}_1^* = \emptyset$.

Finally recall that the adversary sends leakage queries $(\mathcal{B}^{(1)}, g^{(1)}), \ldots, (\mathcal{B}^{(q)}, g^{(q)})$, for $q \in \mathsf{poly}(\lambda)$, and by condition (i) in the definition of semi-adaptive admissibility (cf. Def. 7) we have that for all $\mathcal{B}^* \in \bigcup_{i\in[q]} \mathcal{B}^{(i)}$ either (1) $\exists j \in \mathcal{T} : \mathcal{B}^* \subseteq \mathcal{B}_{\xi(j)}$, or (2) $\forall j \in \mathcal{T} : \mathcal{B}^* \cap \mathcal{B}_{\xi(j)} = \emptyset$.

Hybrid 1. Let $\mathbf{H}_1(\lambda, b)$ be the same as $\mathbf{H}_0(\lambda, b)$ except for the shares of s_0 being re-sampled at the end of the leakage phase. Namely, in $\mathbf{H}_1(\lambda, b)$ we sample $(s^*_{0,\beta})_{\beta\in\mathcal{T}_1^*}$ such that $(s_{0,\beta})_{\beta\in\mathcal{T}_0^*}$, $(s^*_{0,\beta})_{\beta\in\mathcal{T}_1^*}$ are valid shares of s_0 and consistent with the leakage. Then, we answer to A's queries as follows:

- upon receiving a leakage query, use $(s_{0,1}, s_{1,1}), \ldots, (s_{0,n}, s_{1,n})$ to compute the answer;
- upon receiving the tampering query, use $(s_{0,\beta}, s_{1,\beta})_{\beta\in\mathcal{T}_0^*}, (s^*_{0,\beta}, s_{1,\beta})_{\beta\in\mathcal{T}_1^*}$ to compute the answer.

Lemma 2. *For $b \in \{0,1\}$, $\Delta(\mathbf{H}_0(\lambda, b), \mathbf{H}_1(\lambda, b)) = 0$.*

Proof. Let $(\mathbf{S}_{0,\beta})_{\beta\in\mathcal{T}_1^*}$ and $(\mathbf{S}^*_{0,\beta})_{\beta\in\mathcal{T}_1^*}$ be the random variables for the values $(s_{0,\beta})_{\beta\in\mathcal{T}_1^*}$ and $(s^*_{0,\beta})_{\beta\in\mathcal{T}_1^*}$ in experiments $\mathbf{H}_0$ and $\mathbf{H}_1$. More in details, the random variable $(\mathbf{S}^*_{0,\beta})_{\beta\in\mathcal{T}_1^*}$ comes from the distribution of the shares $(s_{0,\beta})_{\beta\in\mathcal{T}_1^*}$ conditioned on the fixed values $(s_{0,\beta})_{\beta\in\mathcal{T}_0^*}$ and the overall leakage Λ. We claim that $(\mathbf{S}^*_{0,\beta})_{\beta\in\mathcal{T}_1^*}$ and $(\mathbf{S}_{1,\beta})_{\beta\in\mathcal{T}_0^*}$ are independent conditioned on the leakage $\mathbf{\Lambda}$. This is because the random variables $(\mathbf{S}_{0,\beta})_{\beta\in\mathcal{T}_1^*}$ and $(\mathbf{S}_{1,\beta})_{\beta\in\mathcal{T}_0^*}$ are independent in isolation, and, by condition (i) in the definition of semi-adaptive admissibility, none of the leakage functions leaks simultaneously from a share in $\mathcal{T}_0^*$ and a share in $\mathcal{T}_1^*$. The latter holds as otherwise there would exist $\mathcal{B}^* \in \bigcup_{i\in[q]} \mathcal{B}^{(i)}$ such that $\mathcal{T}_1^* \cap \mathcal{B}^* \neq \emptyset$ and $\mathcal{T}_0^* \cap \mathcal{B}^* \neq \emptyset$, and therefore: (1) $\forall j \in \mathcal{T} : \mathcal{B}^* \nsubseteq \mathcal{B}_{\xi(j)}$, and (2) $\exists j \in \mathcal{T} : \mathcal{B}^* \cap \mathcal{B}_{\xi(j)} \neq \emptyset$. Finally, by Lemma 1, we can conclude that the two random variables are independent even conditioned on the leakage.

For any string $\bar{s}$, let $\mathbf{B}_0^{\bar{s}}$ and $\mathbf{B}_1^{\bar{s}}$ be, respectively, the event that $(\mathbf{S}_{0,\beta})_{\beta\in\mathcal{T}_1^*} = \bar{s}$ and $(\mathbf{S}^*_{0,\beta})_{\beta\in\mathcal{T}_1^*} = \bar{s}$. Then:

$$\begin{aligned}
&\mathbb{P}[\mathbf{H}_0(\lambda, b) = 1] - \mathbb{P}[\mathbf{H}_1(\lambda, b) = 1] \\
&= \sum_{\bar{s}} \mathbb{P}\left[\mathbf{B}_0^{\bar{s}}\right] \mathbb{P}\left[\mathbf{H}_0(\lambda, b)=1 \middle| \mathbf{B}_0^{\bar{s}}\right] - \sum_{\bar{s}} \mathbb{P}\left[\mathbf{B}_1^{\bar{s}}\right] \mathbb{P}\left[\mathbf{H}_1(\lambda, b)=1 \middle| \mathbf{B}_1^{\bar{s}}\right] \\
&= \sum_{\bar{s}} \mathbb{P}\left[\mathbf{B}_0^{\bar{s}}\right] \left(\mathbb{P}\left[\mathbf{H}_0(\lambda, b)=1 \middle| \mathbf{B}_0^{\bar{s}}\right] - \mathbb{P}\left[\mathbf{H}_1(\lambda, b)=1 \middle| \mathbf{B}_1^{\bar{s}}\right]\right) \qquad (7) \\
&= 0, \qquad (8)
\end{aligned}$$

where Eq. (7) holds because of $(\mathbf{S}^*_{0,\beta})_{\beta\in\mathcal{T}_1^*}$ being re-sampled from the same distribution of $(\mathbf{S}_{0,\beta})_{\beta\in\mathcal{T}_1^*}$ conditioned on the leakage and on the shares in $\mathcal{T}_0^*$ and Eq. (8) holds because, once fixed the value of $\bar{s}$, if both $\mathbf{B}_0^{\bar{s}}$ and $\mathbf{B}_1^{\bar{s}}$ happen, then $(\mathbf{S}_{0,\beta})_{\beta\in\mathcal{T}_1^*} = \bar{s} = (\mathbf{S}^*_{0,\beta})_{\beta\in\mathcal{T}_1^*}$ and the two hybrids are the same. □

Hybrid 2. Let $\mathbf{H}_2(\lambda, b)$ be the same as $\mathbf{H}_1(\lambda, b)$ except for the leakage being performed on fake shares of s_0. Namely, compute $(\hat{s}_0, \hat{s}_1) \leftarrow_\$ \mathsf{Share}_2(0)$, let $\hat{s}_i = (\hat{s}_{0,i}, s_{1,i})$ where $(\hat{s}_{0,1}, \ldots, \hat{s}_{0,n}) \leftarrow_\$ \mathsf{Share}_0(\hat{s}_0)$, and sample the shares $(s^*_{0,\beta})_{\beta\in\mathcal{T}_1^*}$ of $\mathbf{H}_1$ such that $(\hat{s}_{0,\beta})_{\beta\in\mathcal{T}_0^*}, (s^*_{0,\beta})_{\beta\in\mathcal{T}_1^*}$ are valid shares of s_0 and consistent with the leakage. Then:

- upon receiving a leakage query, use $(\hat{s}_{0,1}, s_{1,1}), \ldots, (\hat{s}_{0,n}, s_{1,n})$ to compute the answer;
- upon receiving the tampering query, use $(\hat{s}_{0,\beta}, s_{1,\beta})_{\beta\in\mathcal{T}_0^*}, (s^*_{0,\beta}, s_{1,\beta})_{\beta\in\mathcal{T}_1^*}$ to compute the answer.

Lemma 3. *For $b \in \{0,1\}$, $\Delta((\mathbf{H}_1(\lambda, b), \mathbf{H}_2(\lambda, b))) \leq \epsilon_0(\lambda)$.*

Proof. By reduction to leakage resilience of Σ_0. Suppose towards contradiction that there exist $b \in \{0,1\}$, messages m_0, m_1, and an adversary A able to tell apart $\mathbf{H}_1(\lambda, b)$ and $\mathbf{H}_2(\lambda, b)$ with advantage more than $\epsilon_0(\lambda)$. Let (s_0, s_1) and $(\hat{s}_0, \hat{s}_1)$ be, respectively, a secret sharing of m_b and of the all-zero string under Σ_2. Consider the following reduction trying to distinguish a secret sharing of s_0 and a secret sharing of $\hat{s}_0$ under Σ_0, where we call s_0^{target} the target secret sharing in the leakage oracle.

<u>Adversary $\hat{\mathsf{A}}^{\mathcal{O}_{\mathsf{leak}}((s_{0,i}^{\mathsf{target}})_{i\in[n]},\cdot,\cdot)}(1^\lambda)$:</u>

1. Sample $(s_{1,1}, \ldots, s_{1,n}) \leftarrow\!\!\$\ \mathsf{Share}_1(s_1)$ and run the experiment as in $\mathbf{H}_1$ with the adversary A; upon receiving each leakage function, hard-code into it the shares of s_1 and forward it to the leakage oracle.
2. Eventually, the adversary sends its tampering query. Obtain from the challenger the shares $(s_{0,\beta}^{\mathsf{target}})_{\beta\in\mathcal{T}_0^*}$ (using the augmented property from Thm. 2).
3. For all $\beta \in \mathcal{T}_0$, compute $(\tilde{s}_{0,j}, \tilde{s}_{1,j})_{j\in\mathcal{B}_{\xi(\beta)}} = f_{\xi(\beta)}((s_{0,j}^{\mathsf{target}}, s_{1,j})_{j\in\mathcal{B}_{\xi(\beta)}})$ and compute $\tilde{s}_1 = \mathsf{Rec}_1((\tilde{s}_{1,\beta})_{\beta\in\mathcal{T}_{|k_1}})$.
4. Sample $(s_{0,\beta}^*)_{\beta\in\mathcal{T}_1^*}$ as described in $\mathbf{H}_2$ and compute $\tilde{s}_0$ as follows: for all $\beta \in \mathcal{T}_1$, let $(\tilde{s}_{0,j}, \tilde{s}_{1,j})_{j\in\mathcal{B}_{\xi(\beta)}} = f_{\xi(\beta)}((s_{0,j}^*, s_{1,j})_{j\in\mathcal{B}_{\xi(\beta)}})$ and $\tilde{s}_0 = \mathsf{Rec}_0((\tilde{s}_{0,\beta})_{\beta\in\mathcal{T}})$.
5. Compute the value $\tilde{m} = \mathsf{Rec}_2(\tilde{s}_0, \tilde{s}_1)$. In case $\tilde{m} \in \{m_0, m_1\}$ return $\diamond$ to A, and else return $\tilde{m}$.
6. Output the same as A.

For the analysis, note that the reduction is perfect. In particular, the reduction perfectly simulates $\mathbf{H}_1$ when $(s_{0,i}^{\mathsf{target}})_{i\in[n]}$ is a secret sharing of s_0 and perfectly simulates $\mathbf{H}_2$ when $(s_{0,i}^{\mathsf{target}})_{i\in[n]}$ is a secret sharing of $\hat{s}_0$. Moreover, the leakage requested by A is forwarded to the leakage oracle of $\hat{\mathsf{A}}$ and perfectly simulated by it. Finally, the reduction gets in full $(s_{0,\beta}^{\mathsf{target}})_{\beta\in\mathcal{T}_0^*}$, which allows it to compute $\tilde{s}_1$, and computes $\tilde{s}_0$ by sampling the values $(s_{0,\beta}^*)_{\beta\in\mathcal{T}_1^*}$ as in $\mathbf{H}_1$.

Let us now analyze the admissibility of $\hat{\mathsf{A}}$. The only leakage performed by $\hat{\mathsf{A}}$ is the one requested by A, and augmented leakage resilience can be obtained with 1 extra bit of leakage by Thm. 2. Finally, since $|\mathcal{T}_0^*| \leq k_1(k_1 - 1) \leq k$, it follows that if A is $(k_1 - 1, \ell, 1)$-BLTA, $\hat{\mathsf{A}}$ is $(k, \ell + 1)$-BLA. □

Hybrid 3. Let $\mathbf{H}_3(\lambda, b)$ be the same as $\mathbf{H}_2(\lambda, b)$ except for the shares of s_1 being re-sampled at the end of the leakage phase. Namely, in $\mathbf{H}_3(\lambda, b)$ we sample $(s_{1,\beta}^*)_{\beta\in\mathcal{T}_0^*}$ such that (1) the shares $(s_{1,\beta})_{\beta\in\mathcal{T}_0^*}$ and $(s_{1,\beta}^*)_{\beta\in\mathcal{T}_0^*}$ agree with the same leakage and the same reconstructed secret s_1, and (2) for all $\beta \in \mathcal{T}_0$, applying the tampering function $f_{\xi(\beta)}$ to $(\hat{s}_{0,j}, s_{1,j}^*)_{j\in\mathcal{B}_{\xi(\beta)}}$ or to $(\hat{s}_{0,j}, s_{1,j})_{j\in\mathcal{B}_{\xi(\beta)}}$ leads to the same values $(\tilde{s}_{0,j})_{j\in\mathcal{B}_{\xi(\beta)}}$. Then, we answer to A's queries as follows:

- upon receiving a leakage query, use $(\hat{s}_{0,1}, s_{1,1}), \ldots, (\hat{s}_{0,n}, s_{1,n})$ to compute the answer;
- upon receiving the tampering query, use $(\hat{s}_{0,\beta}, s_{1,\beta}^*)_{\beta\in\mathcal{T}_0^*}, (s_{0,\beta}^*, s_{1,\beta})_{\beta\in\mathcal{T}_1^*}$ to compute the answer.

Lemma 4. *For $b \in \{0,1\}$, $\Delta(\mathbf{H}_2(\lambda, b), \mathbf{H}_3(\lambda, b)) = 0$.*

Proof. The proof is similar to that of Lemma 2, and thus omitted. □

Hybrid 4. Let $\mathbf{H}_4(\lambda, b)$ be the same as $\mathbf{H}_3(\lambda, b)$ except for the leakage being performed on fake shares of s_1. Namely, let $(\hat{s}_{1,i})_{i\in[n]} \leftarrow\!\!\$\ \mathsf{Share}_1(\hat{s}_1)$, where $\hat{s}_1$ comes from $\mathsf{Share}_2(0)$ as in $\mathbf{H}_2$. Then:

- upon receiving a leakage query, use $(\hat{s}_{0,1}, \hat{s}_{1,1}), \ldots, (\hat{s}_{0,n}, \hat{s}_{1,n})$ to compute the answer;
- upon receiving the tampering query, use $(\hat{s}_{0,\beta}, s^*_{1,\beta})_{\beta\in\mathcal{T}_0^*}, (s^*_{0,\beta}, \hat{s}_{1,\beta})_{\beta\in\mathcal{T}_1^*}$ to compute the answer.

Lemma 5. *For $b \in \{0,1\}$, $\Delta(\mathbf{H}_3(\lambda, b), \mathbf{H}_4(\lambda, b)) \leq \epsilon_1(\lambda)$.*

Proof. By reduction to the leakage resilience of Σ_1. Suppose towards contradiction that there exist $b \in \{0,1\}$, messages m_0, m_1, and an adversary A able to tell apart $\mathbf{H}_3(\lambda, b)$ and $\mathbf{H}_4(\lambda, b)$ with advantage more than $\epsilon_1(\lambda)$. Let (s_0, s_1) and $(\hat{s}_0, \hat{s}_1)$ be, respectively, a secret sharing of m_b and of the all-zero string under Σ_2. Consider the following reduction trying to distinguish a secret sharing of s_1 and a secret sharing of $\hat{s}_1$ under Σ_1, where we call s_1^{target} the target secret sharing in the leakage oracle.

<u>Adversary $\hat{\mathsf{A}}^{\mathcal{O}_{\mathsf{leak}}((s^{\mathsf{target}}_{1,i})_{i\in[n]},\cdot,\cdot)}(1^\lambda)$:</u>

1. Sample $(\hat{s}_{0,1}, \ldots, \hat{s}_{0,n}) \leftarrow\!\!\$\ \mathsf{Share}_0(\hat{s}_0)$ and run the experiment as in $\mathbf{H}_3$ with the adversary A; upon receiving each leakage function, hard-code into it the shares of $\hat{s}_0$ and forward it to the leakage oracle.
2. Eventually, the adversary sends its tampering query $(\mathcal{T}, \mathcal{B}, f)$.
3. Sample $(s^*_{0,\beta})_{\beta\in\mathcal{T}_1^*}$ as in $\mathbf{H}_2$. In particular, recall that we can sample these share as a function of just the shares $(s_{0,\beta})_{\beta\in\mathcal{T}_0^*}$ and the leakage. Then, set
$$s'_{0,\beta} := \begin{cases} \hat{s}_{0,\beta} & \text{if } \beta \in \mathcal{T}_0^*, \\ s^*_{0,\beta} & \text{if } \beta \in \mathcal{T}_1^*. \end{cases}$$
Note that this is well defined since $\mathcal{T}_0^* \cap \mathcal{T}_1^* = \emptyset$.
4. For all $i \in [t]$, construct the leakage function g_i that, given as input $(s^{\mathsf{target}}_{1,\beta})_{\beta\in\mathcal{B}_i}$, computes $(\tilde{s}_{0,\beta}, \tilde{s}_{1,\beta})_{\beta\in\mathcal{B}_i} = f_j((s'_{0,\beta}, s^{\mathsf{target}}_{1,\beta})_{\beta\in\mathcal{B}_i})$ and outputs $(\tilde{s}_{0,\beta})_{\beta\in\mathcal{B}_i}$. Send $(\mathcal{B}, (g_1, \ldots, g_t))$ to the leakage oracle obtaining values $(\tilde{s}_{0,\beta})_{\beta\in\mathcal{T}^*}$.
5. Sample the values $(s^*_{1,\beta})_{\beta\in\mathcal{T}_0^*}$ as in $\mathbf{H}_3$ using $(\tilde{s}_{0,\beta})_{\beta\in\mathcal{T}^*}$ and the leakage.
6. For all $j \in \mathcal{T}_0$, compute $(\tilde{s}_{0,\beta}, \tilde{s}_{1,\beta})_{\beta\in\mathcal{B}_{\xi(j)}} = f_j((s'_{0,\beta}, s^*_{1,\beta})_{\beta\in\mathcal{B}_{\xi(j)}})$; then, compute $s_0 = \mathsf{Rec}_0((\tilde{s}_{0,\beta})_{\beta\in\mathcal{T}})$ and $s_1 = \mathsf{Rec}_1((\tilde{s}_{1,\beta})_{\beta\in\mathcal{T}_{|k_1}})$ and let $\tilde{m} = \mathsf{Rec}_2(s_0, s_1)$. In case $\tilde{m} \in \{m_0, m_1\}$ return $\diamond$ to A, and else return $\tilde{m}$ to A.
7. Output the same as A.

For the analysis, note that the reduction is perfect. In particular, the reduction perfectly simulates $\mathbf{H}_3$ when $(s^{\mathsf{target}}_{1,i})_{i\in[n]}$ is a secret sharing of s_1 and perfectly simulates $\mathbf{H}_4$ when $(s^{\mathsf{target}}_{1,i})_{i\in[n]}$ is a secret sharing of $\hat{s}_1$. Moreover, the leakage requested by the adversary A is forwarded to the leakage oracle of $\hat{\mathsf{A}}$ and perfectly simulated by it. Finally, the reduction obtains all the shares $(\tilde{s}_{0,\beta})_{\beta\in\mathcal{T}^*}$, and thus it is able to both compute $\tilde{s}_0$ and sample the values $(s^*_{1,\beta})_{\beta\in\mathcal{T}_0^*}$.

Let us now analyze the admissibility of $\hat{\mathsf{A}}$. The only leakage performed by $\hat{\mathsf{A}}$ is the one requested by A in step 1 plus the one needed in order to get the values $(\tilde{s}_{0,\beta})_{\beta\in\mathcal{T}^*}$ in step 4; summing up, the overall leakage performed by $\hat{\mathsf{A}}$ is:
$$\ell + \sum_{\beta\in\mathcal{T}^*} \log|\mathcal{S}_{0,\beta}| \leq \ell + \sum_{i\in[n]} \log|\mathcal{S}_{0,i}| \leq \ell + n\sigma_0,$$
where the last inequality follows by the fact that $\log|\mathcal{S}_{0,i}| \leq \sigma_0$ for all $i \in [n]$. Therefore, we can conclude that $\hat{\mathsf{A}}$ is $(k_1 - 1, \ell + n\sigma_0)$-BLA. □

Final step. Finally, we show:

Lemma 6. $\Delta(\mathbf{H}_4(\lambda, 0), \mathbf{H}_4(\lambda, 1)) \leq \epsilon_2(\lambda)$.

Proof. By reduction to non-malleability of Σ_2. Suppose by contradiction that there exist messages m_0, m_1 and an adversary A telling apart $\mathbf{H}_4(\lambda, 0)$ and $\mathbf{H}_4(\lambda, 1)$ with advantage more than $\epsilon_2(\lambda)$. Fix values $(\hat{s}_i)_{i\in[n]} = ((\hat{s}_{0,i}, \hat{s}_{1,i})_{i\in[n]})$ and (s_0, s_1) being either a (2-out-of-2) secret sharing of m_0 or of m_1. Consider the following reduction:

Adversary $\hat{\mathsf{A}}^{\mathcal{O}_{\mathsf{nmss}}((s_0^{\mathsf{target}}, s_1^{\mathsf{target}}),\cdot)}(1^\lambda)$:

1. Run the experiment as in $\mathbf{H}_4$ with the adversary A; upon receiving each leakage function, answer using the values $(\hat{s}_i)_{i\in[n]}$.
2. Upon input the tampering query $(\mathcal{T}, \mathcal{B}, f)$, construct the following two tampering functions:
   - Function f_0, upon input s_0, samples $(s^*_{0,\beta})_{\beta\in\mathcal{T}_1^*}$ as in $\mathbf{H}_2$; notice that the reduction knows all the information needed to re-sample the shares, as in particular it samples $(s_{0,\beta})_{\beta\in[n]}$ and simulates the leakage. Then, f_0 computes $(\tilde{s}_{0,j}, \tilde{s}_{1,j})_{j\in\mathcal{B}_{\xi(\beta)}} = f_{\xi(\beta)}((\hat{s}_{0,j}, \hat{s}_{1,j})_{j\in\mathcal{B}_{\xi(\beta)}})$ for all $\beta \in \mathcal{T}_0$ and $(\tilde{s}_{0,j}, \tilde{s}_{1,j})_{j\in\mathcal{B}_{\xi(\beta)}} = f_{\xi(\beta)}((s^*_{0,j}, \hat{s}_{1,j})_{j\in\mathcal{B}_{\xi(\beta)}})$ for all $\beta \in \mathcal{T}_1$ and outputs $\tilde{s}_0 = \mathsf{Rec}_0((\tilde{s}_{0,\beta})_{\beta\in\mathcal{T}})$.
   - Function f_1, upon input s_1, samples $(s^*_{1,\beta})_{\beta\in\mathcal{T}_0^*}$ as in $\mathbf{H}_3$. Then, f_1 computes $(\tilde{s}_{0,j}, \tilde{s}_{1,j})_{j\in\mathcal{B}_{\xi(\beta)}} = f_{\xi(\beta)}((\hat{s}_{0,j}, s^*_{1,j})_{j\in\mathcal{B}_{\xi(\beta)}})$ for all $\beta \in \mathcal{T}_0$ and outputs $\tilde{s}_1 = \mathsf{Rec}_1((\tilde{s}_{1,\beta})_{\beta\in\mathcal{T}})$.
3. Send (f_0, f_1) to the tampering oracle, receiving an answer $\tilde{m}$.
4. Return $\tilde{m}$ to A and output the same as A.

For the analysis, note that the reduction is perfect. In particular, shares $(s^*_{0,\beta})_{\beta\in\mathcal{T}_1^*}$ and $(s^*_{1,\beta})_{\beta\in\mathcal{T}_0^*}$ are computed using s_0 and s_1 respectively; moreover, both $\tilde{s}_0$ and $\tilde{s}_1$ are computed as in experiment $\mathbf{H}_4$ and thus the tampering query is perfectly simulated. Finally, the leakage is computed using the fake shares $(\hat{s}_i)_{i\in[n]}$ as in $\mathbf{H}_4$ and thus, once again, perfectly simulated. The lemma follows. □

Theorem 4. Follows by the above lemmas and the triangular inequality:

$$\begin{aligned}
&\Delta(\mathbf{H}_0(\lambda, 0), \mathbf{H}_0(\lambda, 1))\\
&\leq \sum_{b\in\{0,1\}} \sum_{i\in[4]} \Delta(\mathbf{H}_{i-1}(\lambda, b), \mathbf{H}_i(\lambda, b)) + \Delta(\mathbf{H}_4(\lambda, 0), \mathbf{H}_4(\lambda, 1))\\
&\leq 2\left(\Delta(\mathbf{H}_1(\lambda, b), \mathbf{H}_2(\lambda, b)) + \Delta(\mathbf{H}_3(\lambda, b), \mathbf{H}_4(\lambda, b))\right) + \Delta(\mathbf{H}_4(\lambda, 0), \mathbf{H}_4(\lambda, 1))\\
&\leq 2(\epsilon_0 + \epsilon_1) + \epsilon_2.
\end{aligned}$$

□

5.4 Instantiation

Using a previous construction of bounded leakage-resilient secret sharing scheme against joint leakage under adaptive partitioning, we obtain the following:

Corollary 3. *For every $\ell, n, \lambda \geq 0$, every $k \in O(\sqrt{\log n})$, and every access structure $\mathcal{A}$ over n parties that can be described by a polynomial-size monotone span program for which authorized sets have size greater than k, there exists a $(k, \ell, 1, 2^{-\Omega(\lambda/\log(\lambda))})$-BLR-NMSS with message length $\Omega(\lambda/\log(\lambda))$ realizing $\mathcal{A}$ under semi-adaptive partitioning.*

Proof. By Thm. 4, we need to instantiate Σ_0, Σ_1, and Σ_2. Using [KMS19, Thm. 1] and [KMS19, Cor. 2], we can take $\epsilon_0 = \epsilon_1 = 2^{-\Omega(\lambda/\log(\lambda))}$, $k \in O(\log n)$, and thus $k_1 \in O(\sqrt{\log n})$, $\sigma_0 = \mathsf{poly}(\lambda)$ and any $\ell_0, \ell_1 > 0$. As for Σ_2, we can take the split-state non-malleable code in [Li17, Thm. 1.12], which achieves error $2^{-\Omega(\lambda/\log(\lambda))}$. □

6 Applications

6.1 Lower Bounds for Non-Malleable Secret Sharing

Combining our result from Thm. 3 with the lower bound of Nielsen and Simkin [NS20], we obtain a lower bound on the share size and randomness complexity of non-malleable secret sharing schemes. In particular, we obtain the following:

Corollary 4. *Any τ-out-of-n $(1, 1, \epsilon)$-NMSS must satisfy*

$$\sigma \geq \frac{(\log(1/\epsilon) - 1)(1 - \tau/n)}{\widehat{\tau}},$$

where $\widehat{\tau}$ is the number of shares needed to reconstruct the full *vector of shares and σ is the bit-length of each share.*

Observe that $\widehat{\tau}$ is a simplified notion of entropy. If $\tau = \widehat{\tau}$, then any authorized set can reconstruct all remaining shares, meaning that those shares have no entropy left.

6.2 Bounded-Time Non-Malleability

Here, we revisit the compiler from Ostrovsky *et al.* [OPVV18] in the setting of non-malleable secret sharing against joint tampering.

The basic idea is as follows. First, we commit to the message m using random coins r, thus obtaining a cryptographic commitment c. Then, we secret share the string $m||r$ using an auxiliary secret sharing scheme Σ, thus obtaining shares $s_1, \ldots, s_n$. The final share of the i-th party is set to be $s_i^* = (c, s_i)$. Given an authorized set $\mathcal{I}$, the reconstruction first checks that all commitments in $s_{\mathcal{I}}^*$ are equal, and then uses $s_{\mathcal{I}}$ to recover $m||r$, and verifies consistency of the commitments. If any of these checks fails, it outputs $\perp$; else, it returns m.

The original analysis by Ostrovsky *et al.* shows that if Σ is a 2-out-of-2 secret sharing that is bounded leakage-resilient, statistically one-time non-malleable, and further satisfies additional non-standard properties, then Σ^* is continuously non-malleable. In a follow up work, Brian *et al.* [BFV19] proved that the additional properties on Σ can be avoided if one assumes that Σ satisfies a stronger form of leakage resilience known as *noisy* leakage resilience, and further extended the original analysis to any value $n \geq 2$ and for arbitrary access structures.

Both the proofs in [OPVV18, BFV19] are for the setting of independent tampering. The theorem below says that the same construction works also in the case of joint p-time tampering under selective/semi-adaptive partitioning as long as Σ tolerates joint bounded leakage resilience, where there is a natural trade off between the leakage bound and the number of tampering queries. The main idea behind the proof is to reduce the security of Σ^* to that of Σ, where the bounded leakage is used to simulate multiple tampering queries. The main difference with the original proof is that we need a small leakage for each tampering query, and thus the analysis only works in case the number of tampering queries is *a priori* bounded. Moreover, in the case of semi-adaptive partitioning, we need to make sure that the leakage performed by the reduction does not violate condition (i) in the definition of semi-adaptive admissibility (cf. Def. 7); intuitively, the latter holds thanks to the fact that the tampering queries chosen by the attacker must satisfy condition (ii) in Def. 7.

Let Commit be a non-interactive commitment scheme with message space $\mathcal{M}$, randomness space $\mathcal{R}$ and commitment space $\mathcal{C}$. Let $\Sigma = (\mathsf{Share}, \mathsf{Rec})$ be an auxiliary secret sharing scheme realizing access structure $\mathcal{A}$ with message space $\mathcal{M} \times \mathcal{R}$ and share space $\mathcal{S} = \mathcal{S}_1 \times \ldots \times \mathcal{S}_n$. Define the following secret sharing scheme $\Sigma^* = (\mathsf{Share}^*, \mathsf{Rec}^*)$ with message space $\mathcal{M}$ and share space $\mathcal{S}^* = \mathcal{S}_1^* \times \ldots \times \mathcal{S}_n^*$, where, for each $i \in [n]$, we have $\mathcal{S}_i^* = \mathcal{C} \times \mathcal{S}_i$

Sharing algorithm Share^*: Upon input a value $m \in \mathcal{M}$, sample random coins $r \leftarrow\!\!\$\, \mathcal{R}$ and compute $c = \mathsf{Commit}(m; r)$ and $(s_1, \ldots, s_n) \leftarrow\!\!\$\, \mathsf{Share}(m||r)$. Return the shares $s^* = (s_1^*, \ldots, s_n^*)$ where, for each $i \in [n]$, $s_i^* = (c, s_i)$.

Reconstruction algorithm Rec^*: Upon input shares $(s_i^*)_{i\in\mathcal{I}}$, parse $s_i^* = (c_i, s_i)$ for each $i \in \mathcal{I}$. Hence, proceed as follows.

1. If $\exists i_1, i_2 \in \mathcal{I}$ for which $c_{i_1} \neq c_{i_2}$, return $\perp$; else, let the input shares be $s_i^* = (c, s_i)$.
2. Run $m||r = \mathsf{Rec}((s_i)_{i\in\mathcal{I}})$; if the outcome equals $\perp$, return $\perp$.
3. If $c = \mathsf{Commit}(m; r)$, return m; else, return $\perp$.

Figure 2: Compiler for obtaining bounded-time non-malleability against joint tampering.

Theorem 5. *Let $n \in \mathbb{N}$ and let $\mathcal{A}$ be an arbitrary access structure for n parties without singletons. Assume that:*

1. Commit *is a perfectly binding and computationally hiding non-interactive commitment;*
2. Σ *is a n-party k-joint ℓ-bounded leakage-resilient one-time non-malleable secret sharing scheme realizing access structure $\mathcal{A}$ against joint semi-adaptive (resp., selective) partitioning with information-theoretic security and with message space $\mathcal{M}$ such that $|\mathcal{M}| \in \omega(\log(\lambda))$.*

Then, the secret sharing scheme Σ^* described in Fig. 2 is a n-party k-joint p-time non-malleable secret sharing scheme realizing access structure $\mathcal{A}$ against joint semi-adaptive (resp., selective) partitioning with computational security, as long as $\ell = p \cdot (\gamma + n) + 1$, where $\gamma = \log |\mathcal{C}|$ is the size of a commitment.

Proof. The proof of privacy (w.r.t. access structure $\mathcal{A}$) was already given in [BFV19]. In what follows, we focus on showing joint non-malleability under semi-adaptive partitioning. The proof for the case of selective partitioning is almost the same, the only difference being that the partition $\mathcal{B}$ is fixed at the beginning of the experiment instead of given by the adversary.

Let $\mathbf{JATamper}^{m_0,m_1}_{\Sigma^*,\mathsf{A}}(\lambda, b)$, for $m_0, m_1 \in \mathcal{M}, b \in \{0, 1\}$, be the original experiment defining p-time non-malleability of Σ^*. Consider a modified experiment $\mathbf{H}^{m_0,m_1}_{\Sigma^*,\mathsf{A}}(\lambda, b)$ where we replace $(s_1, \ldots, s_n)$ with a secret sharing of a random and independent value $\hat{m}||\hat{r} \leftarrow\!\!\$\, \mathcal{M} \times \mathcal{R}$. Both the original and the hybrid experiments are depicted in Fig. 3. We first prove that the above experiments are computationally close by induction over the number of tampering queries $p^* \leq p$ asked by the adversary A; towards this, let us denote by $\mathbf{JATamper}^{m_0,m_1}_{\Sigma^*,\mathsf{A}}(\lambda, p^*, b)$ (resp. $\mathbf{H}^{m_0,m_1}_{\Sigma^*,\mathsf{A}}(\lambda, p^*, b)$) the original (resp. hybrid) experiment where the adversary A is limited to ask exactly p queries to the oracle $\mathcal{O}_{\mathsf{nmss}}$. The lemma below constitutes the basis of the induction.

Lemma 7. *For all pairs of distinct messages $m_0, m_1 \in \mathcal{M}$ and for all $b \in \{0, 1\}$,*

$$\{\mathbf{JATamper}^{m_0,m_1}_{\Sigma^*,\mathsf{A}}(\lambda, 1, b)\}_{\lambda\in\mathbb{N}} \overset{\mathsf{s}}{\approx} \{\mathbf{H}^{m_0,m_1}_{\Sigma^*,\mathsf{A}}(\lambda, 1, b)\}_{\lambda\in\mathbb{N}}.$$

```
$\mathbf{JATamper}^{m_0,m_1}_{\Sigma,\mathsf{A}}(\lambda, b)$  $\mathbf{H}^{m_0,m_1}_{\Sigma,\mathsf{A}}(\lambda, b)$ :
$r \leftarrow\!\!\$\ \mathcal{R}$,  $\hat{m}||\hat{r} \leftarrow\!\!\$\ \mathcal{M} \times \mathcal{R}$
$c := \mathsf{Commit}(m_b; r)$
$(s_1, \ldots, s_n) \leftarrow\!\!\$\ \mathsf{Share}(m_b||r)$
$(s_1, \ldots, s_n) \leftarrow\!\!\$\ \mathsf{Share}(\hat{m}||\hat{r})$
$s^* := ((c, s_1), \ldots, (c, s_n))$
$\mathtt{stop} \leftarrow \mathsf{false}$
Return $\mathsf{A}^{\mathcal{O}_{\mathsf{nmss}}(s^*,\cdot,\cdot,\cdot),\mathcal{O}_{\mathsf{leak}}(s^*,\cdot,\cdot)}(1^\lambda, \alpha)$

Oracle $\mathcal{O}_{\mathsf{leak}}(s^*, \mathcal{B}, (g_1, \ldots, g_t))$:
Return $g_1(s^*_{\mathcal{B}_1}), \ldots, g_t(s^*_{\mathcal{B}_t})$

Oracle $\mathcal{O}_{\mathsf{nmss}}(s^*, \mathcal{T}, \mathcal{B}, (f_1, \ldots, f_t))$:
If $\mathtt{stop} = \mathsf{true}$, return $\perp$
$\forall i \in [t] : \tilde{s}^*_{\mathcal{B}_i} := f_i(s^*_{\mathcal{B}_i})$
$\tilde{s}^* = ((\tilde{c}_1, \tilde{s}_1), \ldots, (\tilde{c}_n, \tilde{s}_n))$
If $\exists i_1, i_2 \in \mathcal{T} : \tilde{c}_{i_1} \neq \tilde{c}_{i_2}$
    $\mathtt{stop} \leftarrow \mathsf{true}$ and return $\perp$
Else, let $\tilde{c} := \tilde{c}_i$
$\tilde{m}||\tilde{r} = \mathsf{Rec}(\tilde{s}_{\mathcal{T}})$
If $\tilde{m}||\tilde{r} = \perp$
    $\mathtt{stop} \leftarrow \mathsf{true}$ and return $\perp$
If $c \neq \mathsf{Commit}(\tilde{m}; \tilde{r})$
    $\mathtt{stop} \leftarrow \mathsf{true}$ and return $\perp$
If $\tilde{m} \in \{m_0, m_1\}$
    Return $\diamond$
If $\tilde{m} = \hat{m}$
    If $\tilde{c} = c$ return $\diamond$
    Else return $\perp$
Return $\tilde{m}$
```

Figure 3: Experiment $\mathbf{JATamper}^{m_0,m_1}_{\Sigma,\mathsf{A}}(\lambda, b)$ applied to our scheme. The instructions boxed in red are the modifications introduced by the hybrid experiment.

Proof. The proof is down to the statistical leakage-resilient one-time non-malleability of the underlying scheme Σ. Fix $b = 0$ (the proof for the other case being identical). Assume that there exist two distinct messages m_0, m_1 and an unbounded adversary A which can distinguish between $\mathbf{JATamper}^{m_0,m_1}_{\Sigma^*,\mathsf{A}}(\lambda, 1, 0)$ and $\mathbf{H}^{m_0,m_1}_{\Sigma^*,\mathsf{A}}(\lambda, 1, 0)$ with non-negligible advantage. By an averaging argument, this means that there must exist values $r \in \mathcal{R}$ and $\hat{m}||\hat{r} \in \mathcal{M} \times \mathcal{R}$ such that A distinguishes the two experiments when we fix these particular values of r and $\hat{m}||\hat{r}$. Let $\hat{m}_0 = m_0||r$, $\hat{m}_1 = \hat{m}||\hat{r}$ and $c = \mathsf{Commit}(m_0; r)$ and let $s = (s_1, \ldots, s_n)$ be the target secret sharing of either $\hat{m}_0$ or $\hat{m}_1$. Without loss of generality, we can assume that A is deterministic.[10] Consider the following adversary $\hat{\mathsf{A}}$ attacking Σ.

1. Run $\mathsf{A}(1^\lambda)$.
2. Upon input the only tampering query $(\mathcal{T}, \mathcal{B}, (f_1, \ldots, f_t))$ from A, proceed as follows.
   (a) Choose any $i \in [t]$ such that $\mathcal{B}_i \cap \mathcal{T} \neq \emptyset$ and define the leakage function $\hat{g}_i$ that hard-wires (a description of) f_i and c and returns the commitment $\tilde{c}$ such that $f_i((c, s_j)_{j \in \mathcal{B}_i}) = (\tilde{c}_j, \tilde{s}_j)_{j \in \mathcal{B}_i}$ and $\tilde{c} = \tilde{c}_j$ for all $j \in \mathcal{B}_i \cap \mathcal{T}$; if such commitment does not exist (i.e. there are at least two different $\tilde{c}_{j_1}$ and $\tilde{c}_{j_2}$, with $j_1, j_2 \in \mathcal{B}_i \cap \mathcal{T}$), let $\tilde{c} = \perp$.
   (b) Forward $(\varepsilon, \ldots, \varepsilon, \hat{g}_i, \varepsilon, \ldots, \varepsilon)$ to the target leakage oracle, obtaining the commitment $\tilde{c}$.
   (c) For each $i \in [t]$, define the leakage function $\hat{h}_i$ that hard-wires (a description of) f_i and $c, \tilde{c}$ and returns a bit b_i such that $b_i = 1$ if and only if $\tilde{c}_j = \tilde{c}$ for all $j \in \mathcal{B}_i \cap \mathcal{T}$, where $\tilde{c}_j$ comes from $f_i((c, s_j)_{j \in \mathcal{B}_i}) = (\tilde{c}_j, \tilde{s}_j)_{j \in \mathcal{B}_i}$.
   (d) Forward $(\hat{h}_1, \ldots, \hat{h}_t)$ to the target leakage oracle, obtaining bits $(b_1, \ldots, b_t)$.

[10] It is always possible to fix the random coins of A in order to maximize its distinguishing advantage.

(e) If there exist $i \in [t]$ such that $b_i = 0$ and $\mathcal{B}_i \cap \mathcal{T} \neq \emptyset$, return $\perp$ to A; otherwise, continue as follows.

(f) Define the tampering functions $\hat{f}_i$ that hard-wires c and (a description of) f_i and, upon input $(s_j)_{j \in \mathcal{B}_i}$, returns the values $(\tilde{s}_j)_{j \in \mathcal{B}_i}$ specified by $f_i((c, s_j)_{j \in \mathcal{B}_i}) = (\tilde{c}_j, \tilde{s}_j)_{j \in \mathcal{B}_i}$.

(g) Forward $(\mathcal{T}, \mathcal{B}, (\hat{f}_1, \ldots, \hat{f}_t))$ to the tampering oracle, obtaining $\tilde{m}||\tilde{r} \in \mathcal{M} \times \mathcal{R} \cup \{\perp, \diamond\}$. Hence:

  - If $\tilde{m}||\tilde{r} = \perp$ or is not a valid opening of $\tilde{c}$, return $\perp$ to A.
  - If $\tilde{m} \in \{m_0, m_1\}$, return $\diamond$ to A. Else, if $\tilde{m} = \hat{m}$ return $\diamond$ to A in case $\tilde{c} = c$ and $\perp$ otherwise.
  - Else, return $\tilde{m}$ to A.

3. Output the same guess as that of A.

For the analysis, we next prove that the simulation performed by the above reduction is perfect with overwhelming probability. First, since A is deterministic and no random sampling is involved, $\hat{\mathsf{A}}$ is deterministic. Second, depending on the target $(s_1, \ldots, s_n)$ being either a secret sharing of $\hat{m}_0$ or of $\hat{m}_1$, for every $i \in [t]$, being t the number of subsets of the partition in the current query, the input to the tampering function f_i (resp. leakage function g_i) is identically distributed to the shares in $\mathcal{B}_i$ of the target secret sharing in either experiment $\mathbf{JATamper}^{m_0,m_1}_{\Sigma^*,\mathsf{A}}(\lambda, 0, 1)$ or $\mathbf{H}^{m_0,m_1}_{\Sigma^*,\mathsf{A}}(\lambda, 0, 1)$, with our fixed choice of $r, \hat{m}, \hat{r}$. Third, the answer to A's tampering query is simulated correctly with all but a negligible probability. Indeed:

- If $\mathsf{Rec}(\tilde{s}_{\mathcal{T}})$ yields $\perp$, both the real and the hybrid experiment would return $\perp$, which is perfectly emulated by the reduction.

- If $\mathsf{Rec}(\tilde{s}_{\mathcal{T}})$ yields $\diamond$, it means that the inner secret sharing reconstructs to either $\hat{m}_0 = m_0||r$ or to $\hat{m}_1 = \hat{m}||\hat{r}$. Without loss of generality, assume further that the commitments in the tampered shares are all equal to a single value $\tilde{c}$.[11] There are 4 possible cases: either both experiments output the same $\hat{m}_0$ or $\hat{m}_1$ or one experiment outputs $\hat{m}_0$ while the other one outputs $\hat{m}_1$. However, since the view in the real experiment is independent of the value $\hat{m}$, except with negligible probability $2^{-\omega(\log(\lambda))}$, we can condition on the event that the real experiment does not output this value. Thus, there are only two cases to consider:

  1. Both the real and the hybrid experiment return $\hat{m}_0 = m_0||r$.
  2. The real experiment returns $\hat{m}_0 = m_0||r$ whereas the hybrid returns $\hat{m}_1 = \hat{m}||\hat{r}$.

  In both cases, the output of the two experiments is equal to $\diamond$ in case $\tilde{c} = c$ and $\perp$ otherwise. This is exactly what the reduction does. Hence, the simulation is perfect except with negligible probability.

- If $\mathsf{Rec}(\tilde{s}_{\mathcal{T}})$ yields some value $\tilde{m}||\tilde{r} \notin \{\diamond, \perp\}$, it means in particular that $\tilde{m}||\tilde{r} \notin \{\hat{m}_0, \hat{m}_1\}$. In such a case both experiments return $\perp$ in case the modified commitment $\tilde{c}$ does not match the opening $(\tilde{m}, \tilde{r})$. Otherwise, it means that the modified shares produced by A lead to a valid message $\tilde{m} \in \mathcal{M}$. Thus, the output of both experiment would be either $\diamond$ (in case $\tilde{m}$ is equal to one of the two messages m_0, m_1) or $\tilde{m}$.

Finally, note that the partition used for the leakage queries is the same partition of the tampering query and that the overall leakage performed by $\hat{\mathsf{A}}$ amounts to a commitment and $t \leq n$ bits

[11] In fact, if this is not the case, both experiments would have returned $\perp$, which is once again perfectly emulated by the reduction.

and thus it is ℓ-admissible whenever $\gamma + n \leq \ell$ and also satisfies the restriction on the partition. Therefore, we can conclude that the distinguishing advantage of $\hat{\mathsf{A}}$ is the same as that of A with overwhelming probability, which concludes the proof of the lemma. □

The lemma below constitutes the inductive step.

Lemma 8. *Fix any $p^* \leq p-1$ and assume that for all $b \in \{0,1\}$ and all pairs of distinct messages $m_0, m_1 \in \mathcal{M}$,*

$$\{\mathbf{JATamper}^{m_0,m_1}_{\Sigma^*,\mathsf{A}}(\lambda, p^*, b)\}_{\lambda\in\mathbb{N}} \overset{\mathsf{s}}{\approx} \{\mathbf{H}^{m_0,m_1}_{\Sigma^*,\mathsf{A}}(\lambda, p^*, b)\}_{\lambda\in\mathbb{N}}.$$

Then, $\{\mathbf{JATamper}^{m_0,m_1}_{\Sigma^*,\mathsf{A}}(\lambda, p^*+1, b)\}_{\lambda\in\mathbb{N}} \overset{\mathsf{s}}{\approx} \{\mathbf{H}^{m_0,m_1}_{\Sigma^*,\mathsf{A}}(\lambda, p^*+1, b)\}_{\lambda\in\mathbb{N}}$ for all $b \in \{0,1\}$ and all pairs of distinct messages $m_0, m_1 \in \mathcal{M}$.

Proof. The proof is down to the statistical leakage-resilient one-time non-malleability of Σ. Fix $b = 0$ (the proof for the other case being identical). Assume that there exist two distinct messages $m_0, m_1 \in \mathcal{M}$ an unbounded adversary A which can distinguish between the experiments $\mathbf{JATamper}^{m_0,m_1}_{\Sigma^*,\mathsf{A}}(\lambda, p^*+1, 0)$ and $\mathbf{H}^{m_0,m_1}_{\Sigma^*,\mathsf{A}}(\lambda, p^*+1, 0)$. By an averaging argument, this means that there must exist values $r \in \mathcal{R}$ and $\hat{m}||\hat{r} \in \mathcal{M} \times \mathcal{R}$ such that A distinguishes the two experiments when we fix these particular values of r and $\hat{m}||\hat{r}$. Let $\hat{m}_0 = m_0||r$, $\hat{m}_1 = \hat{m}||\hat{r}$ and $c = \mathsf{Commit}(m_0; r)$ and let $s = (s_1, \ldots, s_n)$ be the target secret sharing of either $\hat{m}_0$ or $\hat{m}_1$. Without loss of generality, we can assume that A is deterministic. Consider the following adversary $\hat{\mathsf{A}}$ attacking Σ.

1. Run $\mathsf{A}(1^\lambda)$.
2. For each $q \in [p^*]$, upon input the q-th tampering query $(\mathcal{T}^{(q)}, \mathcal{B}^{(q)}, (f_1^{(q)}, \ldots, f_{t^{(q)}}^{(q)}))$, proceed as follows.
   (a) Choose any $i \in [t^{(q)}]$ such that $\mathcal{B}_i^{(q)} \cap \mathcal{T}^{(q)} \neq \emptyset$ and define the leakage function $\hat{g}_i$ that hard-wires (a description of) f_i and c and returns the commitment $\tilde{c}$ such that $f_i^{(q)}((c, s_j)_{j\in\mathcal{B}_i^{(q)}}) = (\tilde{c}_j, \tilde{s}_j)_{j\in\mathcal{B}_i^{(q)}}$ and $\tilde{c} = \tilde{c}_j$ for all $j \in \mathcal{B}_i \cap \mathcal{T}$; if such commitment does not exist (i.e. there are at least two different $\tilde{c}_{j_1}$ and $\tilde{c}_{j_2}$, with $j_1, j_2 \in \mathcal{B}_i^{(q)} \cap \mathcal{T}^{(q)}$), let $\tilde{c} = \perp$.

   (b) Forward $(\mathcal{B}^{(q)}, (\varepsilon, \ldots, \varepsilon, \hat{g}_i, \varepsilon, \ldots, \varepsilon))$ to the target leakage oracle, obtaining the commitment $\tilde{c}^{(q)}$.

   (c) For each $i \in [t]$ such that $\mathcal{B}_i^{(q)} \cap \mathcal{T}^{(q)} \neq \emptyset$, define the leakage function $\hat{h}_i$ that hard-wires (a description of) f_i and $c, \tilde{c}^{(q)}$ and returns a bit b_i such that $b_i = 1$ if and only if $\tilde{c}_j = \tilde{c}^{(q)}$ for all $j \in \mathcal{B}_i^{(q)} \cap \mathcal{T}^{(q)}$, where $\tilde{c}_j$ comes from $f_i^{(q)}((c, s_j)_{j\in\mathcal{B}_i^{(q)}}) = (\tilde{c}_j, \tilde{s}_j)_{j\in\mathcal{B}_i^{(q)}}$. For all other $i \in [t]$, simply let $\hat{h}_i = \varepsilon$ (*i.e.*, no leakage from those subset without elements in $\mathcal{T}^{(q)}$).

   (d) Forward $(\mathcal{B}^{(q)}, (\hat{h}_1, \ldots, \hat{h}_t))$ to the target leakage oracle, obtaining bits $(b_1^{(q)}, \ldots, b_t^{(q)})$.

   (e) If $\tilde{c}^{(q)} = \perp$ or there exist $i \in [t]$ such that $b_i^{(q)} = 0$ and $\mathcal{B}_i^{(q)} \cap \mathcal{T}^{(q)} \neq \emptyset$, return $\perp$ to A and self-destruct; otherwise, proceed as follows:
      - Find by brute force the opening $\tilde{m}^{(q)}$ of $\tilde{c}^{(q)}$ (i.e. $\tilde{c}^{(q)} = \mathsf{Commit}(\tilde{m}^{(q)}; \tilde{r}^{(q)})$ for some $\tilde{r}^{(q)} \in \mathcal{R}$); if no such value is found, set $\tilde{m}^{(q)} = \perp$ and self-destruct.
      - If $\tilde{m}^{(q)} \in \{m_0, m_1\}$, re-define $\tilde{m}^{(q)} = \diamond$. Else, if $\tilde{m}^{(q)} = \hat{m}$, re-define $\tilde{m}^{(q)} = \diamond$ in case $\tilde{c}^{(q)} = c$ and $\tilde{m}^{(q)} = \perp$ otherwise; in the latter case, self-destruct.

  - Return $\tilde{m}^{(q)}$ to A.

3. Upon input the last tampering query $(\mathcal{T}^{(p^*+1)}, \mathcal{B}^{(p^*+1)}, (f_1^{(p^*+1)}, \ldots, f_{t^{(p^*+1)}}^{(p^*+1)}))$, proceed as follows.
   (a) Check that the simulation up to the first p^* queries did not cause any inconsistency due to the fact that the outcome of the q-th tampering query should have been $\perp$ because $(\tilde{s}_i)_{i \in \mathcal{T}^{(q)}}$ was not a valid secret sharing.
      i. Without loss of generality, assume that $1 \in \mathcal{T}^{(p^*+1)}$ (it is always possible to permute the indices) and that the output of A is equal to 0 whenever it believes that the target secret sharing is distributed as in the real experiment.
      ii. Define the special set $\hat{\mathcal{S}} \subseteq \mathcal{S}_1 \times \ldots, \times \mathcal{S}_n$ such that $\hat{\mathcal{S}}$ contains all the possible secret sharings of m_0 and m_1 that are compatible with the answer to the tampering queries being $\tilde{m}^{(1)}, \ldots, \tilde{m}^{(p)}$.
      iii. Define the following special leakage function $\hat{h}_{\mathsf{check}} : \mathcal{S}_1 \to \{0, 1\}$.
         - The function hard-wires a description of A, the values (c, m_0, m_1), a description of the final tampering query $(\mathcal{T}^{(p^*+1)}, \mathcal{B}^{(p^*+1)}, (f_1^{(p^*+1)}, \ldots, f_{t^{(p^*+1)}}^{(p^*+1)}))$, the answer to the previous tampering queries $(\tilde{m}^{(1)}, \ldots, \tilde{m}^{(p^*)})$ and the set $\hat{\mathcal{S}}$.
         - Let $\hat{s}^* = ((c, s_1), (c, \hat{s}_2), \ldots, (c, \hat{s}_n))$ be the target secret sharing for each possible set of compatible shares $(s_1, \hat{s}_2, \ldots, \hat{s}_n) \in \hat{\mathcal{S}}$.
         - The output of the function is a bit $\tilde{b}$ such that $\tilde{b} = 1$ if and only if $\mathsf{A}(\tilde{m}^{(1)}, \ldots, \tilde{m}^{(p^*)}, \tilde{m}^*) = 0$ more often when $\hat{s}^*$ is a valid secret sharing of message m_0, where $\tilde{m}^*$ is the output of the $\mathcal{O}_{\mathsf{nmss}}$ oracle in the hybrid experiment upon input $(\mathcal{T}^{(p^*+1)}, \mathcal{B}^{(p^*+1)}, (f_1^{(p^*+1)}, \ldots, f_{t^{(p^*+1)}}^{(p^*+1)}))$ with target secret sharing $\hat{s}^*$.
      iv. Forward $((\{1\}, \ldots, \{n\}), (\hat{h}_{\mathsf{check}}, \varepsilon, \ldots, \varepsilon))$ to the target leakage oracle,[12] obtaining a bit $\tilde{b}$.

   (b) Define the same functions $g_i^{(p^*+1)}$ and $\hat{h}_i^{(p^*+1)}$ considered in step 2a and 2c and forward them to the target leakage oracle, obtaining either the mauled commitment $\tilde{c}^{(p^*+1)}$ or $\perp$; in the latter case, return $\perp$ to A and self-destruct.
   (c) Define the tampering function $\hat{f}_i$ that hard-wires c and (a description of) $f_i^{(p^*+1)}$ and, upon input $(s_j)_{j \in \mathcal{B}_i}$, returns the values $(\tilde{s}_j^{(p^*+1)})_{j \in \mathcal{B}_i}$ specified by $f_i^{(p^*+1)}((c, s_j)_{j \in \mathcal{B}_i}) = (\tilde{c}_i^{(p^*+1)}, \tilde{s}_j^{(p^*+1)})_{j \in \mathcal{B}_i}$.
   (d) Forward $(\mathcal{T}^{(p^*+1)}, \mathcal{B}^{(p^*+1)}, (\hat{f}_1, \ldots, \hat{f}_{t^{(p^*+1)}}))$ to the target tampering oracle, obtaining $\tilde{m}^{(p^*+1)} || \tilde{r}^{(p^*+1)} \in \mathcal{M} \times \mathcal{R} \cup \{\diamond, \perp\}$. Hence:
      - If $\tilde{m}^{(p^*+1)} || \tilde{r}^{(p^*+1)} = \perp$ or is not a valid opening of $\tilde{c}^{(p^*+1)}$, return $\perp$ to A.
      - If $\tilde{m}^{(p^*+1)} || \tilde{r}^{(p^*+1)} = \diamond$, return $\diamond$ to A in case $\tilde{c}^{(p^*+1)} = c$ and $\perp$ otherwise.
      - If $\tilde{m}^{(p^*+1)} \in \{m_0, m_1\}$, return $\diamond$ to A. Else, if $\tilde{m}^{(p^*+1)} = \hat{m}$, return $\diamond$ to A in case $\tilde{c}^{(p^*+1)} = c$ and $\perp$ otherwise.
      - Else, return $\tilde{m}^{(p^*+1)}$ to A.

4. Upon receiving a bit b' from A, in case $\tilde{b} = 1$ output b' and else return 0.

Attacker $\hat{\mathsf{A}}$ runs in exponential time. Since A is deterministic and no random sampling is involved, $\hat{\mathsf{A}}$ is deterministic. We now show that its distinguishing advantage is negligibly close to that of A. Indeed:

$$\left| \mathbb{P}\left[\mathbf{JATamper}_{\Sigma^*, \hat{\mathsf{A}}}^{m_0, m_1}(\lambda, 1, 0) = 1 \right] - \mathbb{P}\left[\mathbf{JATamper}_{\Sigma^*, \hat{\mathsf{A}}}^{m_0, m_1}(\lambda, 1, 1) = 1 \right] \right|$$

[12] Note that this query is an independent-leakage query and can be performed using any other partition $\mathcal{B}$.

$$
\begin{aligned}
&= \left| \mathbb{P}\left[\mathbf{JATamper}^{m_0,m_1}_{\Sigma^*,\hat{\mathsf{A}}}(\lambda,1,0) = 1 \wedge \tilde{b} = 1 \right] \right. && (9)\\
&\qquad \left. -\mathbb{P}\left[\mathbf{JATamper}^{m_0,m_1}_{\Sigma^*,\hat{\mathsf{A}}}(\lambda,1,1) = 1 \wedge \tilde{b} = 1 \right] \right| \\
&\geq \frac{1}{\mathsf{poly}(\lambda)} \left| \mathbb{P}\left[\mathbf{JATamper}^{m_0,m_1}_{\Sigma^*,\hat{\mathsf{A}}}(\lambda,1,0) = 1 \middle| \tilde{b} = 1 \right] \right. && (10)\\
&\qquad \left. -\mathbb{P}\left[\mathbf{JATamper}^{m_0,m_1}_{\Sigma^*,\hat{\mathsf{A}}}(\lambda,1,1) = 1 \middle| \tilde{b} = 1 \right] \right| \\
&\geq \frac{1}{\mathsf{poly}(\lambda)} \left(\frac{1}{\mathsf{poly}(\lambda)} - \mathsf{negl}(\lambda) \right), && (11)
\end{aligned}
$$

where Eq. (9) follows because when $\tilde{b} = 0$, the reduction $\hat{\mathsf{A}}$ returns 0 unconditionally and this cancels its distinguishing advantage; Eq. (10) holds as the induction hypothesis implies that $\tilde{b} = 1$ with non-negligible probability, otherwise A generates an invalid secret sharing $(\tilde{s}^*_1, \ldots, \tilde{s}^*_n)$ within the first p tampering queries with overwhelming probability, which in turn means that A can distinguish using less than $p+1$ outputs from the decoding. Finally, Eq. (11) holds because an analysis identical to that of Lemma 7 shows that the view of A is perfectly simulated (except with negligible probability) conditioned on $\tilde{b} = 1$, and thus in this case $\hat{\mathsf{A}}$ retains essentially the same advantage as that of A.

In order to conclude the proof, it remains to show that $\hat{\mathsf{A}}$ is ℓ-admissible, for ℓ as in the statement of the theorem, and also that $\hat{\mathsf{A}}$ satisfies condition (i) in the definition of semi-adaptive admissibility. Note that the adversary $\hat{\mathsf{A}}$ makes leakage queries in steps 2b, 2d, 3(a)iv and 3b. In particular, $\hat{\mathsf{A}}$ performs in step 3b the exact same leakage performed in each tampering query in steps 2b and 2d, that is, the mauled commitment and up to n bits. Moreover, the leakage performed in step 3(a)iv amounts to exactly 1 bit. Therefore, the total leakage performed by $\hat{\mathsf{A}}$ amounts to at most

$$(p^*+1)(\gamma+n)+1 \leq p \cdot (\gamma+n)+1 = \ell.$$

Finally, $\hat{\mathsf{A}}$ converts all the tampering queries from A into leakage queries using the same partition, except that $\hat{\mathsf{A}}$ performs no leakage from the subsets that do not intersect the reconstruction subset (*i.e.*, $\mathcal{B}^{(q)}_i \cap \mathcal{T} = \emptyset$). Therefore, if A satisfies condition (ii) in the definition of semi-adaptive admissibility, we have that, for the last tampering query and any other tampering query q, for all $i \in \mathcal{T}^{(q)}$, either there exists $j \in \mathcal{T}^{(p^*+1)}$ such that $\mathcal{B}_{\xi(i)} \subseteq \mathcal{B}_{\xi(j)}$ or, for all $j \in \mathcal{T}^{(p^*+1)}$, we have that $\mathcal{B}^{(q)}_{\xi(i)} \cap \mathcal{B}^{(p^*+1)}_{\xi(j)} = \emptyset$. However, since no leakage is performed from $\mathcal{B}^{(q)}_i$ whenever $\mathcal{B}^{(q)}_i \cap \mathcal{T}^{(q)} = \emptyset$, this is equivalent to saying that, for all $i \in [t^{(q)}]$, either there exists $j \in \mathcal{T}^{(p^*+1)}$ such that $\mathcal{B}_i \subseteq \mathcal{B}_{\xi(j)}$, or for all $j \in \mathcal{T}^{(p^*+1)}$ we have that $\mathcal{B}^{(q)}_i \cap \mathcal{B}^{(p^*+1)}_{\xi(j)} = \emptyset$. But this is exactly condition (i) in the definition of semi-adaptive admissibility. The lemma follows. □

Combining Lemma 7 and Lemma 8, we get that, for all $b \in \{0,1\}$ and all pairs of distinct messages $m_0, m_1 \in \mathcal{M}$,

$$\{\mathbf{JATamper}^{m_0,m_1}_{\Sigma^*,\mathsf{A}}(\lambda,b)\}_{\lambda\in\mathbb{N}} \overset{\mathsf{s}}{\approx} \{\mathbf{H}^{m_0,m_1}_{\Sigma^*,\mathsf{A}}(\lambda,b)\}_{\lambda\in\mathbb{N}}.$$

The lemma below concludes the proof of the theorem.

Lemma 9. *For all pairs of distinct messages $m_0, m_1 \in \mathcal{M}$,*

$$\{\mathbf{H}^{m_0,m_1}_{\Sigma^*,\mathsf{A}}(\lambda,0)\}_{\lambda\in\mathbb{N}} \overset{\mathsf{c}}{\approx} \{\mathbf{H}^{m_0,m_1}_{\Sigma^*,\mathsf{A}}(\lambda,1)\}_{\lambda\in\mathbb{N}}.$$

Proof. The proof is down to the computational hiding property of the non-interactive commitment scheme. Assume that there exist two distinct messages $m_0, m_1 \in \mathcal{M}$ and a PPT adversary

A telling apart $\mathbf{H}^{m_0,m_1}_{\Sigma^*,\mathsf{A}}(\lambda, 0)$ and $\mathbf{H}^{m_0,m_1}_{\Sigma^*,\mathsf{A}}(\lambda, 1)$ with non-negligible advantage. Fix $r \in \mathcal{R}$ and let $\hat{c} = \mathsf{Commit}(m_b; r)$ be the target commitment, where $b \in \{0, 1\}$. Consider the following adversary $\hat{\mathsf{A}}$ attacking the hiding property of Commit.

1. Sample $(s_1, \ldots, s_n) \leftarrow\!\!\$\ \mathsf{Share}(\hat{m}||\hat{r})$, where $\hat{m}||\hat{r} \leftarrow\!\!\$\ \mathcal{M} \times \mathcal{R}$. Then, sample random coins $r_{\mathsf{A}} \leftarrow\!\!\$\ \mathcal{R}_{\mathsf{A}}$ and run $\mathsf{A}(r_{\mathsf{A}})$.
2. Upon input the q-th tampering query $(\mathcal{T}^{(q)}, \mathcal{B}^{(q)}, (f_1^{(q)}, \ldots, f_t^{(q)}))$ from A, proceed as follows:
   - For each $i \in [t]$, compute
   $$(\tilde{s}^*_j)_{j \in \mathcal{B}_i} = f_i^{(q)}((\hat{c}, s_j)_{j \in \mathcal{B}_i}) = ((\tilde{c}_j, \tilde{s}_j)_{j \in \mathcal{B}_i})$$
   and let $\tilde{m}^{(q)}||\tilde{r}^{(q)} = \mathsf{Rec}(\tilde{s}_{\mathcal{T}^{(q)}})$, where $\tilde{s} = (\tilde{s}_1, \ldots, \tilde{s}_n)$.
   - If $\exists i_i, i_2 \in \mathcal{T}^{(q)}$ s.t. $\tilde{c}_{i_1} \neq \tilde{c}_{i_2}$, return $\perp$ to A and self-destruct.
   - If $\tilde{m}^{(q)} = \perp$ or $\tilde{c}_1 \neq \mathsf{Commit}(\tilde{m}^{(q)}; \tilde{r})$, return $\perp$ to A and self-destruct.
   - If $\tilde{m}^{(q)} \in \{m_0, m_1\}$, return $\diamond$ to A.
   - If $\tilde{m}^{(q)} = \hat{m}$, return $\diamond$ to A in case $\tilde{c}_1 = \hat{c}_1$ and $\perp$ otherwise; in the latter case, self-destruct.
   - Else, return $\tilde{m}^{(q)}$ to A.
3. Return the same guess as A.

For the analysis, note that the simulation done by $\hat{\mathsf{A}}$ is perfect. In particular, depending on the value $\hat{c}$ being a commitment to either m_0 or m_1, the view of A is identical to the one in either experiment $\mathbf{H}^{m_0,m_1}_{\Sigma^*,\mathsf{A}}(\lambda, 0)$ or $\mathbf{H}^{m_0,m_1}_{\Sigma^*,\mathsf{A}}(\lambda, 1)$, therefore $\hat{\mathsf{A}}$ distinguishes with non-negligible advantage. Finally, the only random sampling occurs in step 1 and can be de-randomized by fixing the initial random tape of $\hat{\mathsf{A}}$ and; once the random tape is fixed, all the subsequent steps of the reduction are deterministic and thus $\hat{\mathsf{A}}$ is deterministic. This concludes the proof. □

□

Combining together Thm. 5 with Cor. 1–3 yields Thm. 1.

7 Conclusions

We presented new constructions of non-malleable secret sharing schemes against joint tampering with the shares, both in the setting of selective and adaptive partitioning.

Our constructions for selective partitioning are for threshold access structures and tolerate joint tampering with maximal subsets of unauthorized parties, *i.e.*, of size equal to the privacy threshold. Our construction for adaptive partitioning is for general access structures, but tolerates joint tampering with much smaller subsets of size $k \in O(\sqrt{\log n})$ (where n is the number of parties) and under some restrictions on the way the partitions are determined by the attacker. Removing the latter limitation is an intriguing open question.

The above results hold for any *a priori* fixed bound $p > 0$ on the number of tampering queries, and under computational assumptions. We leave it as an open problem to design *continuously* non-malleable (*i.e.*, for $p = p(\lambda)$ being an arbitrary polynomial in the security parameter) secret sharing schemes tolerating joint tampering under selective/adaptive partitioning.

Another interesting question would be to improve the rate, *i.e.*, the ratio between message size and maximal size of a share, for non-malleable secret sharing against joint tampering. Note that, in the computational setting, it is always possible to boost the rate as follows: First, share the secret key $\kappa \in \{0,1\}^\lambda$ of an authenticated symmetric encryption using a secret sharing scheme with poor rate, obtaining shares $s_1, \ldots, s_n$; hence, encrypt the message m using κ, obtaining a ciphertext c, and define the final i-th share to be $s_i^* = (c, s_i)$. Such a rate-optimizing compiler was originally analyzed in the setting of non-malleable codes [DPW10, AAG$^+$16, CFV19], and more recently in the setting of non-malleable secret sharing against independent tampering [FV19]. While this transformation may be proven secure even in the setting of joint tampering with the shares, it yields a rate asymptotically approaching one, which is still far from the optimal share size of $O(\mu/n)$ [Kra94] (where μ is the message size).

Acknowledgments

We thank Ashutosh Kumar for clarifications on the tampering model in [GK18a] and for pointing out an issue in a previous version of the proof of Thm. 4 (leading to the restriction of semi-adaptive partitioning).

References

[AAG$^+$16] Divesh Aggarwal, Shashank Agrawal, Divya Gupta, Hemanta K. Maji, Omkant Pandey, and Manoj Prabhakaran. Optimal computational split-state non-malleable codes. In Eyal Kushilevitz and Tal Malkin, editors, *TCC 2016-A, Part II*, volume 9563 of *LNCS*, pages 393–417. Springer, Heidelberg, January 2016.

[ADKO15a] Divesh Aggarwal, Yevgeniy Dodis, Tomasz Kazana, and Maciej Obremski. Non-malleable reductions and applications. In Rocco A. Servedio and Ronitt Rubinfeld, editors, *47th ACM STOC*, pages 459–468. ACM Press, June 2015.

[ADKO15b] Divesh Aggarwal, Stefan Dziembowski, Tomasz Kazana, and Maciej Obremski. Leakage-resilient non-malleable codes. In Yevgeniy Dodis and Jesper Buus Nielsen, editors, *TCC 2015, Part I*, volume 9014 of *LNCS*, pages 398–426. Springer, Heidelberg, March 2015.

[ADN$^+$19a] 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 Alexandra Boldyreva and Daniele Micciancio, editors, *CRYPTO 2019, Part II*, volume 11693 of *LNCS*, pages 510–539. Springer, Heidelberg, August 2019.

[ADN$^+$19b] Divesh Aggarwal, Nico Döttling, Jesper Buus Nielsen, Maciej Obremski, and Erick Purwanto. Continuous non-malleable codes in the 8-split-state model. In Yuval Ishai and Vincent Rijmen, editors, *EUROCRYPT 2019, Part I*, volume 11476 of *LNCS*, pages 531–561. Springer, Heidelberg, May 2019.

[AKO17] Divesh Aggarwal, Tomasz Kazana, and Maciej Obremski. Inception makes non-malleable codes stronger. In Yael Kalai and Leonid Reyzin, editors, *TCC 2017, Part II*, volume 10678 of *LNCS*, pages 319–343. Springer, Heidelberg, November 2017.

[AO19] Divesh Aggarwal and Maciej Obremski. A constant-rate non-malleable code in the split-state model. Cryptology ePrint Archive, Report 2019/1299, 2019. https://eprint.iacr.org/2019/1299.

[BFV19] Gianluca Brian, Antonio Faonio, and Daniele Venturi. Continuously non-malleable secret sharing for general access structures. In Dennis Hofheinz and Alon Rosen, editors, *TCC 2019, Part II*, volume 11892 of *LNCS*, pages 211–232. Springer, Heidelberg, December 2019.

[Bla79] G. R. Blakley. Safeguarding cryptographic keys. *Proceedings of AFIPS 1979 National Computer Conference*, 48:313–317, 1979.

[BS19] Saikrishna Badrinarayanan and Akshayaram Srinivasan. Revisiting non-malleable secret sharing. In Yuval Ishai and Vincent Rijmen, editors, *EUROCRYPT 2019, Part I*, volume 11476 of *LNCS*, pages 593–622. Springer, Heidelberg, May 2019.

[CDV94] Marco Carpentieri, Alfredo De Santis, and Ugo Vaccaro. Size of shares and probability of cheating in threshold schemes. In Tor Helleseth, editor, *EUROCRYPT'93*, volume 765 of *LNCS*, pages 118–125. Springer, Heidelberg, May 1994.

[CFV19] Sandro Coretti, Antonio Faonio, and Daniele Venturi. Rate-optimizing compilers for continuously non-malleable codes. In Robert H. Deng, Valérie Gauthier-Umaña, Martín Ochoa, and Moti Yung, editors, *ACNS 19*, volume 11464 of *LNCS*, pages 3–23. Springer, Heidelberg, June 2019.

[CG14] Mahdi Cheraghchi and Venkatesan Guruswami. Non-malleable coding against bit-wise and split-state tampering. In Yehuda Lindell, editor, *TCC 2014*, volume 8349 of *LNCS*, pages 440–464. Springer, Heidelberg, February 2014.

[CGL16] Eshan Chattopadhyay, Vipul Goyal, and Xin Li. Non-malleable extractors and codes, with their many tampered extensions. In Daniel Wichs and Yishay Mansour, editors, *48th ACM STOC*, pages 285–298. ACM Press, June 2016.

[CL18] Eshan Chattopadhyay and Xin Li. Non-malleable extractors and codes for composition of tampering, interleaved tampering and more. Cryptology ePrint Archive, Report 2018/1069, 2018. https://eprint.iacr.org/2018/1069.

[DKO13] Stefan Dziembowski, Tomasz Kazana, and Maciej Obremski. Non-malleable codes from two-source extractors. In Ran Canetti and Juan A. Garay, editors, *CRYPTO 2013, Part II*, volume 8043 of *LNCS*, pages 239–257. Springer, Heidelberg, August 2013.

[DP07] Stefan Dziembowski and Krzysztof Pietrzak. Intrusion-resilient secret sharing. In *48th FOCS*, pages 227–237. IEEE Computer Society Press, October 2007.

[DPW10] Stefan Dziembowski, Krzysztof Pietrzak, and Daniel Wichs. Non-malleable codes. In Andrew Chi-Chih Yao, editor, *ICS 2010*, pages 434–452. Tsinghua University Press, January 2010.

[FMNV14] Sebastian Faust, Pratyay Mukherjee, Jesper Buus Nielsen, and Daniele Venturi. Continuous non-malleable codes. In Yehuda Lindell, editor, *TCC 2014*, volume 8349 of *LNCS*, pages 465–488. Springer, Heidelberg, February 2014.

[FNSV18] Antonio Faonio, Jesper Buus Nielsen, Mark Simkin, and Daniele Venturi. Continuously non-malleable codes with split-state refresh. In Bart Preneel and Frederik Vercauteren, editors, *ACNS 18*, volume 10892 of *LNCS*, pages 121–139. Springer, Heidelberg, July 2018.

[FV19] Antonio Faonio and Daniele Venturi. Non-malleable secret sharing in the computational setting: Adaptive tampering, noisy-leakage resilience, and improved rate. In Alexandra Boldyreva and Daniele Micciancio, editors, *CRYPTO 2019, Part II*, volume 11693 of *LNCS*, pages 448–479. Springer, Heidelberg, August 2019.

[GK18a] Vipul Goyal and Ashutosh Kumar. Non-malleable secret sharing. In Ilias Diakonikolas, David Kempe, and Monika Henzinger, editors, *50th ACM STOC*, pages 685–698. ACM Press, June 2018.

[GK18b] Vipul Goyal and Ashutosh Kumar. Non-malleable secret sharing for general access structures. In Hovav Shacham and Alexandra Boldyreva, editors, *CRYPTO 2018, Part I*, volume 10991 of *LNCS*, pages 501–530. Springer, Heidelberg, August 2018.

[KMS19] Ashutosh Kumar, Raghu Meka, and Amit Sahai. Leakage-resilient secret sharing against colluding parties. In David Zuckerman, editor, *60th FOCS*, pages 636–660. IEEE Computer Society Press, November 2019.

[Kra94] Hugo Krawczyk. Secret sharing made short. In Douglas R. Stinson, editor, *CRYPTO'93*, volume 773 of *LNCS*, pages 136–146. Springer, Heidelberg, August 1994.

[Li17] Xin Li. Improved non-malleable extractors, non-malleable codes and independent source extractors. In Hamed Hatami, Pierre McKenzie, and Valerie King, editors, *49th ACM STOC*, pages 1144–1156. ACM Press, June 2017.

[LL12] Feng-Hao Liu and Anna Lysyanskaya. Tamper and leakage resilience in the split-state model. In Reihaneh Safavi-Naini and Ran Canetti, editors, *CRYPTO 2012*, volume 7417 of *LNCS*, pages 517–532. Springer, Heidelberg, August 2012.

[NS20] Jesper Buus Nielsen and Mark Simkin. Lower bounds for leakage-resilient secret sharing. In Anne Canteaut and Yuval Ishai, editors, *EUROCRYPT 2020, Part I*, volume 12105 of *LNCS*, pages 556–577. Springer, Heidelberg, May 2020.

[OPVV18] Rafail Ostrovsky, Giuseppe Persiano, Daniele Venturi, and Ivan Visconti. Continuously non-malleable codes in the split-state model from minimal assumptions. In Hovav Shacham and Alexandra Boldyreva, editors, *CRYPTO 2018, Part III*, volume 10993 of *LNCS*, pages 608–639. Springer, Heidelberg, August 2018.

[RB89] Tal Rabin and Michael Ben-Or. Verifiable secret sharing and multiparty protocols with honest majority (extended abstract). In *21st ACM STOC*, pages 73–85. ACM Press, May 1989.

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

[SV19] Akshayaram Srinivasan and Prashant Nalini Vasudevan. Leakage resilient secret sharing and applications. In Alexandra Boldyreva and Daniele Micciancio, editors, *CRYPTO 2019, Part II*, volume 11693 of *LNCS*, pages 480–509. Springer, Heidelberg, August 2019.