Non-Malleable Secret Sharing in the Computational Setting: Adaptive Tampering, Noisy-Leakage Resilience, and Improved Rate

Antonio Faonio[1]* and Daniele Venturi[2]

[1] IMDEA Software Institute, Madrid, Spain
[2] Department of Computer Science, Sapienza University of Rome, Italy

Abstract. We revisit the concept of *non-malleable* secret sharing (Goyal and Kumar, STOC 2018) in the computational setting. In particular, under the assumption of one-to-one one-way functions, we exhibit a *computationally* private, *threshold* secret sharing scheme satisfying all of the following properties.

- **Continuous non-malleability:** No computationally-bounded adversary tampering independently with all the shares can produce mauled shares that reconstruct to a value related to the original secret. This holds even in case the adversary can tamper *continuously*, for an *unbounded* polynomial number of times, with the same target secret sharing, where the next sequence of tampering functions, as well as the subset of shares used for reconstruction, can be chosen *adaptively* based on the outcome of previous reconstructions.
- **Resilience to noisy leakage:** Non-malleability holds even if the adversary can additionally leak information independently from all the shares. There is no bound on the length of leaked information, as long as the overall leakage does not decrease the min-entropy of each share by too much.
- **Improved rate:** The information rate of our final scheme, defined as the ratio between the size of the message and the maximal size of a share, asymptotically approaches 1 when the message length goes to infinity.

Previous constructions achieved information-theoretic security, sometimes even for arbitrary access structures, at the price of *at least one* of the following limitations: (i) Non-malleability only holds against one-time tampering attacks; (ii) Non-malleability holds against a bounded number of tampering attacks, but both the choice of the tampering functions and of the sets used for reconstruction is non-adaptive; (iii) Information rate asymptotically approaching zero; (iv) No security guarantee in the presence of leakage.

Keywords: Secret sharing, non-malleability, leakage resilience, computational security.

* Supported by the Spanish Government through the projects Datamantium (ref. RTC-2016-4930-7), SCUM (RTI2018-102043-B-I00), and ERC2018-092822, and by the Madrid Regional Government under project BLOQUES (ref. S2018/TCS-4339).

1 Introduction

In a secret sharing (SS) scheme, a trusted dealer divides a secret message m into shares that are distributed to n parties, in such a way that any authorized subset of parties can efficiently determine the secret, whereas unauthorized subsets of parties have (statistically) no information about the message. In this paper, we focus on *threshold* secret sharing (TSS), where the unauthorized subsets are those with at most $\tau - 1$ players, for a parameter $\tau \leq n$.

The above type of SS is also known as τ-out-of-n TSS, and was originally introduced by Shamir [56] and Blakey [14]. SS has found many applications to cryptography, ranging from data storage [45] and threshold cryptography [28], to secure message transmission [31], multi-party computation [40,20,12], and private circuits [46,38,9].

An important parameter of an SS scheme is its *information rate*, defined as the ratio between the size of the message and the maximal size of a share. It is well-known that the best possible information rate for TSS satisfying statistical privacy is 1, meaning that the size of each share must at least be equal to that of the message being shared [11].

1.1 Non-Malleable Secret Sharing

Classical SS offers no guarantee in the presence of a *tampering adversary* modifying (possibly all!) the shares. Motivated by this shortcoming, Goyal and Kumar [42] introduced *one-time non-malleable* secret sharing (NMSS), which intuitively guarantees that even if all of the shares are tampered once, the reconstructed message is either equal to the original shared value or independent of it. The only limitation is that the adversary is restricted to change the shares *independently*, a model sometimes known under the name of *individual tampering*. As usual, in order to reconstruct the secret, only $\varrho \leq n$ shares are required, and typically the reconstruction threshold ϱ equals the privacy threshold τ.

Recently, the topic of NMSS has received a lot of attention. We summarize the state of the art below, and in Tab. 1.

- In their original paper, Goyal and Kumar [42] gave a construction of NMSS with 1-time non-malleability against individual tampering. The rate of this construction is $\Theta(\frac{1}{n \log \mu})$, where μ is the size of the message. In the same paper, the authors also propose a more complex construction that satisfies 1-time non-malleability in a stronger model where the adversary is allowed to jointly tamper subsets of up to $\tau - 1$ shares.
  In [43], the same authors construct NMSS satisfying 1-time non-malleability against individual and joint tampering, and further supporting arbitrary monotone access structures. The rate of these constructions asymptotically approaches zero when the length of the message goes to infinity.
- Badrinarayanan and Srinivasan [10] construct NMSS with improved rate. In particular, they put forward a stronger security model called p-time non-malleability, in which the adversary can tamper with the same target secret

sharing $s = (s_1, \ldots, s_n)$ for $p \geq 1$ times, by *non-adaptively* specifying sequences of tampering functions

$$(f_1^{(1)}, \ldots, f_n^{(1)}), \ldots, (f_1^{(p)}, \ldots, f_n^{(p)}) \tag{1}$$

yielding mauled shares $\tilde{s}^{(q)} = (\tilde{s}_1^{(q)}, \ldots, \tilde{s}_n^{(q)})$, for each $q \in [p]$. Non-malleability here means that for every reconstruction set $\mathcal{T}$ with size at least τ, fixed *before* tampering takes place, the secrets reconstructed out of $\tilde{s}_{\mathcal{T}}^{(1)}, \ldots, \tilde{s}_{\mathcal{T}}^{(p)}$ are independent of the original message.

The main result of [10] are NMSS schemes with p-time non-malleability, both for threshold access structures (with $\varrho = \tau \geq 4$), and for arbitrary 4-monotone access structures, with rates, respectively, $\Theta\left(\frac{1}{p^3 \cdot \tau \cdot \log^2 n}\right)$ and $\Theta\left(\frac{1}{p^3 \cdot \tau_{\mathsf{max}} \cdot \log^2 n}\right)$ (where τ_{max} is the maximum size of a minimal authorized subset). Importantly, the maximal value of p is a priori fixed and, in fact, the shares' size can depend on it. Moreover, they proved that, in the information-theoretic setting, it is impossible to construct NMSS achieving non-malleability against an unbounded polynomial number of tampering attempts.

– Aggarwal *et al.* [2] consider a strengthening of p-time non-malleability, in which the adversary tampers non-adaptively p times, as in Eq. (1), but additionally specifies p different sets $\mathcal{T}_1, \ldots, \mathcal{T}_p$ for the reconstruction of each mauled shares $\tilde{s}^{(1)}, \ldots, \tilde{s}^{(p)}$. In other words, the requirement is now that $\tilde{s}_{\mathcal{T}_1}^{(1)}, \ldots, \tilde{s}_{\mathcal{T}_p}^{(p)}$ are independent of the original message. They dub their model p-time non-malleability under *non-adaptive concurrent reconstruction*, since the sets $\mathcal{T}_1, \ldots, \mathcal{T}_p$ are specified in a non-adaptive fashion.

Reference	Access Structure	Non-Malleability	Leakage Resilience	Rate	Assumption	Notes
[42]	Threshold ($\tau \geq 2$)	1-time	✗	$\Theta\left(\frac{1}{n \log \mu}\right)$	—	IT
	Threshold ($\tau \geq 2$)	1-time	✗	$\Theta\left(\mu^{-9}\right)$	—	JT
[43]	Arbitrary (monotone)	1-time	✗	$\Theta\left(\frac{1}{n \log \mu}\right)$	—	IT
	Threshold ($\tau = n$)	1-time	✗	$\Theta(\mu^{-6})$	—	JT
[10]	Threshold ($\tau \geq 4$)	p-time	✗	$\Theta\left(\frac{1}{p^3 \cdot \tau \cdot \log^2 n}\right)$	—	IT, NAT
	Arbitrary (4-monotone)	p-time	✗	$\Theta\left(\frac{1}{p^3 \cdot \tau_{\mathsf{max}} \cdot \log^2 n}\right)$	—	IT, NAT
[2]	Arbitrary (3-monotone)	p-time	✗	$\Theta\left(\frac{1}{n \log \mu}\right)$	–	IT, NAT, NACR
[58]	Arbitrary (4-monotone)	1-time	✗	$\Theta(1)$	–	IT
[49]	Arbitrary (monotone)	1-time	ℓ-Bounded	$\Theta\left(\frac{1}{\ell n \log n \log \mu}\right)$	–	IT
This paper	Threshold ($\tau \leq \varrho - 1$)	poly-time	Noisy	$\Omega\left(\frac{\mu}{\mu + n^2 \lambda^8}\right)$	1-to-1 OWFs	IT, ACR

Table 1: Comparison of state-of-the-art NMSS schemes. The value n denotes the number of parties, μ denotes the size of the message, ℓ denotes the leakage parameter, λ denotes the security parameter, and τ (resp. ϱ) is the privacy (resp. reconstruction) threshold in case of TSS, where $\varrho = \tau$ unless stated otherwise. In case of general access structures, τ_{max} is the maximum size of a minimal authorized subset. IT stands for "individual tampering", JT for "joint tampering", NAT for "non-adaptive tampering", NACR for "non-adaptive concurrent reconstruction", and ACR for "adaptive concurrent reconstruction".

The main result of [2] is a construction of NMSS with rate $\Theta(\frac{1}{n \log \mu})$, satisfying p-time non-malleability under non-adaptive concurrent reconstruction.
– Srinivasan and Vasudevan [58] construct the first NMSS for 4-monotone access structures, and satisfying 1-time non-malleability with rate $\Theta(1)$.
– Finally, Kumar, Meka, and Sahai [49] construct NMSS with 1-time non-malleability, but where the adversary is additionally allowed to adaptively leak information on the shares independently, i.e. they considered for the first time *leakage-resilient* NMSS (LR-NMSS). Note that here, the choice of the tampering functions can adaptively depend on the leakage. The rate of this scheme asymptotically approaches zero.

1.2 Our Contributions

All the above mentioned works construct NMSS, with different characteristics, in the *information-theoretic* setting, where both the privacy and the non-malleability of the scheme holds even against unbounded adversaries. A natural question is whether one can improve the state of the art in the *computational* setting, where the adversary for privacy and non-malleability is computationally bounded. Note that this is particularly appealing, in view of the fact that fully-fledged *continuous* non-malleability is impossible to achieve in the information-theoretic setting [10]. Hence, the following question is open:

Can we construct NMSS where a computationally-bounded *adversary can tamper* adaptively*, with the* same *target shares, for an* unbounded polynomial *number of times, and under* adaptive concurrent reconstruction*?*

In this work, we answer the above question affirmatively for the case of threshold access structures and individual tampering, assuming 1-to-1 one-way functions (OWFs). Our final scheme has rate asymptotically approaching 1, and furthermore satisfies leakage resilience.

Theorem 1 (Main Theorem, Informal). *Let $\tau, \varrho, n \in \mathbb{N}$ be such that $\tau, \varrho \leq n$ and $\tau \leq \varrho - 1$. Assuming* 1*-to-*1 *OWFs, there exists* noisy-leakage-resilient, continuously non-malleable *τ-out-of-n secret sharing (LR-CNMSS) under* adaptive concurrent reconstruction *(where at least ϱ parties are needed to reconstruct the secret), with information rate (asymptotically) one.*

We observe that leakage resilience holds in the so-called *noisy*-leakage model, where the actual amount of information that can be leaked independently from each share is unbounded, as long as the uncertainty of each share does not decrease by too much. Also, notice that there is a minimal gap[3] between the reconstruction threshold ϱ and the privacy threshold τ (i.e., $\tau \leq \varrho - 1$). Interestingly, as we explain in §4.2, CNMSS cannot exist unconditionally for the optimal parameters $\tau = \varrho$, and thus our work leaves open the question of constructing

[3] Secret sharing scheme with a gap between reconstruction and privacy are known in literature as *ramp* secret sharing scheme.

TSS where both privacy and continuous non-malleability hold statistically, as long as $\tau < \varrho$.

A final remark is that the definition of continuous non-malleability uses a special self-destruct feature, in which after the first *invalid* mauled secret sharing is found (i.e., a collection of shares $\tilde{s}_{\mathcal{T}_q}^{(q)}$ whose reconstruction equals an error symbol $\perp$), the answer to all *future* tampering queries is by default set to be $\perp$. As we show in §4.3, such a feature is necessary, in the sense that without it no CNMSS exists (even without considering leakage and concurrent reconstruction).

1.3 Tamper-Resilient Threshold Signatures

As an application, we consider a generalization of the classical transformation from standard security to tamper-proof security via non-malleable codes [33], in the setting of threshold cryptography. For concreteness, we focus on threshold signatures, which allow to secret share a signing key among n servers, in such a way that any subset of at least ϱ servers can interact in order to produce the signature of a message. The standard security guarantee here is that an adversary corrupting up to $\tau - 1$ servers cannot forge a valid signature, even after observing several transcripts of the signing protocol with the honest servers.

Given any CNMSS, we show how to compile a non-interactive threshold signature into an interactive (2-round) threshold signature that additionally is secure in the presence of *continuous* tampering attacks. More precisely, we imagine an external forger corrupting the memory of (possibly all!) the servers independently (say via a malware installed on each of the servers), and observing several signatures produced using arbitrarily modified secret-key shares.

A similar application was recently considered in [2]. The main advantage of our model is that the attacker is allowed to tamper continuously with the memory of the servers, and further can adaptively choose the subset of servers participating in each invocation of the signature protocol; on the negative side, our adversary is not allowed to fully corrupt any of the servers, whereas in the model of [2] the forger, after tampering once, obtains the secret-key shares of $\tau - 1$ servers. In our perspective, this difference stems from the fact that [2] makes a non-black-box usage of the underlying NMSS, which allows to exploit a slightly stronger form of non-malleability which, although not formalized by the authors, seems to be met by their specific construction. (I.e., non-malleability still holds even if the attacker learns a subset of the original shares, after tampering is over; such a property is sometimes known as *augmented* non-malleability in the non-malleable codes literature [1,25].) In contrast, our compiler only makes black-box calls to the underlying primitives.

1.4 Further Related Works

Robust secret sharing. In *robust* SS (see, e.g. [54,16,55,13]), a monolithic adversary can (non-adaptively) corrupt up to τ players, and thus jointly tamper their shares. Robustness guarantees that given all the $\varrho = n$ shares, the reconstructed message is identical to the original shared value.

While robustness is a strong form of non-malleability, it is clearly impossible when more than $n/2$ shares are corrupted (even in the computational setting).

Non-malleable codes. The concept of NMSS is intimately related to the notion of non-malleable codes (NMCs) [33]. Intuitively, a NMC allows to encode a message in such a way that tampering with the resulting codeword via a function $f \in \mathcal{F}$, where $\mathcal{F}$ is a set of allowed tampering functions that is a parameter in the definition, yields a modified codeword that either decodes to the original message or to an unrelated value. Several constructions of NMCs exist in the literature, for different families $\mathcal{F}$; one of the most popular choices is to think of the tampering function as a sequence of n functions $f = (f_1, \ldots, f_n)$, where each function f_i modifies a different chunk of the codeword arbitrarily, yet independently. This is often known as the n-split-state model [33,51,32,22,21,4,19,7,3,18,17,50,47,6], the most general case being the case $n = 2$.

As shown by Aggarwal *et al.* [7], every NMC in the 2-split-state model is a 2-out-of-2 NMSS in disguise. Similarly, it is easy to see that any (leakage-resilient) *continuously* NMC (LR-CNMC) in the 2-split-state model [37,34,53,25] is a 2-out-of-2 LR-CNMSS as per our definition.

Leakage-resilient codes. When no tampering is considered, our definition of LR-CNMSS collapses to that of leakage-resilient secret sharing, as originally introduced by Davì, Dziembowski, and Venturi, for the case $n = \tau = \varrho = 2$ [27]. This topic recently received renewed attention, see, in particular, [2,58,49].

2 Technical Highlights

Intuitively, the proof of Thm. 1 proceeds in two steps. In the first step, we show how to obtain LR-CNMSS with information rate asymptotically approaching 0, assuming 1-to-1 OWFs. In the second step, we show how to boost the asymptotic rate generically, from 0 to 1, under the same assumption. Below, we explain these two steps with some details, after presenting our security model informally.

2.1 Security Model

Let Σ be an n-party TSS, with reconstruction threshold ϱ (i.e., given at least ϱ shares we can efficiently reconstruct the message) and privacy threshold τ (i.e., $\tau-1$ shares reveal no information on the message to the eyes of a computationally-bounded adversary). In order to define continuous non-malleability for TSS, we consider an efficient adversary interacting with a target secret sharing $s = (s_1, \ldots, s_n)$ of some message $m \in \mathcal{M}$, via the following queries.

– **Tampering:** The attacker can specify a sequence of efficiently-computable functions $(f_1^{(q)}, \ldots, f_n^{(q)})$, yielding mauled shares

$$\tilde{s}^{(q)} = (\tilde{s}_1^{(q)}, \ldots, \tilde{s}_n^{(q)}) = (f_1^{(q)}(s_1), \ldots, f_n^{(q)}(s_n)),$$

along with a set $\mathcal{T}_q \subseteq [n]$, with size $\tilde{\varrho} \geq \varrho$. The answer to such a query is the message $\tilde{m}^{(q)}$ which is reconstructed using the shares $\tilde{s}^{(q)}_{\mathcal{T}_q}$. The above queries can be chosen in a fully-adaptive fashion for all $q \in [p]$, where p is an *arbitrary polynomial* in the security parameter; however, after the first tampering query generating an invalid message $\perp$ during reconstruction, the system switches to a *"self-destruct mode"* in which the answer to future tampering queries is automatically set to $\perp$.

– **Leakage:** The attacker can specify an efficiently-computable function g, and an index $i \in [n]$, upon which it obtains $g(s_i)$. These queries can be chosen in a fully-adaptive fashion, as long as the uncertainty of each share conditioned on the leakage (measured via conditional average min-entropy [30]) is reduced at most by a value $\ell \in \mathbb{N}$ that is a parameter of the scheme.

The formal definition of leakage-resilient continuous non-malleability essentially says that for each pair of messages $m_0, m_1 \in \mathcal{M}$, the adversary's view in the above experiment is computationally indistinguishable in the two cases where $m = m_0$ and $m = m_1$. Note that when $n = \tau = \varrho = 2$, and further when ℓ is an upper bound on the total amount of leakage, our definition collapses to the standard notion of a LR-CNMC in the split-state model [51,7].

One might observe that our definition is game based, whereas all previous definitions of non-malleable secret sharing are simulation based. While it would be possible to give a simulation-based definition for LR-CNMSS, it is not hard to show that the two formulations are equivalent, as long as the length of the shared value is super-logarithmic in the security parameter. The same equivalence, in fact, holds true for the case of LR-CNMCs [33,53].

We also remark that the limitations of computational security and self-destruct are somewhat inherent. First, as shown by [10], no TSS scheme with $\varrho = \tau$, and satisfying statistical privacy, can achieve information-theoretic continuous non-malleability w.r.t. an arbitrary polynomial number of tampering queries; as we explain in §4.2, however, the latter might still be possible with a non-zero gap $\tau < \varrho$. Second, as we formally prove in §4.3, it is also impossible to achieve continuous non-malleability without a self-destruct capability. The latter is reminiscent of similar impossibility results in the settings of tamper-resilient cryptography and non-malleable codes [39,37]. Note that both these impossibility results hold even without considering leakage and concurrent reconstruction.

2.2 First Step: Achieving Continuous Non-Malleability (Poor Rate)

A scheme with low privacy. Consider the following simple idea, inspired by [42], how to construct a 2-out-of-n CNMSS by leveraging any CNMC in the split-state model (i.e., any 2-out-of-2 CNMSS). To share a message $m \in \mathcal{M}$, we enumerate over all the possible pairs of distinct indices smaller than n, and for each such pair we compute a 2-out-of-2 CNMSS of the message. In other words, for each subset $\mathcal{H} = \{h_1, h_2\} \in \binom{[n]}{2}$, we consider a non-malleable split-state encoding $s_{\mathcal{H}} := (s_{\mathcal{H},h_1}, s_{\mathcal{H},h_2})$ of the message m, which we assign to the indices h_1 and h_2. The final share s^*_i for party $i \in [n]$ is then defined to be the collection of all the

shares $s_{\mathcal{H},i}$, where $\mathcal{H}$ is such that $i \in \mathcal{H}$. Reconstruction is defined in the natural way, i.e. given an authorized set $\mathcal{H}' = \{h'_1, h'_2\}$, we simply ignore all the shares but $s_{\mathcal{H}'}$, and use $(s_{\mathcal{H}',h'_1}, s_{\mathcal{H}',h'_2})$ to reconstruct the message.

Intuitively, the above scheme is secure because the $\binom{n}{2}$ shares of the message m are independently sampled, and furthermore the reconstruction for an authorized set $\mathcal{H}$ is independent of all the shares but one. In particular, the 2-threshold privacy property follows easily by privacy of the underlying CNMC. As for continuous non-malleability, consider a sequence of hybrid experiments, one hybrid for each subset $\mathcal{H}$ in $\binom{[n]}{2}$ in lexicographic order: In each hybrid step, we change the distribution of the target secret sharing $s^* = (s^*_1, \ldots, s^*_n)$ by letting $(s_{\mathcal{H},h_1}, s_{\mathcal{H},h_2})$ be a 2-out-of-2 CNMSS of m_0 for all sets in $\binom{[n]}{2}$ up to $\mathcal{H}$, whereas we use m_1 to define the remaining shares.

For the proof, we can build a reduction to the continuous non-malleability of the underlying split-state encoding. In particular, the simulation of a generic tampering query of the form $(\mathcal{T}, (f_1, \ldots, f_n))$, proceeds as follows:

- If $\mathcal{T}$ and $\mathcal{H}$ do not share any index, then they cannot possibly interfere with each other. In particular, the reduction knows all the shares for the positions in $\mathcal{T}$, and therefore it can simulate the answer without even querying the underlying tampering oracle for the split-state CNMC.
- If $\mathcal{T}$ and $\mathcal{H}$ share (at least) an index, then we can use the target tampering oracle to compute the mauled shares corresponding to $\mathcal{T}$ using the tampering oracle corresponding to $\mathcal{H}$. However, there is a catch. Let, e.g., be $\mathcal{T} = \{t_1, t_2\}$ and $\mathcal{H} = \{h_1, h_2\}$, and suppose $t_2 = h_1$. To compute the tampered share $\tilde{s}_{\mathcal{T},t_2}$, we need to know the value $s_{\mathcal{H},h_1}$, which is only accessible through the tampering oracle; as a consequence, the reduction would only be able to obtain the reconstructed message corresponding to $(\tilde{s}_{\mathcal{T},t_2}, \tilde{s}_{\mathcal{T},t_1})$, which is possibly different from the reconstructed message corresponding to $(\tilde{s}_{\mathcal{T},t_1}, \tilde{s}_{\mathcal{T},t_2})$. We bypass this problem by assuming that the underlying split-state CNMC has *symmetric decoding*, namely the decoding output is invariant w.r.t. the order of the two shares. As we explain later, this property is satisfied by known schemes.

Amplifying the privacy. Intuitively, the transformation above is based on the fact that by composing a secret sharing for an access structure $\mathcal{A}$ with a secret sharing for an access structure $\mathcal{A}'$, we obtain a new secret sharing for access structure $\mathcal{A} \cup \mathcal{A}'$. Unfortunately, we cannot generalize this idea to go from ϱ-out-of-ϱ to ϱ-out-of-n secret sharing for any $\varrho \leq n$, as for efficiency we need $\binom{n}{\varrho} \approx n^\varrho$ to be polynomial in n.

The key idea behind our main construction of CNMSS is to compose together $\binom{n}{2}$ secret sharing schemes with different access structures, such that their union gives the desired ϱ-threshold access structure. Specifically, consider the following construction of a ϱ-out-of-n TSS based on a split-state CNMC, on an authenticated secret-key encryption (AE) scheme, and on an auxiliary $(\varrho - 3)$-out-of-$(n - 2)$ TSS.

For a fixed pair of indices $\mathcal{H} = \{h_1, h_2\} \in \binom{[n]}{2}$, pick a uniformly random key $\kappa_{\mathcal{H}}$ for the AE scheme, compute a split-state encoding of $\kappa_{\mathcal{H}}$, and call the resulting shares $(s_{\mathcal{H},h_1}, s_{\mathcal{H},h_2})$; hence, encrypt the message m under the key $\kappa_{\mathcal{H}}$ obtaining a ciphertext $c_{\mathcal{H}}$, and secret share $c_{\mathcal{H}}$ using the auxiliary TSS, yielding shares $(s_{\mathcal{H},h_3}, \ldots, s_{\mathcal{H},h_n})$ where $\{h_3, \ldots, h_n\} = [n] \setminus \mathcal{H}$. Notice that this scheme has access structure $\mathcal{A}_{\mathcal{H}} = \{\mathcal{S} \subset [n] : |\mathcal{S}| \geq \varrho, \mathcal{H} \subset \mathcal{S}\}$. By repeating the above procedure for each set $\mathcal{H} \in \binom{[n]}{2}$, we obtain that the final share s_i^* for party $i \in [n]$ is the collection of all the shares $s_{\mathcal{H},i}$, so that $\bigcup_{\mathcal{H} \in \binom{[n]}{2}} \mathcal{A}_{\mathcal{H}}$ yields the ϱ-threshold access structure, as desired. Moreover, the size of each share is still polynomial in the number of parties.

The proof of threshold privacy is rather straightforward, at least if we set the privacy threshold for the final scheme to be $\tau \leq \varrho - 2$. However, in the computational setting, we can even show privacy $\tau \leq \varrho - 1$. The key idea is that either the adversary has enough shares to reconstruct the underlying ciphertext (but in this case it does not have access to the secret key, and therefore it learns nothing by semantic security of the encryption scheme), or, the adversary knows at most $\varrho - 3$ shares of the ciphertext (which by perfect privacy of the auxiliary TSS reveal nothing about the ciphertext).

Proving continuous non-malleability. The intuition for non-malleability of the secret sharing scheme with access structure $\mathcal{A}_{\mathcal{H}}$ is that by tampering the shares corresponding to indices h_1, h_2, the adversary either obtains the original key or a completely unrelated value: In the former case, by the authenticity of the AE scheme, the adversary cannot produce a new ciphertext that decrypts correctly; in the latter case, by the semantic security of the AE scheme, the adversary cannot produce a ciphertext that decrypts to a related message (under the unrelated key generated via tampering).

Next, we analyze how continuous non-malleability is preserved when we compose together the different secret sharing schemes with access structure $\mathcal{A}_{\mathcal{H}}$ (for $\mathcal{H} \in \binom{[n]}{2}$). In contrast to the simple composition for the 2-out-of-n CNMSS construction hinted above, in the new composed scheme the share of party i consists of both the shares of a split-state encoding of a key, and the shares of a ciphertext under an auxiliary standard TSS. Hence, in a tampering query, the adversary could swap these two kinds of shares, with the consequence that the reconstruction procedure of the underlying $(\varrho - 3)$-out-of-$(n - 2)$ TSS would depend on one of the two shares of the split-state CNMC. To resolve this problem we rely on two different ideas: First, we additionally assume that the split-state CNMC is resilient to *noisy leakage*; second, we make sure that the reconstruction procedure of the auxiliary TSS does not leak information about single shares.

The second idea is the most important one. In fact, by simply assuming leakage resilience we could at most tolerate an a priori bounded number of tampering queries. The reason for this is that, even if each reconstruction leaks just a single bit of a share $s_{\mathcal{H},i}$ under the split-state CNMC, after $|s_{\mathcal{H},i}|$ consecutive tampering queries this share could be leaked without provoking a self-destruct. The latter is better understood by looking at Shamir's TSS, where to share $m \in \mathcal{M}$ we pick

a random polynomial of degree ϱ that evaluates to m at point 0, and distribute to the i-th party the share s_i obtained by evaluating the polynomial at point $i \in [n]$. The reconstruction algorithm, given any set of ϱ shares s_i, interpolates the corresponding points, thus obtaining a polynomial that is evaluated on the origin. It is easy to see that such a reconstruction procedure, under tampering attacks, potentially leaks a lot of information about the single points (without the risk of self-destruct). In particular, the reconstruction algorithm is a linear function of the shares, and thus perturbing one point by a multiplicative factor, allows to recover the value of a share in full via a single tampering query.

We now show how to avoid the above leakage. Fix some index $i \in [n]$ for the i-th share. Given an authorized set of size ϱ, we let our reconstruction procedure select two different subsets[4] of size $\varrho - 3$, such that one subset includes the index i, whereas the second subset excludes it. Thus, we run the standard reconstruction procedure twice, one for each subset, and we accept the reconstructed message if and only if the two runs yield the same value, otherwise we return an error message (which triggers a self-destruct). The main observation is that the second run of the reconstruction algorithm is independent of $s_{\mathcal{H},i}$, and thus, conditioned on the returned message not being $\perp$, the output of the reconstruction is independent of $s_{\mathcal{H},i}$. On the other hand, when the returned message is equal to $\perp$, the output of the reconstruction could indeed leak information about the share with index i, but notice that this situation triggers a self-destruct, and thus such leakage happens only once.

More in details, for the proof we perform a hybrid argument over all sets $\mathcal{H} = \{h_1, h_2\} \in \binom{[n]}{2}$, where at each step we change the shared value of the secret sharing relative to the access structure $\mathcal{A}_{\mathcal{H}}$. To show that each pair of adjacent hybrids are computationally indistinguishable, we consider a reduction to the continuous non-malleability of the underlying split-state CNMC. Denote by $(s_{\mathcal{H},h_1}, s_{\mathcal{H},h_2})$ the target codeword. Note that the reduction can sample all the randomness necessary to create the shares $s^*_1, \ldots, s^*_n$, except for the shares $s^*_{h_1}, s^*_{h_2}$ for which the values $s_{\mathcal{H},h_1}, s_{\mathcal{H},h_2}$ are missing and will be defined through the target tampering oracle. Now, suppose the adversary sends a tampering query $(\mathcal{T} = \{t_1, \ldots, t_\varrho\}, (f_1, \ldots, f_n))$, and suppose that $t_1 = h_1$ and $t_3 = h_2$.[5] While the reduction cannot simulate the tampered shares $\tilde{s}^*_{h_1}$ and $\tilde{s}^*_{h_2}$ locally, it can use the tampering oracle to obtain the decoding relative to the split-state codeword $(\tilde{s}_{\mathcal{T},t_1}, \tilde{s}_{\mathcal{T},t_2})$; in fact, $\tilde{s}_{\mathcal{T},t_2}$ can be computed by the reduction itself—as it knows the share $s^*_{t_2}$ in full—and hard-wired into the description of the right tampering function, whereas the value $\tilde{s}_{\mathcal{T},t_1}$ can be perfectly emulated inside the tampering oracle by hard-wiring into the left tampering function all the information known about $s^*_{h_1}$.

[4] In retrospect, this is the reason why we set the reconstruction/privacy threshold of the underlying TSS to $\varrho - 3$ (i.e., 2 shares for decoding the non-malleable encoding and $\varrho - 3 + 1 = \varrho - 2$ shares to run the reconstruction procedure of the TSS twice).

[5] Clearly, the reduction needs to handle many other cases; however, this particular case is enough to illustrate our technique.

In order to complete the simulation, the reduction still needs to run twice the reconstruction process of the underlying TSS, given the tampered shares $\tilde{s}_{\mathcal{T},t_3}, \ldots, \tilde{s}_{\mathcal{T},t_\varrho}$. Note that since the values $\tilde{s}_{\mathcal{T},t_4}, \ldots, \tilde{s}_{\mathcal{T},t_\varrho}$ can be computed locally, the reduction can perform one reconstruction (yielding a first reconstructed ciphertext c_1). However, in order to run the second reconstruction, it needs the value $\tilde{s}_{\mathcal{T},t_3}$ which is not directly available, as it might depend on $s_{\mathcal{H},t_3} = s_{\mathcal{H},h_2}$. The idea is then to get the second ciphertext c_2 via a leakage query. We claim that, as long as $c_1 = c_2$, such leakage does not decrease the min-entropy of $s_{\mathcal{H},h_2}$; roughly speaking, the reason is that $c_2 = c_1$ can be also computed as a function of $\tilde{s}_{\mathcal{H},t_4}, \ldots, \tilde{s}_{\mathcal{H},t_\varrho}$, which are known by the reduction and independent of $s_{\mathcal{H},t_3}$.

Notice that the double-reconstruction trick—i.e., running the reconstruction procedure twice, in the above example one with t_3 and one without—is sufficient to prove that the reconstruction does not leak information about one specific share. However, we need to ensure that no information about any of the shares is leaked. One simple idea would be to lift the previous argument by repeating the reconstruction for all subsets of size $\varrho-3$. Nicely, in the case of, e.g. Shamir's TSS this is not necessary. In fact, we can have a more efficient reconstruction procedure that only checks two subsets. This is because if two different subsets of size $\varrho - 3$ yield polynomials with identical evaluation in the origin, then they must encode the same polynomial, and since these two subsets cover an entire authorized set, then we are ensured that using any other subset would yield the same reconstructed message.

Instantiating the construction. All that remains is to construct a split-state CNMC with the special symmetric decoding feature, and for which the non-malleability property still holds even in the presence of noisy (independent) leakage from the left and right shares.

We do this by revisiting the recent construction of Ostrovsky *et al.* [53], which gives a split-state CNMC assuming non-interactive, perfectly binding commitments (which in turn can be based on 1-to-1 OWFs). In their scheme, a split-state encoding of a message m is a pair of values $(L, R) = ((com, L'), (com, R'))$, where com is a non-interactive commitment to the message m using randomness δ, and (L', R') is a split-state encoding of the string $m||\delta$ obtained by running an auxiliary code satisfying leakage-resilient one-time non-malleability, in the information-theoretic setting and in the bounded-leakage model. The decoding algorithm first checks that the left and right share contain the same commitment. If not, it returns $\perp$. Else, it decodes (L', R') obtaining a string $m' = m||\delta$, and returns m if and only if δ is a valid opening of com w.r.t. m.

Our first observation is that the above code satisfies symmetric decoding, as long as the inner encoding (L', R') does. Additionally, we extend the security proof of [53] to show that if the auxiliary split-state code is secure in the noisy-leakage model, so is the final encoding. As a side result, and thanks to the power of noisy leakage, we even obtain a simpler proof.

The missing piece of the puzzle is then to exhibit a split-state code satisfying leakage-resilient one-time non-malleability, in the information-theoretic setting and in the noisy-leakage model, and with symmetric decoding. Luckily, it turns

out that the coding scheme by Aggarwal *et al.* [7], based on the inner-product extractor [23], already satisfies all these requirements. We refer the interested reader to the full version of this paper [36] for the details.

2.3 Second Step: Amplifying the Rate

Next, we describe another generic transformation yielding LR-CNMSS with information rate asymptotically approaching 1, starting from a LR-CNMSS with asymptotic rate 0, and an AE scheme. Such transformations, in the setting of non-malleable codes, are sometimes known as rate compilers [8,1,25].

Our rate compiler generalizes a construction by Agrawal *et al.* [1] in the setting of split-state NMCs, which has been very recently analyzed also in the case of continuous tampering [25]. In order to secret share the message $m \in \mathcal{M}$, we first sample a uniformly random key κ for the AE scheme, and then we encrypt the message m under this key, yielding a ciphertext c. Hence, we secret share the key κ using the underlying rate-0 secret sharing scheme, yielding n shares $(\kappa_1, \ldots, \kappa_n)$. Finally, we set the share of party $i \in [n]$ to be $s_i = (\kappa_i, c)$. The reconstruction procedure, given ϱ shares, first checks that all shares contain the same ciphertext c. If not, an error is triggered. Else, the secret key is reconstructed from the shares and used to decrypt the unique ciphertext c.

Note that the length of the secret key is independent of the size of the message, and thus the above construction achieves information rate asymptotically approaching 1. As for security, it is not hard to show that the compiled scheme inherits the threshold privacy property from the underlying rate-0 secret sharing. Here, we additionally need to rely on the semantic security of the AE scheme to argue that the ciphertext c reveals nothing about the message.

Proving continuous non-malleability. Turning to continuous non-malleability, the main step of the proof is a game hop in which the values $(\kappa_1, \ldots, \kappa_n)$ result from a secret sharing of an unrelated key $\kappa' \neq \kappa$. In order to establish the indistinguishability between this modified experiment and the original experiment, we consider a reduction to the continuous non-malleability of the underlying LR-CNMSS. Such a reduction can interact with a target secret sharing $(\kappa_1, \ldots, \kappa_n)$ that is either a secret sharing of κ or of κ'. The main obstacle, here, comes from the simulation of tampering queries. In fact, although the reduction can perfectly emulate the distribution of the individual shares $s_i = (\kappa_i, c)$ inside the tampering oracle, as the ciphertext c can be sampled locally, the difficulty is that to emulate the output of the reconstruction w.r.t. a given subset $\mathcal{T} = \{t_1, \ldots, t_{\tilde{\varrho}}\}$ we need to: (i) ensure that all of the mauled shares $\tilde{s}_{t_j} = (\tilde{\kappa}_{t_j}, \tilde{c}_{t_j})$ actually contain the same ciphertext, i.e. $\tilde{c}_{t_1} = \ldots = \tilde{c}_{t_{\tilde{\varrho}}} = \tilde{c}$, and (ii) use the mauled secret key $\tilde{\kappa}$ received by the reduction in response to a tampering query in order to obtain the decryption of the unique ciphertext $\tilde{c}$ (if such a ciphertext exists).

We overcome both of the above obstacles by exploiting the fact that the starting CNMSS is resilient to noisy leakage. This is crucial in our setting, since the size of the ciphertext might very well exceed the maximal length of a share of the secret key. Hence, generalizing a trick from [34,25], we proceed to check

equality of all the ciphertexts in a block-wise fashion, by leaking blocks of λ bits from each share, where λ is the security parameter. This leakage routine continues until eventually we obtain the entire ciphertext $\tilde{c}$, unless some of the blocks leaked from each share differ, in which case we answer the tampering query by $\perp$ and trigger a self-destruct.

It remains to show that the above methodology does not result in too much leakage. Intuitively, this holds because up to the point where the leaked blocks of the ciphertexts are all the same, the leakage on each share can be thought of as a function of the other shares, so that this leakage does not decrease the min-entropy of each share more than conditioning on the other shares, which is fine since in known constructions the mutual information between the shares is very low. On the other hand, when a self-destruct is triggered, we reveal only λ bits of information; by a standard argument, this causes a min-entropy drop of roughly λ bits, which again is tolerated by the underlying scheme.

3 Preliminaries

3.1 Standard Cryptographic Primitives

Threshold Secret Sharing An n-party secret sharing scheme Σ consists of a pair of polynomial-time algorithms $(\mathsf{Share}, \mathsf{Rec})$ specified as follows: (i) The randomized sharing algorithm Share takes as input a message $m \in \mathcal{M}$, and outputs n shares $s_1, \ldots, s_n$ where each $s_i \in \mathcal{S}_i$; (ii) The deterministic algorithm Rec takes as input a certain number of candidate shares and outputs a value in $\mathcal{M} \cup \{\perp\}$. Given $s = (s_1, \ldots, s_n)$ and a subset $\mathcal{I} \subseteq [n]$, we often write $s_{\mathcal{I}}$ to denote the shares $(s_i)_{i \in \mathcal{I}}$.

Definition 1 (Threshold secret sharing). *Let $n, \tau, \varrho \in \mathbb{N}$, with $\tau \leq \varrho \leq n$. We say that $\Sigma = (\mathsf{Share}, \mathsf{Rec})$ is an (n, τ, ϱ)-threshold secret sharing scheme ((n, τ, ϱ)-TSS for short) over message space $\mathcal{M}$ and share space $\mathcal{S} = \mathcal{S}_1 \times \cdots \times \mathcal{S}_n$ if it is an n-party secret sharing with the following properties.*

(i) **ϱ-Threshold Reconstruction:** *For all messages $m \in \mathcal{M}$, and for all subsets $\mathcal{I} \subseteq [n]$ such that $|\mathcal{I}| \geq \varrho$, we have that $\mathsf{Rec}((\mathsf{Share}(m))_{\mathcal{I}}) = m$, with overwhelming probability over the randomness of the sharing algorithm.*

(ii) **τ-Threshold Privacy:** *For all pairs of messages $m_0, m_1 \in \mathcal{M}$, and for all unqualified subsets $\mathcal{U} \subseteq [n]$ such that $|\mathcal{U}| < \tau$, we have that*

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

If the ensembles $\{(\mathsf{Share}(1^\lambda, m_0))_{\mathcal{U}}\}_{\lambda \in \mathbb{N}}$ and $\{(\mathsf{Share}(1^\lambda, m_1))_{\mathcal{U}}\}_{\lambda \in \mathbb{N}}$ are statistically close (resp. identically distributed), we speak of statistical *(resp.* perfect*) τ-threshold privacy.*

Typical TSS schemes achieve the optimal parameters $\varrho = \tau$. However, having a small gap between the privacy and reconstruction threshold makes sense too, and looking ahead our constructions will have minimal gap $\varrho - \tau \geq 1$.

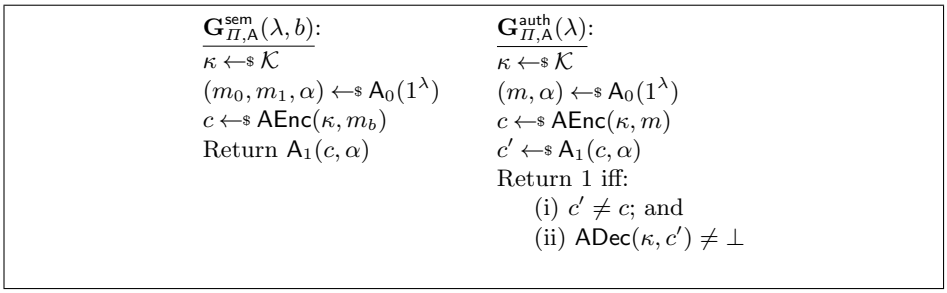

Fig. 1: Experiments defining security of authenticated encryption.

Special reconstruction. We will need TSS schemes meeting an additional reconstruction property, called *special reconstruction*. This means that for any subset $\mathcal{I} \subset [n]$ of size at least $\varrho + 1$, and for any $m \in \mathcal{M}$ which is secret shared as in $(s_1,\ldots,s_n) \leftarrow\!\!\$\ \mathsf{Share}(m)$, if there are two subsets $\mathcal{I}_1,\mathcal{I}_2 \subset \mathcal{I}$ of size ϱ such that

$$\mathsf{Rec}((s_i)_{i\in\mathcal{I}_1}) = \mathsf{Rec}((s_i)_{i\in\mathcal{I}_2}),$$

then the above equation holds for all subsets $\mathcal{I}_1,\mathcal{I}_2 \subset \mathcal{I}$ of size ϱ.

Authenticated Encryption A (secret-key) authenticated encryption (AE) scheme is a tuple of polynomial-time algorithms $\Pi = (\mathsf{KGen},\mathsf{AEnc},\mathsf{ADec})$ specified as follows: (i) The randomized algorithm KGen takes as input the security parameter $\lambda \in \mathbb{N}$, and outputs a uniform key $\kappa \leftarrow\!\!\$\ \mathcal{K}$; (ii) The randomized algorithm AEnc takes as input a key $\kappa \in \mathcal{K}$ and a message $m \in \mathcal{M}$, and outputs a ciphertext $c \in \mathcal{C}$; (iii) The deterministic algorithm ADec takes as input a key $\kappa \in \mathcal{K}$ and a ciphertext $c \in \{0,1\}^*$, and outputs a value $m \in \mathcal{M} \cup \{\perp\}$, where $\perp$ denotes an invalid ciphertext. We call $\mathcal{K}$, $\mathcal{M}$, $\mathcal{C}$, respectively, the key, message, and ciphertext space of Π.[6]

We say that Π meets correctness if for all $\kappa \in \mathcal{K}$, and all messages $m \in \mathcal{M}$, we have that $\mathbb{P}\left[\mathsf{ADec}(\kappa,\mathsf{AEnc}(\kappa,m)) = m\right] = 1$ (where the probability is taken over the randomness of AEnc). As for security, we will need AE schemes that satisfy two properties (see below for formal definitions). The first property, usually known as *semantic security*, says that it is hard to distinguish the encryption of any two (adversarially chosen) messages. The second property, usually called *authenticity*, says that, without knowing the secret key, it is hard to produce a valid ciphertext (i.e., a ciphertext that does not decrypt to $\perp$).

Definition 2 (Security of AE). *Let $\Pi = (\mathsf{KGen},\mathsf{AEnc},\mathsf{ADec})$ be an AE scheme. We say that Π is secure if the following holds for the games defined in Fig. 1.*

$$\forall\ PPT\ \mathsf{A}:\ \left\{\mathbf{G}^{\mathsf{sem}}_{\Pi,\mathsf{A}}(\lambda,0)\right\}_{\lambda\in\mathbb{N}} \approx_c \left\{\mathbf{G}^{\mathsf{sem}}_{\Pi,\mathsf{A}}(\lambda,1)\right\}_{\lambda\in\mathbb{N}},$$
$$\mathbb{P}\left[\mathbf{G}^{\mathsf{auth}}_{\Pi,\mathsf{A}}(\lambda)=1\right] \in \mathtt{negl}(\lambda).$$

[6] These sets typically depend on the security parameter, but we drop this dependency to simplify notation.

Note that since both authenticity and semantic security are one-time guarantees, in principle, information-theoretic constructions with such properties are possible when $|\mathcal{K}| \geq |\mathcal{M}|$. However, we are interested in constructions where $|\mathcal{M}| \gg |\mathcal{K}|$, for which the existence of one-way functions is necessary.

3.2 Non-Malleable Codes

A split-state code $\Gamma = (\mathsf{Enc}, \mathsf{Dec})$ consists of a pair of polynomial-time algorithms specified as follows: (i) The randomized encoding algorithm Enc takes as input a message $m \in \mathcal{M}$ and returns a split-state codeword $(L, R) \in \mathcal{L} \times \mathcal{R}$; (ii) The (deterministic) decoding algorithm Dec takes as input a codeword $(L, R) \in (\{0,1\}^*)^2$ and outputs a value in $\mathcal{M} \cup \{\perp\}$, where $\perp$ denotes an *invalid* codeword. A codeword (L, R) such that $\mathsf{Dec}(L, R) \neq \perp$ is called a *valid* codeword; we call $\mathcal{M}$ the message space, and $\mathcal{L}, \mathcal{R}$ the left and right codeword space.

We say that Γ satisfies *correctness* if, for all $m \in \mathcal{M}$, we have that $\mathsf{Dec}(\mathsf{Enc}(m)) = m$ with overwhelming probability over the randomness of the encoding.

Noisy leakage. We will leverage codes where non-malleability (as defined below) is satisfied even in the presence of adversaries that can obtain *independent leakage* on the two shares of a target encoding (L, R).

Following a long tradition in leakage-resilient cryptography [29,52,35], we model the leakage as an arbitrary function of its input. The only restriction is that the overall leakage on L does not decrease the min-entropy of L more than a fixed amount $\ell \in \mathbb{N}$ (that is a parameter of the scheme). Of course, an analogous condition must be satisfied for the leakage on the right side R. We formalize this restriction via a notion of *admissibility*, as defined below.

Definition 3 (Admissible adversaries for split-state codes). *Let $\Gamma = (\mathsf{Enc}, \mathsf{Dec})$ be a split-state code. We say that a PPT adversary A is ℓ-admissible if it outputs a sequences of leakage queries (chosen adaptively) $(g^{(q)}_{\mathsf{left}}, g^{(q)}_{\mathsf{right}})_{q \in [p]}$, with $p(\lambda) \in \mathtt{poly}(\lambda)$, such that for all messages $m \in \mathcal{M}$:*

$$\widetilde{\mathbb{H}}_\infty \left(\mathbf{L} | \mathbf{R}, g^{(1)}_{\mathsf{left}}(\mathbf{L}), \cdots, g^{(p)}_{\mathsf{left}}(\mathbf{L}) \right) \geq \widetilde{\mathbb{H}}_\infty(\mathbf{L}|\mathbf{R}) - \ell$$

$$\widetilde{\mathbb{H}}_\infty \left(\mathbf{R} | \mathbf{L}, g^{(1)}_{\mathsf{right}}(\mathbf{R}), \cdots, g^{(p)}_{\mathsf{right}}(\mathbf{R}) \right) \geq \widetilde{\mathbb{H}}_\infty(\mathbf{R}|\mathbf{L}) - \ell,$$

where $(\mathbf{L}, \mathbf{R})$ is the joint random variable corresponding to $\mathsf{Enc}(1^\lambda, m)$.

Note that we measure the min-entropy drop due to the leakage w.r.t. the conditional average min-entropy of $L|R$ and $R|L$. We find this meaningful as it allows to capture automatically the correlation between L and R. Alternatively, we could define admissibility by conditioning only on the leakage (without further considering the other share in the equations above); we observe, however, that these two notions of admissibility are equivalent up to a small loss in the leakage parameter. This is due to the fact that, in known instantiations [7,50], the mutual information between L and R is small, a property sometimes known as conditional independence [53,34,25].

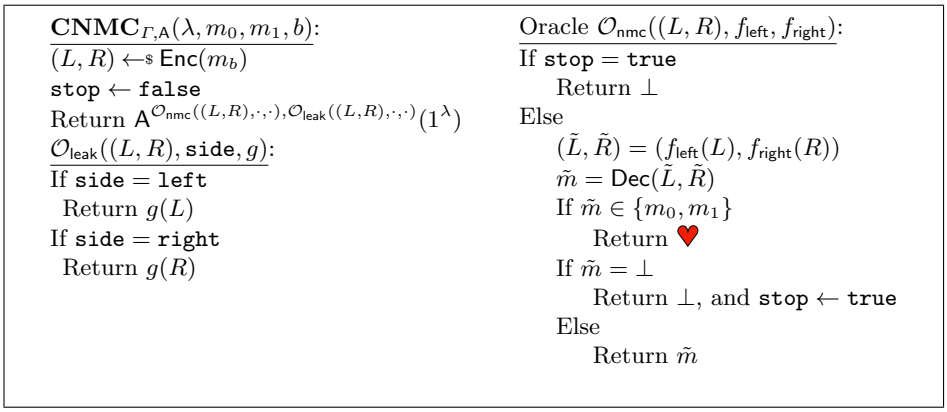

Fig. 2: Experiment defining continuously non-malleable codes in the split-state model. The tampering oracle $\mathcal{O}_{\mathsf{nmc}}$ is implicitly parameterized by the flag $\mathtt{stop}$.

Continuous non-malleability. Intuitively, a split-state code is non-malleable [33,51] if no adversary tampering *independently* (yet arbitrarily) with the two sides of a given target encoding (L, R) of some value m, can generate a modified codeword $(\tilde{L}, \tilde{R})$ that decodes to a value related to m. Continuous non-malleability [37] is a strengthening of this guarantee, where the attacker is allowed to tamper continuously, and adaptively, with (L, R), until a decoding error occurs, after which the system "self-destructs" and stops answering tampering queries. Such a self-destruct capability, that in practice might be implemented via a public write-once flag, is well known to be necessary for achieving continuous non-malleability, as otherwise simple attacks are possible [39].

We formalize continuous non-malleability for split-state non-malleable codes using a game-based definition. Simulation-based definitions also exist, but the two formulations are known to be equivalent as long as the messages to be encoded have super-logarithmic length in the security parameter [33,53]. In order to model (split-state) tampering attacks, we use a stateless leakage oracle $\mathcal{O}_{\mathsf{leak}}$ and a stateful oracle $\mathcal{O}_{\mathsf{nmc}}$ that are initialized with a target encoding (L, R) of either of two messages $m_0, m_1 \in \mathcal{M}$. The goal of the attacker is to distinguish which message was encoded, while performing both leakage and tampering attacks: The leakage oracle allows the adversary to obtain information from L and R, while the tampering oracle allows the adversary to tamper with L and R independently. In case the decoded message corresponding to a modified codeword $(\tilde{L}, \tilde{R})$ is equal to one of the original messages m_0, m_1, the oracle returns a special symbol 💔, as otherwise it would be trivial to distinguish which message was encoded by querying the oracle with, e.g., the identity function.

Definition 4 (Split-state continuously non-malleable codes). *Let $\Gamma = (\mathsf{Enc}, \mathsf{Dec})$ be a split-state code. We say that Γ is an ℓ-noisy leakage-resilient split-state* continuously non-malleable *code (ℓ-LR-CNMC for short) if for all $m_0, m_1 \in \mathcal{M}$ and for all PPT ℓ-admissible adversaries A as per Def. 3, we have*

that

$$\{\mathbf{CNMC}_{\Gamma,\mathsf{A}}(\lambda, m_0, m_1, 0)\}_{\lambda\in\mathbb{N}} \approx_c \{\mathbf{CNMC}_{\Gamma,\mathsf{A}}(\lambda, m_0, m_1, 1)\}_{\lambda\in\mathbb{N}}, \quad (2)$$

where, for $b \in \{0,1\}$*, experiment* $\mathbf{CNMC}_{\Gamma,\mathsf{A}}(\lambda, m_0, m_1, b)$ *is depicted in Fig. 2.*

Message uniqueness. An important property that must be satisfied by any split-state continuously non-malleable code is that of *message uniqueness* (MU) [37,53]. Informally, this means that if we fix the left side L of an encoding, there are no[7] two right sides R_1, R_2, such that both (L, R_1) and (L, R_2) are *valid* codewords that decode to *different* messages $m_1 \neq m_2$. (An analogous guarantee must hold if we fix the right side.)

A simple observation, due to [53], is that both the left side L and the right side R of a split-state non-malleable encoding constitute a perfectly binding commitment to the message.

Lemma 1 ([53]). *Let* Γ *be a split-state code satisfying MU. Then, for any string* $L \in \{0,1\}^*$ *(resp.* $R \in \{0,1\}^*$*), there exists at most a single value* $m \in \mathcal{M}$ *such that* $\mathsf{Dec}(L, R) = m \neq \perp$ *for some* $R \in \{0,1\}^*$ *(resp. for some* $L \in \{0,1\}^*$*).*

Additional properties. For our main construction, we will need CNMCs satisfying two additional properties as defined below. The first property, called symmetric decoding, says that for all possible inputs L, R, decoding (L, R) yields the same as decoding (R, L). Note that this implies some (very weak) form of resilience against tampering via permutations, in that any split-state continuously non-malleable code with symmetric decoding is still secure w.r.t. attackers that first tamper the two states (L, R) independently, and later swap L and R.

Definition 5 (Symmetric decoding). *We say that a split-state code* $\Gamma = (\mathsf{Enc}, \mathsf{Dec})$ *has symmetric decoding if for all* $L, R \in (\{0,1\}^*)^2$*, we have that* $\mathsf{Dec}(L, R) = \mathsf{Dec}(R, L)$*.*

The second property, called codewords uniformity, requires that, for any message, the encoder outputs codewords that are uniform over the set of all possible encodings of the message.

Definition 6 (Codewords uniformity). *We say that a split-state code* $\Gamma = (\mathsf{Enc}, \mathsf{Dec})$ *has codewords uniformity if for all* $m \in \mathcal{M}$*, we have that* $\mathsf{Enc}(1^\lambda, m)$ *is distributed uniformly over the set of all possible pairs* (L, R) *s.t.* $\mathsf{Dec}(L, R) = m$*.*

4 Continuously Non-Malleable Secret Sharing

4.1 Non-Malleability under Adaptive Concurrent Reconstruction

We now give the definition of leakage-resilient continuously non-malleable secret sharing (LR-CNMSS) under adaptive concurrent reconstruction. We focus on

[7] Observe that "perfect" MU, as opposed to "computational" MU is wlog. in the plain model.

the case of threshold secret sharing, where the adversary is allowed to tamper (possibly all!) the shares arbitrarily, but independently. Non-malleability intuitively guarantees that the reconstructed message, where the indices $\mathcal{T}$ (with $|\mathcal{T}| = \tilde{\varrho} \geq \varrho$) used for reconstruction are also chosen by the adversary, is independent of the original message.

Importantly, in our model, the adversary is allowed to tamper continuously, and adaptively, with the same target secret sharing; the set used for reconstruction in each tampering attempt is also adversarial, and moreover can be chosen adaptively based on the outcome of previous queries. This feature, known as *concurrent reconstruction*, was already considered in previous work [2], although in a non-adaptive setting. There are only two limitations: (i) The adversary is computationally bounded; (ii) After the first tampering query yielding a mauled secret sharing that reconstructs to $\perp$, the answer to all future tampering queries will be $\perp$ by default. The second limitation is sometimes known as "self-destruct feature" in the literature of non-malleable codes [37]. Both of these limitations are somewhat *necessary* (see below).

In order to make our model even stronger, we further allow the adversary to leak information independently from all the shares. The only restriction here is that the leakage does not decrease the amount of uncertainty contained in each of the shares by too much. This leads to the notion of admissible adversary, which is similar in spirit to the notion of admissible adversaries for codes (cf. §3.2), as defined below.

Definition 7 (Admissible adversaries for secret sharing). *Let $\Sigma = (\mathsf{Share}, \mathsf{Rec})$ be an n-party secret sharing scheme. We say that a PPT adversary A is ℓ-admissible if it outputs a sequence of leakage queries (chosen adaptively) $(i, g_i^{(q)})_{i\in[n],q\in[p]}$, with $p(\lambda) \in \mathtt{poly}(\lambda)$, such that for all $i \in [n]$, and for all $m \in \mathcal{M}$:*

$$\widetilde{\mathbb{H}}_\infty\left(\mathbf{S}_i|(\mathbf{S}_j)_{j\neq i}, g_i^{(1)}(\mathbf{S}_i), \cdots, g_i^{(p)}(\mathbf{S}_i)\right) \geq \widetilde{\mathbb{H}}_\infty(\mathbf{S}_i|(\mathbf{S}_j)_{j\neq i}) - \ell,$$

where $(\mathbf{S}_1, \ldots, \mathbf{S}_n)$ is the random variable corresponding to $\mathsf{Share}(1^\lambda, m)$.

Definition 8 (Continuously non-malleable threshold secret sharing). *Let $n, \tau, \varrho, \ell \in \mathbb{N}$. Let $\Sigma = (\mathsf{Share}, \mathsf{Rec})$ be an n-party secret sharing over message space $\mathcal{M}$ and share space $\mathcal{S} = \mathcal{S}_1 \times \cdots \times \mathcal{S}_n$. We say that Σ is an ℓ-noisy leakage-resilient* continuously non-malleable *(n, τ, ϱ)-threshold secret sharing scheme under* adaptive concurrent reconstruction *((n, τ, ϱ, ℓ)-LR-CNMSS for short) if it is an (n, τ, ϱ)-TSS as per Def. 1, and additionally for all pairs of messages $m_0, m_1 \in \mathcal{M}$, and all PPT ℓ-admissible adversaries A as per Def. 7, we have:*

$$\{\mathbf{CNMSS}_{\Sigma,\mathsf{A}}(\lambda, m_0, m_1, 0)\}_{\lambda\in\mathbb{N}} \approx_c \{\mathbf{CNMSS}_{\Sigma,\mathsf{A}}(\lambda, m_0, m_1, 1)\}_{\lambda\in\mathbb{N}},$$

where, for $b \in \{0, 1\}$, experiment $\mathbf{CNMSS}_{\Sigma,\mathsf{A}}(\lambda, m_0, m_1, b)$ is depicted in Fig. 3.

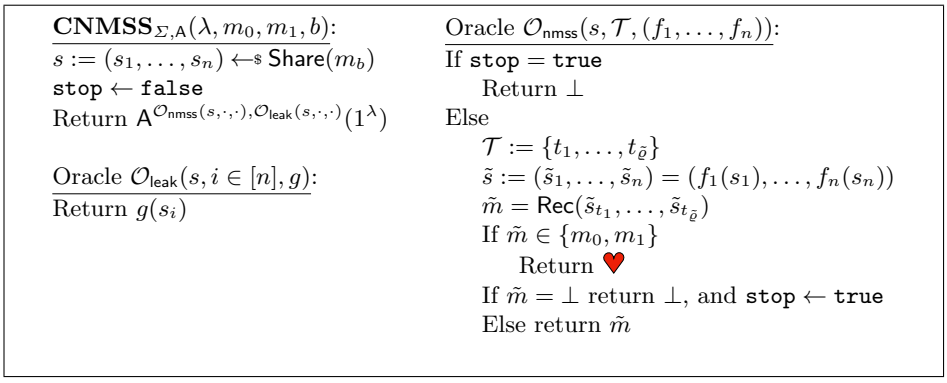

Fig. 3: Experiment defining leakage-resilient continuously non-malleable secret sharing against individual tampering, under adaptive concurrent reconstruction. Note that the oracle $\mathcal{O}_{\mathsf{nmss}}$ is implicitly parameterized by the flag `stop`.

Remark 1 (On game-based security). Note that Def. 8 is game based in spirit. This is in contrast with all previous definitions of non-malleable secret sharing, which instead are simulation based. While, one could also formulate a simulation-based definition for LR-CNMSS, it is not hard to show that the two formulations are equivalent as long as the shared value has super-logarithmic length in the security parameter. A similar equivalence holds for the case of (continuously) non-malleable codes [33,53].

Remark 2 (On the relation with CNMCs). When $\ell = 0$, $n = 2$, and $\tau = \varrho = 2$, one obtains the definition of split-state CNMCs as a special case. In fact, similar to [7], one can show that any split-state CNMC satisfies 2-threshold privacy.

In the following subsections, we show that both limitations of computational security and self-destruct are somewhat inherent in our model (even when no leakage is allowed, i.e. $\ell = 0$). This is immediate for the case $n = 2 = \tau = \varrho = 2$, as the same limitations hold for the case of split-state CNMCs [37]. The theorems below[8] generalize the impossibility results of [37] for certain values of n, τ, ϱ.

4.2 Shared-Value Uniqueness

Consider the following natural generalization of the MU property for continuously non-malleable codes (cf. §3.2) to the case of TSS schemes.[9]

Definition 9 (Shared-value uniqueness). *Let $\Sigma = (\mathsf{Share}, \mathsf{Rec})$ be an n-party TSS with reconstruction threshold $\varrho \leq n$. We say that Σ satisfies shared-value uniqueness (SVU) if for all subsets $\mathcal{I} = \{i_1, \ldots, i_\varrho\} \subseteq [n]$, there exists $j^* \in$*

[8] We stress that the attacks described in the proof of Thm. 2 and Thm. 3 do not require to change the reconstruction set $\mathcal{T}$ among different queries, and thus even hold without considering concurrent reconstruction.

[9] As for MU, "perfect" SVU, rather than "computational" SVU, is wlog. in the plain model.

$[\varrho]$ such that for all shares $s_{i_1}, \ldots, s_{i_{j^*-1}}, s_{i_{j^*+1}}, \ldots, s_{i_\varrho}$, and for all $s_{i_{j^*}}, s'_{i_{j^*}}$, we have that either

$$m = \mathsf{Rec}(s_{i_1}, \ldots, s_{i_{j^*}}, \ldots, s_{i_\varrho}) = \mathsf{Rec}(s_{i_1}, \ldots, s'_{i_{j^*}}, \ldots, s_{i_\varrho}) = m', \qquad (3)$$

where $m, m' \in \mathcal{M}$, or at least one of m, m' equals $\perp$.

Intuitively, the above property says that for every possible choice of an authorized set $\mathcal{I}$, there exists at least one index $i_{j^*} \in \mathcal{I}$, such that if we fix arbitrarily all the shares but the one in position i_{j^*}, the reconstruction process can possibly output a single outcome within the space of all valid messages. The theorem below (whose proof appears in the full version [36]) says that SVU is necessary for achieving continuous non-malleability (without leakage) for threshold secret sharing, in the computational setting.

Theorem 2. *For any $n, \tau, \varrho \in \mathbb{N}$, with $\tau \leq \varrho \leq n$, every $(n, \tau, \varrho, 0)$-LR-CNMSS must also satisfy SVU.*

Notice that in the information-theoretic setting, when the privacy threshold τ equals the reconstruction threshold ϱ, and when considering the authorized set $\mathcal{I} = [\varrho]$, statistical privacy implies that for each $i^* \in [\varrho]$ there always exist shares $(s_1, \ldots, s_{i^*-1}, s_{i^*}, s_{i^*+1}, \ldots, s_\varrho)$ and $(s_1, \ldots, s_{i^*-1}, s'_{i^*}, s_{i^*+1}, \ldots, s_\varrho)$ that violate SVU. Hence, CNMSS with the optimal parameters $\tau = \varrho$ is impossible in the information-theoretic setting, a fact recently established in [10].

Corollary 1 ([10]). *For any $n, \tau, \varrho \in \mathbb{N}$, with $\tau = \varrho \leq n$, there is no $(n, \tau, \varrho, 0)$-LR-CNMSS in the information-theoretic setting.*

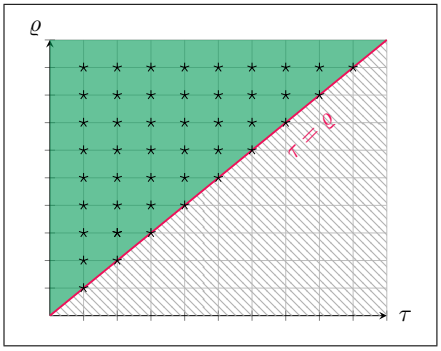

Fig. 4: Possible parameters ϱ, τ of CNMSS. Values on the red line require computational assumptions.

Mind the gap. What if there is a small gap between the reconstruction threshold ϱ and the privacy threshold τ (e.g., $\tau \leq \varrho - 1$)? In this case, the above impossibility result does not apply. For concreteness, let Σ be an $(n, \varrho - 1, \varrho)$-TSS and consider the reconstruction set $\mathcal{I} = [\varrho]$. By perfect privacy, since any collection of $\varrho - 2$ shares reveals no information on the shared value, for every sequence of shares $s_1, \ldots, s_{\varrho-2}$, and for every message $\hat{m} \in \mathcal{M}$, there exist at least two shares $\hat{s}_{\varrho-1}, \hat{s}_\varrho$ such that running the reconstruction algorithm upon $(s_1, \ldots, s_{\varrho-2}, \hat{s}_{\varrho-1}, \hat{s}_\varrho)$ yields $\hat{m}$ as output. However, there is no guarantee that a pair of shares $(\hat{s}'_{\varrho-1}, \hat{s}'_\varrho)$ yielding another message $\hat{m}' \neq \hat{m}$, and such that, e.g., $\hat{s}'_{\varrho-1} = \hat{s}_{\varrho-1}$, actually exists. This circumvents the attack described above. Put differently, whenever $\tau = \varrho - 1$, given any collection of $\varrho - 1$ shares, we can consider two cases (cf. also Fig. 4):

– There are at least two possible *valid* outcomes for the reconstruction procedure. In this case, a computationally unbounded attacker can still find a sequence of shares violating SVU, and thus continuous non-malleability requires computational assumptions.
– The shared value is information-theoretically determined, i.e. there exists an inefficient algorithm which can reconstruct the message. In this case, SVU is not violated, and thus it is plausible that TSS with perfect privacy and statistical continuous non-malleability exists.

4.3 Necessity of Self-Destruct

Finally, in the full version [36], we show that continuous non-malleability as per Def. 8 is impossible without assuming self-destruct. This fact is reminiscent of a similar impossibility result for continuously non-malleable codes [37], and tamper-resilient cryptography [39].

Theorem 3. *For any $n, \tau, \varrho \in \mathbb{N}$, with $\tau \leq \varrho \leq n$, there is no $(n, \tau, \varrho, 0)$-LR-CNMSS without assuming the self-destruct capability.*

5 A Scheme with Poor Rate

Before describing our scheme, we introduce some useful notation. The shares will be of the form $s_i^* = (s_{\mathcal{H},i})_{\mathcal{H} \in \binom{[n]}{2}}$ (see Fig. 5), where $i \in [n]$. Given a set $\mathcal{A} \subseteq [n]$, we identify with $\hat{\mathcal{A}}$ the first two indices (according to the natural order) of $\mathcal{A}$.

Our threshold secret sharing $\Sigma^* = (\mathsf{Share}^*, \mathsf{Rec}^*)$, which is formally depicted in Fig. 5, is based upon the following ingredients:

– An authenticated secret-key encryption (AE) scheme $\Pi = (\mathsf{AEnc}, \mathsf{ADec})$ (cf. §3.1), with message space $\mathcal{M}$, ciphertext space $\mathcal{C}$, and key space $\mathcal{K} = \{0,1\}^\lambda$.
– An $(n-2)$-party secret sharing scheme $\Sigma = (\mathsf{Share}, \mathsf{Rec})$, with reconstruction threshold equal to $\varrho - 3$, message space $\mathcal{C}$, and share space $\mathcal{S}^{n-2}$ (cf. §3.1).
– A split-state encoding $\Gamma = (\mathsf{Enc}, \mathsf{Dec})$, with message space $\mathcal{K}$ and codeword space $\mathcal{L} \times \mathcal{R}$ (cf. §3.2).

The main intuition behind the construction has been already discussed in §2. The formal proof of the theorem below can be found in the full version [36].

Theorem 4. *Let $n, \varrho, \ell, \ell^* \in \mathbb{N}$ be such that $n \geq \varrho > 2$. Assuming that Π is a secure AE scheme, that Σ is a $(n-2, \varrho-3, \varrho-3)$-TSS with perfect threshold privacy and with the special reconstruction property, and that Γ is an ℓ-LR-CNMC with symmetric decoding and with codewords uniformity, the secret sharing scheme Σ^* of Fig. 3 is an $(n, \varrho-1, \varrho, \ell^*)$-LR-CNMSS, as long as $\ell = \ell^* + 2\gamma + O(\log \lambda)$ where $\gamma = \log |\mathcal{C}|$ is the size of a ciphertext under Π.*

Basic Construction of LR-CNMSS

Let $\Pi = (\mathsf{AEnc}, \mathsf{ADec})$, $\Sigma = (\mathsf{Share}, \mathsf{Rec})$, and $\Gamma = (\mathsf{Enc}, \mathsf{Dec})$ be as described in the text. Consider the following construction of an n-party secret sharing $\Sigma^* = (\mathsf{Share}^*, \mathsf{Rec}^*)$ with reconstruction threshold $\varrho \leq n$, and message space $\mathcal{M}^* = \mathcal{M}$.

Sharing function $\mathsf{Share}^*(m)$**:** The secret sharing of a message $m \in \mathcal{M}^*$ is a collection of shares $s^* = (s_1^*, \ldots, s_n^*)$, where $s_i^* = (s_{\mathcal{H},i})_{\mathcal{H} \in \binom{[n]}{2}}$ and for any $\mathcal{H} = \{h_1, h_2\} \in \binom{[n]}{2}$ the share $s_{\mathcal{H},i}$ is computed following the steps below:

1. Let $\bar{\mathcal{H}} = [n] \setminus \mathcal{H} = \{h_3, \ldots, h_n\}$;
2. Sample $\kappa_{\mathcal{H}} \leftarrow\!\!\$\ \mathcal{K}$ and run $c_{\mathcal{H}} \leftarrow\!\!\$\ \mathsf{AEnc}(\kappa_{\mathcal{H}}, m)$;
3. Compute $(s_{\mathcal{H},h_1}, s_{\mathcal{H},h_2}) \leftarrow\!\!\$\ \mathsf{Enc}(\kappa_{\mathcal{H}})$ and $(s_{\mathcal{H},h_3}, \ldots, s_{\mathcal{H},h_n}) \leftarrow\!\!\$\ \mathsf{Share}(c_{\mathcal{H}})$.

Reconstruction function $\mathsf{Rec}^*(s_{\mathcal{I}}^*)$: Let $\mathcal{I} = \{i_1, \ldots, i_\varrho\}$. Wlog. we assume that the set $\mathcal{I}$ is ordered and that is made of exactly ϱ indices. (If not, we can just order it and use only the first ϱ indices.)

1. Let $\hat{\mathcal{I}} = \{i_1, i_2\}$, and parse $s_{\mathcal{I}}^* = (s_{i_1}^*, \ldots, s_{i_\varrho}^*)$, where for each $j \in [\varrho]$ we have $s_{i_j}^* = (s_{\mathcal{H},i_j})_{\mathcal{H} \in \binom{[n]}{2}}$;
2. Compute $\kappa = \mathsf{Dec}(s_{\hat{\mathcal{I}},i_1}, s_{\hat{\mathcal{I}},i_2})$, and for sets $\mathcal{A}_1 = \{i_3, \ldots, i_{\varrho-1}\}$ and $\mathcal{A}_2 = \{i_4, \ldots, i_\varrho\}$ let $c_1 = \mathsf{Rec}((s_{\hat{\mathcal{I}},a})_{a \in \mathcal{A}_1})$ and $c_2 = \mathsf{Rec}((s_{\hat{\mathcal{I}},a})_{a \in \mathcal{A}_2})$;
3. If $c_1 \neq c_2$ output $\perp$, else let $c = c_1 = c_2$ and return $m = \mathsf{ADec}(\kappa, c)$.

Fig. 5: A construction of leakage-resilient continuously non-malleable secret sharing for threshold access structures, in the computational setting.

Instantiating the construction. In the full version of this paper [36], we show how to instantiate Thm. 4, under the assumption of 1-to-1 OWFs. It is well-known that authenticated encryption can be constructed in a black-box way from any OWF, whereas we can use the classical Shamir's construction [56] for the underlying TSS scheme. The latter is easily seen to meet the special reconstruction property.

It remains to exhibit a split-state CNMC with the required properties, which we do by revisiting the construction (and security analysis) of [53].

6 Boosting the Rate

6.1 Information Rate of Secret Sharing

An important measure of the efficiency of a secret sharing scheme is its information rate, defined as the ratio between the size of the message and the maximum size of a share as function of the size of the message and the number of shares.[10]

[10] One can also define a more general notion of information rate for secret sharing schemes [15], which depends on the entropy of the distribution $\mathbf{M}$ of the input message. The above definition is obtained as a special case, by considering the uniform distribution.

Rate-Optimizing Compiler for LR-CNMSS

Let $\Sigma' = (\mathsf{Share}', \mathsf{Rec}')$ be an n-party TSS over message space $\mathcal{M}' := \mathcal{K}$ and share space $\mathcal{S}'^n$. Let $\Pi = (\mathsf{AEnc}, \mathsf{ADec})$ be an authenticated secret-key encryption scheme with key space $\mathcal{K}$, message space $\mathcal{M}$, and ciphertext space $\mathcal{C}$. Consider the following construction of a derived n-party TSS over message space $\mathcal{M}$ and share space $\mathcal{S} := (\mathcal{S}' \times \mathcal{C})^n$.

Sharing function $\mathsf{Share}(m)$**:** Sample $\kappa \leftarrow\!\!\$\ \mathcal{K}$, and compute $c \leftarrow\!\!\$\ \mathsf{AEnc}(\kappa, m)$. Let $(\kappa_1, \ldots, \kappa_n) \leftarrow\!\!\$\ \mathsf{Share}'(\kappa)$. Output $s = (s_1, \ldots, s_n)$, where $s_i := (\kappa_i, c)$ for all $i \in [n]$.

Reconstruction function $\mathsf{Rec}(s_{\mathcal{I}})$**:** Parse $s_{\mathcal{I}} = (s_{i_1}, \ldots, s_{i_\varrho})$, where $s_{i_j} = (\kappa_{i_j}, c_{i_j})$ for all $j \in [\varrho]$. Let $\kappa = \mathsf{Rec}'(\kappa_{i_1}, \ldots, \kappa_{i_\varrho})$; if $\kappa = \perp$, return $\perp$. Else, if $c_{i_1} = \ldots = c_{i_\varrho} := c$, output $\mathsf{ADec}(\kappa, c)$, and otherwise output $\perp$.

Fig. 6: Boosting the rate of any leakage-resilient continuously non-malleable secret sharing (in the computational setting).

Definition 10 (Rate of secret sharing). *Let $\Sigma = (\mathsf{Share}, \mathsf{Rec})$ be an n-party secret sharing over message space $\mathcal{M}$ and share space $\mathcal{S} = \mathcal{S}_1 \times \cdots \times \mathcal{S}_n$. We define the information rate of Σ to be the ratio*

$$\eth(\mu, n, \lambda) := \min_{i \in [n]} \frac{\mu}{\sigma_i(\mu, n, \lambda)}$$

where $\mu = \log |\mathcal{M}|$ and $\sigma_i(\mu, n, \lambda) = \log |\mathcal{S}_i|$ denote, respectively, the bit-length of the message and of the i-th share under Σ. Moreover, we say that Σ has asymptotic rate 0 (resp. 1) if $\inf_{\lambda \in \mathbb{N}} \lim_{\mu \to \infty} \eth(\mu, n, \lambda)$ is 0 (resp. 1).

In the full version [36], we show an instantiation of the TSS scheme from §5 with shares of length $O(n^2 \cdot \max\{\lambda^8, \mu + \lambda\})$. Hence, we have obtained:

Corollary 2. *Let $\lambda \in \mathbb{N}$ be the security parameter. Under the assumption of 1-to-1 OWFs, there exists a noisy-leakage-resilient continuously non-malleable n-party threshold secret sharing for μ-bit messages, with rate $\Omega\left(\frac{\mu}{n^2 \cdot (\lambda^8 + \mu)}\right)$.*

6.2 A Rate-Optimizing Compiler

In this section, we show how to optimize the rate of any LR-CNMSS, under computational assumptions. We will achieve this through a so-called rate compiler, i.e. a black-box transformation that takes any LR-CNMSS with asymptotic rate 0 and returns a LR-CNMSS with asymptotic rate 1.

Our compiler is formally described in Fig. 6, and is inspired by a beautiful idea of Aggarwal *et al.* [1], who considered a similar question for the case of (one-time) non-malleable codes against split-state tampering; recently, their approach was also analyzed in the case of continuous tampering [25]. Intuitively, the construction works as follows. The sharing function samples a uniformly random key κ for a symmetric encryption scheme, and secret shares κ using the underlying rate-0 threshold secret sharing, obtaining shares $\kappa_1, \ldots, \kappa_n$. Next, the input

message m is encrypted under the key κ, yielding a ciphertext c, and the final share of each player is defined to be $s_i = (\kappa_i, c)$. Importantly, the reconstruction function, before obtaining the key κ and decrypting the ciphertext c, checks that the ciphertext contained in every given share is the same.

Note that when the initial secret sharing scheme is a 2-out-of-2 TSS, i.e. Σ' is actually a split-state LR-CNMC, we obtain as a special case one of the rate compilers analyzed in [25]. A notable advantage of our result, however, is that we can instantiate the construction in the plain model (whereas Coretti *et al.* assume a CRS). In the full version [36], we establish the following result.

Theorem 5. *Let $n, \tau, \varrho \in \mathbb{N}$, with $\tau \leq \varrho \leq n$. Assuming that Σ' is an $(n, \tau, \varrho, \ell')$-LR-CNMSS, and that Π is a secure AE scheme, the secret sharing scheme Σ of Fig. 6 is an (n, τ, ϱ, ℓ)-LR-CNMSS as long as $\ell' = \ell + \lambda + O(\log \lambda)$.*

Note that since the key size is independent of the message size, the length of a share is $\mu + \mathtt{poly}(n, \lambda)$, thus yielding a rate of $\frac{\mu}{\mu + \mathtt{poly}(n,\lambda)}$. This asymptotically approaches 1 when the message size goes to infinity.

Corollary 3. *Under the assumption of 1-to-1 OWFs, there exists a noisy-leakage-resilient continuously non-malleable threshold secret sharing with asymptotic information rate 1.*

7 Threshold Signatures under Adaptive Memory Corruptions

7.1 Syntax

An n-party threshold signature is a tuple $\Pi = (\mathsf{KGen}, \Xi, \mathsf{Vrfy})$ specified as follows. (i) The PPT algorithm KGen takes as input the security parameter, and outputs a verification key $vk \in \mathcal{VK}$, and n secret keys $sk_1, \ldots, sk_n \in \mathcal{SK}$; (ii) $\Xi = (\mathsf{P}_1, \ldots, \mathsf{P}_n)$ specifies a set of protocols which can be run by a subset $\mathcal{I}$ of n interactive PPT Turing machines $\mathsf{P}_1, \ldots, \mathsf{P}_n$, where each P_i takes as input a message $m \in \mathcal{M}$ and secret key sk_i, and where we denote by $(\sigma, \xi) \xleftarrow{\Xi}{}_{\$} \langle \mathsf{P}_i(sk_i, m) \rangle_{i \in \mathcal{I}}$ a run of Ξ by the parties $(\mathsf{P}_i)_{i \in \mathcal{I}}$, yielding a signature σ and transcript ξ. (iii) The deterministic polynomial-time algorithm Vrfy takes as input the verification key vk, and a pair (m, σ), and returns a bit.

For a parameter $\varrho \leq n$, we say that an n-party threshold signature is ϱ-correct if for all $\lambda \in \mathbb{N}$, all $(vk, sk_1, \ldots, sk_n)$ output by $\mathsf{KGen}(1^\lambda)$, all messages $m \in \mathcal{M}$, and all subsets $\mathcal{I}$ such that $|\mathcal{I}| \geq \varrho$, the following holds:

$$\mathbb{P}\left[\mathsf{Vrfy}(vk, (m, \sigma)) = 1 : \ (\sigma, \xi) \xleftarrow{\Xi}{}_{\$} \langle \mathsf{P}_i(sk_i, m) \rangle_{i \in \mathcal{I}}\right] = 1.$$

We also consider *non-interactive* threshold signature schemes. Such schemes are fully specified by a tuple of polynomial-time algorithms $(\mathsf{KGen}, \mathsf{TSign}, \mathsf{Combine}, \mathsf{Vrfy})$, such that $\mathsf{KGen}, \mathsf{Vrfy}$ are as in the interactive case, whereas the protocol Ξ, run by a subset $\mathcal{I}$ of the parties, has the following simple structure:

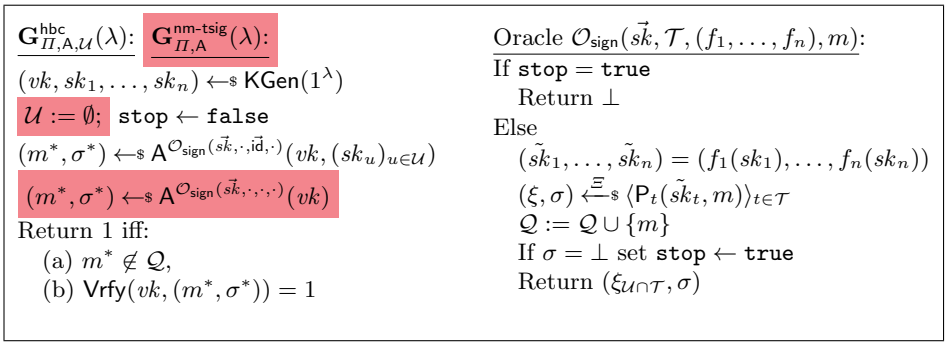

Fig. 7: Experiments defining privacy and continuous non-malleability for threshold signatures. The vector $\vec{\mathsf{id}}$ contains the identity function (repeated n times).

- For each $i \in \mathcal{I}$, party P_i computes locally $\sigma_i \leftarrow\!\!\$\ \mathsf{TSign}(sk_i, m)$ and broadcasts the resulting signature share σ_i;
- For each $i \in \mathcal{I}$, party P_i locally computes $\sigma \leftarrow\!\!\$\ \mathsf{Combine}(vk, (\sigma_i)_{i\in\mathcal{I}})$; most notably, algorithm $\mathsf{Combine}$ only uses public information.

7.2 Security Model

We assume authenticated and private channels between each pair of parties. The standard security notion for threshold signatures deals with an adversary A statically corrupting a subset $\mathcal{U}$ of the players, with size below the reconstruction threshold of the scheme. The guarantee is that the attacker should not be able to forge a valid signature on a fresh message, even after seeing a polynomial number of executions of the signature protocol on several messages and involving different subsets of the players; note that, for each such subset $\mathcal{I}$, the attacker learns the transcript of the signature protocol relative to the players in $\mathcal{U} \cap \mathcal{I}$. Below, we formalize this guarantee in the honest-but-curious case.

Definition 11 (Privacy for threshold signatures). *Let $\Pi = (\mathsf{KGen}, \Xi, \mathsf{Vrfy})$ be an n-party threshold signature scheme. We say that Π is τ-private against honest-but-curious adversaries if for all PPT attackers A, and all subsets $\mathcal{U} \subset [n]$ such that $|\mathcal{U}| < \tau$:*

$$\mathbb{P}\left[\mathbf{G}^{\mathsf{hbc}}_{\Pi,\mathsf{A},\mathcal{U}}(\lambda) = 1\right] \in \mathtt{negl}(\lambda).$$

where the game $\mathbf{G}^{\mathsf{hbc}}_{\Pi,\mathsf{A},\mathcal{U}}(\lambda)$ is described in Fig. 7.

Non-malleability. Next, we consider an adversary able to corrupt the memory of each party independently. The security guarantee is still that of existential unforgeability, except that the attacker can now see a polynomial number of executions of the signature protocol under related secret-key shares, where both the modified shares and the subset of parties used for each signature computation, can be chosen adaptively. However, since in this case no player is actually

corrupted and the protocol's messages are sent via private channels, for each run of the signature protocol the attacker only learns the signature (but not the transcript).

Definition 12 (Tamper-resilient threshold signatures). *Let* $\Pi = (\mathsf{KGen}, \Xi, \mathsf{Vrfy})$ *be an* n*-party threshold signature scheme. We say that* Π *is secure under continuous memory tampering if for all PPT adversaries* A*:*

$$\mathbb{P}\left[\mathbf{G}^{\mathsf{nm\text{-}tsig}}_{\Pi,\mathsf{A}}(\lambda) = 1\right] \in \mathtt{negl}(\lambda),$$

where the game $\mathbf{G}^{\mathsf{nm\text{-}tsig}}_{\Pi,\mathsf{A}}(\lambda)$ *is described in Fig. 7.*

7.3 The Compiler

Given an n-party threshold signature $\Pi = (\mathsf{KGen}, \Xi, \mathsf{Vrfy})$, and an n-party TSS $\Sigma = (\mathsf{Share}, \mathsf{Rec})$, consider the following modified n-party threshold signature $\Pi^* = (\mathsf{KGen}^*, \Xi^*, \mathsf{Vrfy}^*)$.

- **Key generation** $\mathsf{KGen}^*(1^\lambda)$**:** Upon input the security parameter, run $(vk, sk_1, \ldots, sk_n) \leftarrow\!\!\$\ \mathsf{KGen}(1^\lambda)$, compute $(sk_{i,1}, \ldots, sk_{i,n}) \leftarrow\!\!\$\ \mathsf{Share}(sk_i)$ for each $i \in [n]$, set $sk_i^* = (sk_{i',i})_{i' \in [n]}$, and output $(vk, sk_1^*, \ldots, sk_n^*)$.
- **Signature protocol** $\Xi^* = (\mathsf{P}_1^*, \ldots, \mathsf{P}_n^*)$**:** For any subset $\mathcal{I} \subset [n]$, and any message $m \in \mathcal{M}$, the protocol $\langle \mathsf{P}_i^*(sk_i^*, m)\rangle_{i\in\mathcal{I}}$ proceeds as follows:
  - Party P_i^* parses $sk_i^* = (sk_{i',i})_{i' \in [n]}$ and sends $sk_{i',i}$ to the i'-th party, for every $i' \in \mathcal{I} \setminus \{i\}$.
  - Party P_i^* waits to receive the messages $sk_{i,i''}$ for every $i'' \in \mathcal{I} \setminus \{i\}$, and afterwards it computes $sk_i = \mathsf{Rec}((sk_{i,i''})_{i'' \in \mathcal{I}})$.
  - The players run $(\xi, \sigma) \xleftarrow{\Xi}\!\!\$\ \langle \mathsf{P}_i(sk_i, m)\rangle_{i\in\mathcal{I}}$.
- **Verification algorithm** Vrfy^***:** Return the same as $\mathsf{Vrfy}(vk, (m, \sigma))$.

Intuitively, in the above protocol we first create a verification key vk and secret-key shares $(sk_1, \ldots, sk_n)$ under Π; hence, each value sk_i is further divided into n shares $(sk_{i,1}, \ldots, sk_{i,n})$ via the secret sharing Σ. The final secret-key share sk_i^* for the i-th party consists of the shares $(sk_{1,i}, \ldots, sk_{n,i})$, i.e. the collection of all the i-th shares under Σ. In order to sign a message, each player first sends to each other player the corresponding share. This way, party P_i can reconstruct sk_i, and the involved players can then run the original signature protocol Ξ. The proof of the theorem below appears in the full version [36].

Theorem 6. *For any* $n, \varrho, \tau \in \mathbb{N}$ *such that* $n \geq \varrho \geq \tau$*, assuming that* Π *is non-interactive,* ϱ*-correct, and* τ*-private against honest-but-curious adversaries, and that* Σ *is an* $(n, \tau, \varrho, 0)$*-LR-CNMSS, then the above defined threshold signature* Π^* *is* ϱ*-correct,* τ*-private against honest-but-curious adversaries, and secure under continuous memory tampering.*

We give a sketch for the proof of Thm. 6. We focus on showing security against continuous tampering, as honest-but-curious security readily follows from the privacy of the CNMSS Σ and the honest-but-curious security of Π.

The proof is a classical hybrid argument where we switch step by step from the real distribution to a distribution where all the tampering queries are applied to shares which encode dummy secret keys. In this last experiment, the reduction can simulate the tampering oracle $\mathcal{O}_{\mathsf{sign}}(\vec{sk}, \cdot, \cdot, \cdot)$ as a function of the dummy shares only, and therefore the simulation is independent of the real secret keys. Thus, we can rely on the unforgeability of the non-interactive threshold signature scheme to conclude the proof.

However, there is a subtlety. In particular, in one of the intermediate hybrid steps, the adversary might, for example, overwrite the shares relative to a secret key sk_i with shares that reconstruct to an unrelated secret key $\tilde{sk}_i$, while keeping all the other shares untouched. If the starting threshold signature scheme would be interactive, we would need to be able to simulate a run of the signature protocol where all the inputs are the same but the i-th input, which lies out of the capability of an honest-but-curious adversary. On the other hand, if the threshold signature scheme is non-interactive as we assume, this problem disappears, as we can first run the signature protocol using the original secret-key shares, and later simulate the (single) message of the i-th server thanks to the knowledge of the mauled secret-key share $\tilde{sk}_i$.

8 Conclusions and Open Problems

We have initiated the study of *non-malleable, threshold* secret sharing withstanding a powerful adversary that can obtain both *noisy leakage* from each of the shares *independently*, and an *arbitrary polynomial* number of reconstructed messages corresponding to shares which can be *arbitrarily* related to the original ones (as long as the shares are modified *independently*). Importantly, in our model, both the tampering functions (mauling the original target secret sharing) and the reconstruction subsets (specifying which shares contribute to the reconstructed message) can be chosen *adaptively* by the attacker. Our main result establishes the existence of such schemes in the *computational setting*, under the minimal assumption of 1-to-1 OWFs, and with information rate asymptotically approaching 1 (as the message length goes to infinity).

Our work leaves several interesting open problems. We mention some of them below.

- **Mind the gap:** As we show, continuous non-malleability is impossible to achieve in the information-theoretic setting whenever the reconstruction threshold ϱ (i.e., the minimal number of shares required to reconstruct the message) is equal to the privacy threshold τ (i.e., any collection of $\tau - 1$ shares computationally hides the message). Our schemes, however, have a minimal gap $\varrho - \tau \geq 1$. It remains open to construct CNMSS for the optimal parameters $\varrho = \tau$, possibly with information-theoretic security (even without considering leakage and adaptive concurrent reconstruction).
- **Optimal rate:** It is well known that, in the computational setting, there exist robust threshold secret sharing schemes with optimal information rate n [48] (i.e., the size of each share is μ/n where μ is the message size). It

remains open whether continuously non-malleable threshold secret sharing schemes with such rate exist, and under which assumptions.

- **Arbitrary access structures:** Can we construct continuously non-malleable secret sharing beyond the threshold access structure, e.g. where the sets of authorized players can be represented by an arbitrary polynomial-size monotone span program, as in [43]?
- **Joint tampering:** Can we construct continuously non-malleable secret sharing where the non-malleability property holds even if joint tampering with the shares is allowed, as in [42,43]?
- **Applications:** Finally, it would be interesting to explore other applications of continuously non-malleable secret sharing besides tamper resistance, e.g. in the spirit of non-malleable cryptography, as in [41,26,44,24,42].

References

1. Aggarwal, D., Agrawal, S., Gupta, D., Maji, H.K., Pandey, O., Prabhakaran, M.: Optimal computational split-state non-malleable codes. In: TCC. pp. 393–417 (2016)
2. Aggarwal, D., Damgaard, I., Nielsen, J.B., Obremski, M., Purwanto, E., Ribeiro, J., Simkin, M.: Stronger leakage-resilient and non-malleable secret-sharing schemes for general access structures. Cryptology ePrint Archive, Report 2018/1147 (2018), https://ia.cr/2018/1147
3. Aggarwal, D., Dodis, Y., Kazana, T., Obremski, M.: Non-malleable reductions and applications. In: STOC. pp. 459–468 (2015)
4. Aggarwal, D., Dodis, Y., Lovett, S.: Non-malleable codes from additive combinatorics. In: STOC. pp. 774–783 (2014)
5. Aggarwal, D., Dodis, Y., Lovett, S.: Non-malleable codes from additive combinatorics. SIAM J. Comput. **47**(2), 524–546 (2018)
6. Aggarwal, D., Döttling, N., Nielsen, J.B., Obremski, M., Purwanto, E.: Continuous non-malleable codes in the 8-split-state model. Cryptology ePrint Archive, Report 2017/357 (2017), https://ia.cr/2017/357
7. Aggarwal, D., Dziembowski, S., Kazana, T., Obremski, M.: Leakage-resilient non-malleable codes. In: TCC. pp. 398–426 (2015)
8. Agrawal, S., Gupta, D., Maji, H.K., Pandey, O., Prabhakaran, M.: A rate-optimizing compiler for non-malleable codes against bit-wise tampering and permutations. In: TCC. pp. 375–397 (2015)
9. Ananth, P., Ishai, Y., Sahai, A.: Private circuits: A modular approach. In: CRYPTO. pp. 427–455 (2018)
10. Badrinarayanan, S., Srinivasan, A.: Revisiting non-malleable secret sharing. Cryptology ePrint Archive, Report 2018/1144 (2018), https://ia.cr/2018/1144
11. Beimel, A.: Secret-sharing schemes: A survey. In: International Conference on Coding and Cryptology (IWCC). pp. 11–46 (2011)
12. Ben-Or, M., Goldwasser, S., Wigderson, A.: Completeness theorems for non-cryptographic fault-tolerant distributed computation (extended abstract). In: STOC. pp. 1–10 (1988)
13. Bishop, A., Pastro, V., Rajaraman, R., Wichs, D.: Essentially optimal robust secret sharing with maximal corruptions. In: EUROCRYPT. pp. 58–86 (2016)
14. Blakley, G.R.: Safeguarding cryptographic keys. Proceedings of AFIPS 1979 National Computer Conference **48**, 313–317 (1979)

15. Blundo, C., Santis, A.D., Gargano, L., Vaccaro, U.: On the information rate of secret sharing schemes. Theor. Comput. Sci. **154**(2), 283–306 (1996)
16. Carpentieri, M., Santis, A.D., Vaccaro, U.: Size of shares and probability of cheating in threshold schemes. In: EUROCRYPT. pp. 118–125 (1993)
17. Chandran, N., Kanukurthi, B., Raghuraman, S.: Information-theoretic local non-malleable codes and their applications. In: TCC. pp. 367–392 (2016)
18. Chattopadhyay, E., Goyal, V., Li, X.: Non-malleable extractors and codes, with their many tampered extensions. In: STOC. pp. 285–298 (2016)
19. Chattopadhyay, E., Zuckerman, D.: Non-malleable codes against constant split-state tampering. In: FOCS. pp. 306–315 (2014)
20. Chaum, D., Crépeau, C., Damgård, I.: Multiparty unconditionally secure protocols (extended abstract). In: STOC. pp. 11–19 (1988)
21. Cheraghchi, M., Guruswami, V.: Capacity of non-malleable codes. In: Innovations in Theoretical Computer Science. pp. 155–168 (2014)
22. Cheraghchi, M., Guruswami, V.: Non-malleable coding against bit-wise and split-state tampering. In: TCC. pp. 440–464 (2014)
23. Chor, B., Goldreich, O.: Unbiased bits from sources of weak randomness and probabilistic communication complexity. SIAM J. Comput. **17**(2), 230–261 (1988)
24. Coretti, S., Dodis, Y., Tackmann, B., Venturi, D.: Non-malleable encryption: Simpler, shorter, stronger. In: TCC. pp. 306–335 (2016)
25. Coretti, S., Faonio, A., Venturi, D.: Rate-optimizing compilers for continuously non-malleable codes. Cryptology ePrint Archive, Report 2019/055 (2019), https://ia.cr/2019/055
26. Coretti, S., Maurer, U., Tackmann, B., Venturi, D.: From single-bit to multi-bit public-key encryption via non-malleable codes. In: TCC. pp. 532–560 (2015)
27. Davì, F., Dziembowski, S., Venturi, D.: Leakage-resilient storage. In: SCN. pp. 121–137 (2010)
28. Desmedt, Y., Frankel, Y.: Shared generation of authenticators and signatures (extended abstract). In: CRYPTO. pp. 457–469 (1991)
29. Dodis, Y., Haralambiev, K., López-Alt, A., Wichs, D.: Cryptography against continuous memory attacks. In: FOCS. pp. 511–520 (2010)
30. Dodis, Y., Ostrovsky, R., Reyzin, L., Smith, A.D.: Fuzzy extractors: How to generate strong keys from biometrics and other noisy data. SIAM J. Comput. **38**(1), 97–139 (2008)
31. Dolev, D., Dwork, C., Waarts, O., Yung, M.: Perfectly secure message transmission. J. ACM **40**(1), 17–47 (1993)
32. Dziembowski, S., Kazana, T., Obremski, M.: Non-malleable codes from two-source extractors. In: CRYPTO. pp. 239–257 (2013)
33. Dziembowski, S., Pietrzak, K., Wichs, D.: Non-malleable codes. In: Innovations in Computer Science. pp. 434–452 (2010)
34. Faonio, A., Nielsen, J.B., Simkin, M., Venturi, D.: Continuously non-malleable codes with split-state refresh. In: ACNS. pp. 1–19 (2018)
35. Faonio, A., Nielsen, J.B., Venturi, D.: Fully leakage-resilient signatures revisited: Graceful degradation, noisy leakage, and construction in the bounded-retrieval model. Theor. Comput. Sci. **660**, 23–56 (2017)
36. Faonio, A., Venturi, D.: Non-Malleable Secret Sharing in the Computational Setting: Adaptive Tampering, Noisy-Leakage Resilience, and Improved Rate. Cryptology ePrint Archive, Report 2019/105 (2019), https://ia.cr/2019/105
37. Faust, S., Mukherjee, P., Nielsen, J.B., Venturi, D.: Continuous non-malleable codes. In: TCC. pp. 465–488 (2014)

38. Faust, S., Rabin, T., Reyzin, L., Tromer, E., Vaikuntanathan, V.: Protecting circuits from leakage: the computationally-bounded and noisy cases. In: EUROCRYPT. pp. 135–156 (2010)
39. Gennaro, R., Lysyanskaya, A., Malkin, T., Micali, S., Rabin, T.: Algorithmic tamper-proof (ATP) security: Theoretical foundations for security against hardware tampering. In: TCC. pp. 258–277 (2004)
40. Goldreich, O., Micali, S., Wigderson, A.: How to play any mental game or A completeness theorem for protocols with honest majority. In: STOC. pp. 218–229 (1987)
41. Goyal, V., Jain, A., Khurana, D.: Witness signatures and non-malleable multi-prover zero-knowledge proofs. Cryptology ePrint Archive, Report 2015/1095 (2015), http://ia.cr/2015/1095
42. Goyal, V., Kumar, A.: Non-malleable secret sharing. In: STOC. pp. 685–698 (2018)
43. Goyal, V., Kumar, A.: Non-malleable secret sharing for general access structures. In: CRYPTO. pp. 501–530 (2018)
44. Goyal, V., Pandey, O., Richelson, S.: Textbook non-malleable commitments. In: STOC. pp. 1128–1141 (2016)
45. HashiCorp: The Vault project. https://www.vaultproject.io/, accessed: 2018-12-22
46. Ishai, Y., Sahai, A., Wagner, D.A.: Private circuits: Securing hardware against probing attacks. In: CRYPTO. pp. 463–481 (2003)
47. Kanukurthi, B., Obbattu, S.L.B., Sekar, S.: Four-state non-malleable codes with explicit constant rate. In: TCC. pp. 344–375 (2017)
48. Krawczyk, H.: Secret sharing made short. In: CRYPTO. pp. 136–146 (1993)
49. Kumar, A., Meka, R., Sahai, A.: Leakage-resilient secret sharing. Cryptology ePrint Archive, Report 2018/1138 (2018), https://ia.cr/2018/1138
50. Li, X.: Improved non-malleable extractors, non-malleable codes and independent source extractors. In: STOC. pp. 1144–1156 (2017)
51. Liu, F., Lysyanskaya, A.: Tamper and leakage resilience in the split-state model. In: CRYPTO. pp. 517–532 (2012)
52. Naor, M., Segev, G.: Public-key cryptosystems resilient to key leakage. SIAM J. Comput. **41**(4), 772–814 (2012)
53. Ostrovsky, R., Persiano, G., Venturi, D., Visconti, I.: Continuously non-malleable codes in the split-state model from minimal assumptions. In: CRYPTO. pp. 608–639 (2018)
54. Rabin, T., Ben-Or, M.: Verifiable secret sharing and multiparty protocols with honest majority (extended abstract). In: STOC. pp. 73–85 (1989)
55. Rogaway, P., Bellare, M.: Robust computational secret sharing and a unified account of classical secret-sharing goals. In: CCS. pp. 172–184 (2007)
56. Shamir, A.: How to share a secret. Commun. ACM **22**(11), 612–613 (1979)
57. Shoup, V.: Sequences of games: A tool for taming complexity in security proofs. Cryptology ePrint Archive, Report 2004/332 (2004), http://ia.cr/2004/332
58. Srinivasan, A., Vasudevan, P.N.: Leakage resilient secret sharing and applications. Cryptology ePrint Archive, Report 2018/1154 (2018), https://ia.cr/2018/1154