Non-Malleable Secret Sharing for General Access Structures

Vipul Goyal Ashutosh Kumar

CMU. goyal@cs.cmu.edu UCLA. a@ashutoshk.com

Abstract. Goyal and Kumar (STOC'18) recently introduced the notion of *non-malleable secret sharing.* Very roughly, the guarantee they seek is the following: the adversary may potentially tamper with all of the shares, and still, *either* the reconstruction procedure outputs the original secret, *or*, the original secret is "destroyed" and the reconstruction outputs a string which is completely "unrelated" to the original secret. Prior works on non-malleable codes in the 2 split-state model imply constructions which can be seen as 2-out-of-2 non-malleable secret sharing (NMSS) schemes. Goyal and Kumar proposed constructions of t-out-of-n NMSS schemes. These constructions have already been shown to have a number of applications in cryptography.

We continue this line of research and construct NMSS for more general access structures. We give a generic compiler that converts any statistical (resp. computational) secret sharing scheme realizing any access structure into another statistical (resp. computational) secret sharing scheme that not only realizes the same access structure but also ensures statistical non-malleability against a computationally unbounded adversary who tampers each of the shares arbitrarily and independently. Instantiating with known schemes we get unconditional NMMS schemes that realize any access structures generated by polynomial size monotone span programs. Similarly, we also obtain conditional NMMS schemes realizing access structure in **monotone P** (resp. **monotone NP**) assuming one-way functions (resp. witness encryption).

Towards considering more general tampering models, we also propose a construction of n-out-of-n NMSS. Our construction is secure even if the adversary could divide the shares into any two (possibly overlapping) subsets and then arbitrarily tamper the shares in each subset. Our construction is based on a property of inner product and an observation that the inner-product based construction of Aggarwal, Dodis and Lovett (STOC'14) is in fact secure against a tampering class that is stronger than 2 split-states. We also show applications of our construction to the problem of non-malleable message transmission.

1 Introduction

Secret sharing is a fundamental primitive in cryptography which allows a dealer to distribute shares of a secret among several parties, such that only authorized subsets of parties can recover the secret; the secret is "hidden" from all the unauthorized set of parties. Shamir [Sha79] and Blakley [Bla79] initiated the study of secret sharing by constructing threshold secret sharing schemes that only allows at least t-out-of-n parties to reconstruct the secret. A rich line of works have studied the construction of secret sharing schemes for more advanced access structures [KW93, Bei, Bei11, KNY14].

A number of works have studied the setting where the primary goal of the adversary is to instead *tamper* with the secret. This relates to the line of works on error detecting codes such as algebraic manipulation detection(AMD) codes [CDF$^+$08], and, verifiable secret sharing [RBO89]. A more detailed overview of the related works can be found later in this section.

Non-malleable secret sharing. Very recently, Goyal and Kumar [GK18] initiated a systematic study of what they call *non-malleable secret sharing.* Very roughly, the guarantee is the following: the adversary may potentially tamper with all of the shares, and still, *either* the reconstruction procedure outputs the original secret, *or*, the original secret is "destroyed" and the reconstruction outputs a string which is completely "unrelated" to the original secret. This is a natural guarantee which is inspired by applications in cryptography.

As noted by [GK18], 2-out-of-2 non-malleable secret sharing (NMSS) is equivalent to non-malleable codes in the 2 split-state model. Constructing such split state non-malleable codes has proven to be surprisingly hard. Though a brilliant line of works [DPW10, LL12, DKO13, ADL14, CGL16, Li17], such 2-split-state codes have been constructed. However such an implication does not hold if the number of shares is more than 2. To see this, consider a (contrived) example of a 3 split-state non-malleable code where the encoding functions encodes the message using a 2 split-state non-malleable code to obtain the first two states and outputs the message (in the clear) in the third state. The decoding function simply ignores the third state and uses the first two states to decode the message. Such a construction is a valid 3 split-state non-malleable code that is not a 3-out-of-3 secret sharing scheme (in fact, it has no secrecy at all). Towards that end, Goyal and Kumar proposed a construction of t-out-of-n NMSS scheme where reconstruction could be done given *any* t shares, any set of less than t shares has no information about the original secret, and, non-malleability is guaranteed even if an adversary may tamper with each share.

Even though a relatively new primitive, non-malleable coding in the split state model (or 2-out-of-2 NMSS) has already found a number of applications in cryptography including in tamper-resilient cryptography [DPW10], designing multi-prover interactive proof systems [GJK15] and obtaining efficient encryption schemes [CDTV16]. Very recently, non-malleable codes in the split-state model were used as 2-out-of-2 non-malleable secret sharing scheme to obtain 3-round protocol for non-malleable commitments [GPR16].

Our Question. We study the following natural question in this work:

Can we get non-malleable secret sharing schemes for access structures beyond threshold?

As noted before, known results on split state non-malleable codes provide 2-out-of-2 NMSS. Goyal and Kumar [GK18] recently took a significant step forward by constructing t-out-of-n NMSS schemes. However to our knowledge, NMSS are not known access structures beyond threshold. For example, can we get NMSS schemes for access structures which can be represented using log depth circuits or polynomial sized boolean formulas? Can we get a NMSS for all of **monotone P** ? Or even better, can we get a NMSS for all of **monotone NP** ?

Existing secret-sharing schemes. As noted by Goyal and Kumar, most of the secret sharing schemes known are linear [Bei, chapter 4] and have nice algebraic and geometric properties, which are harnessed to obtain efficient sharing and reconstruction procedures. *Non-malleable secret sharing schemes on the other hand cannot be linear.* As the secret is a linear combination of the shares in a linear secret sharing scheme, the adversary can perform local operations on each of the shares and encode any linear function of the secret. Indeed, the malleability of linear secret sharing schemes, such as polynomials based Shamir's secret sharing scheme [Sha79], forms the basis of secure multi-party computation protocols [BOGW88]. For the purpose of constructing NMMS, any such alteration is an "attack" and the goal is to build secret sharing schemes that necessarily prohibit any such attacks.

1.1 Our Results

Generic compiler for individual tampering. Recall that an access structure $\mathcal{A}$ is a monotone collection of subsets of parties (such that every subsets of parties in this set are authorized to reconstruct the secret; other subset of parties are unauthorized). Our first main result is the following:

Theorem 1. *(informal) For any access structure $\mathcal{A}$ that does not contain singletons*[1]*, if there exists an efficient statistical (resp. computational) secret sharing scheme realizing access structure $\mathcal{A}$, then there exists an efficient statistical (resp. computational) secret sharing scheme realizing $\mathcal{A}$ that is statistically non-malleable against an adversary who tampers each of the shares arbitrarily and independently.*

Karchmer and Wigderson [KW93] gave an efficient[2] secret sharing scheme for access structures that can be described by a polynomial-size monotone *span program.* This is a general class for which efficient secret sharing schemes are known

[1] We note that this is a necessary assumption, as otherwise the notion of non malleability becomes meaningless. A single authorized party can recover the message and trivially encode any related message

[2] A statistical secret sharing scheme is efficient if the sharing and reconstruction functions run in $\mathbf{poly}\big(n, k, \log(1/\epsilon)\big)$ time where k is the size of the message and $\epsilon > 0$ is the statistical error.

and includes undirected connectivity in a graph. Instantiating our compiler with their scheme, we obtain the following corollary.

Corollary 1. *(informal) For any access structure that can be described by a polynomial-size monotone* span program *and does not contain a singleton, there exists an efficient statistical secret sharing scheme that is statistically non-malleable against an adversary who arbitrarily tampers each of the shares independently.*

In an unpublished work (mentioned in [Bei11, KNY14]), Yao constructed an efficient computational secret-sharing scheme for access structures whose characteristic function are computable by monotone circuit of polynomial-size (assuming just one-way functions). Using this scheme, we get,

Corollary 2. *(informal) If one-way functions exist, then for any access structure $\mathcal{A}$ that does not contain singletons and is computable by monotone boolean circuits of polynomial size, there exists an efficient computational secret sharing scheme that realizes $\mathcal{A}$ and is statistically non-malleable against an adversary who arbitrarily tampers each of the shares independently.*

Observe that the secret sharing scheme resulting from the above theorem has *statistical* non-malleability (even though the secrecy is computational). Furthermore, Komargodski et al. [KNY14], constructed efficient computational secret sharing scheme for every **monotone NP** access structure assuming one way functions and witness-encryption for **NP** [GGSW13]. This gives us the following:

Corollary 3. *(informal) If one-way functions and witness-encryption for* **NP** *exist, then for every* **monotone NP** *access structure $\mathcal{A}$ that does not contain singletons and supports efficient membership queries, there exists an efficient computational secret sharing scheme that realizes $\mathcal{A}$ and is statistically non-malleable against an adversary who arbitrarily tampers each of the shares independently.*

We say that an access structure supports efficient membership queries, if it is possible to efficiently decide whether a given subset of parties is authorized or not. For t-out-of-n, this is trivial. Similarly, for access structures based on polynomial sized monotone boolean circuits, one can execute the corresponding circuit to decide whether the input subset is authorized or not.

Towards Stronger Tampering Models. In addition to the individual tampering model, Goyal and Kumar [GK18] also considered *joint* tampering where an adversary may divide the set of shares into two *disjoint* sets and may tamper with the shares in each set jointly. They additionally required the two subsets to have different cardinalities (i.e., both of them must not have equal number of shares). This holds even for the basic case of n-out-of-n secret sharing. We present a new construction of n-out-of-n NMSS against a significantly more general class of tampering functions. In particular, the adversary may partition the shares into any two (possibly overlapping) sets having up to $n-1$ shares. For example, the adversary may use the first $n-1$ shares to produce the tampered version of first $\frac{n}{2}$ shares, and uses the last $n-1$ shares to produce the last $\frac{n}{2}$ shares.

Theorem 2. *(informal) For any integer $n \geq 2$, there exists an efficient statistical secret sharing scheme that encodes a secret into n shares, allows for reconstruction of the secret only when all the n shares are available, and is also statistically non-malleable against an adversary who partitions the n shares into any two (possibly overlapping) non-empty subsets of its choice having up to $n-1$ shares each, and then, arbitrarily tampers the shares in each of the subsets (independently of the shares in the other subset).*

Our techniques in fact extend to allow the tampering of each share to depend on *all* the n shares in a limited way (see Section 4 for more details).

Ito et al. [ISN89] showed that every access structure has a (possibly inefficient) secret sharing scheme. In a manner similar to their construction, we can use the above n-out-of-n NMSS scheme for every minimal authorized set and obtain the following existential result.

Corollary 4. *For any access structure $\mathcal{A}$ that does not contain singletons, there exists a (possibly inefficient) statistical secret sharing scheme that realizes $\mathcal{A}$ and is statistically non-malleable against an adversary who chooses any minimal authorized set, partitions it into two subset and arbitrarily tampers shares in each of the subsets independently.*

Interesting Corollaries of Our Techniques. We observe that the inner-product construction of non-malleable codes of Aggarwal et al. [ADL14] can in fact withstand tampering which is stronger than 2 split state tampering.

Corollary 5. *(informal) The 2 split-state non-malleable code of Aggarwal et al. [ADL14] encodes a message as two vectors L and R of length λ over prime field Z_p. This scheme is even secure against an adversary*

$$\widetilde{L} \leftarrow f_1(L) \odot g_1(R)$$

$$\widetilde{R} \leftarrow f_2(L) \odot g_2(R)$$

where (f_1, f_2, g_1, g_2) are arbitrary tampering functions and $\odot$ represents coordinate-wise multiplication of two vectors (that is $L \odot R = (L_1 \times R_1, L_2 \times R_2, \ldots, L_\lambda \times R_\lambda)$).

Compared to leakage-resilient non-malleable codes where the tampering of the left share can depend on a bounded amount of information about the the right share, in the above, *the tampered left share can be exactly equal to the right share.*

As an application of NMSS, [GK18] initiated the study of *non-malleable message transmission.* This guarantees that the receiver either receives the original message, or, the original message is essentially destroyed and the receiver receives an "unrelated" message, when the network is under the influence of an adversary who can execute arbitrary protocol on each of the nodes in the network (apart from the sender and the receiver). The adversary is even allowed to add a bounded number of arbitrary hidden links which it can use in addition to the original links for communicating amongst corrupt nodes.

Our techniques allow us to obtain a *strict improvement* over the results in [GK18]. In fact, our result is tight. We first informally define the notion of non-malleable paths. For a network represented by an undirected graph G, let G' be the induced subgraph of G with sender S and R removed. We define a collection of paths from S to R to be non-malleable if in the induced subgraph G' any node is reachable by nodes present on at most one of these paths.

Corollary 6. *In any network, with a designated sender S and receiver R, if there exists a collection of n non-malleable paths from S to R, then non-malleable secure message transmission protocol is possible with respect to an adversary which adds at most $n-2$ arbitrary hidden links in the network and byzantinely corrupts all nodes other than S and R. Moreover, the bound of $n-2$ is tight.*

1.2 Our Techniques

First we briefly recall the construction of t-out-of-n NMSS secure against an adversary which tampers each share independently [GK18].

Construction of [GK18]. Assume $t \geq 3$. First they encode the secret m using a 2 split-state non-malleable code to obtain $l, r \leftarrow \mathbf{NMEnc}(m)$. Then they share l using any t-out-of-n secret-sharing scheme to obtain $l_1, \ldots, l_n$, and, encode r using a 2-out-of-n *leakage-resilient* secret-sharing scheme to obtain $r_1, \ldots, r_n$. Final shares are of the form $share_i = (l_i, r_i)$. Given an adversary A who tampers with each share $share_i$ arbitrarily and independently, we would like to construct a split state adversary (f, g) against the underlying non-malleable code. A (somewhat oversimplified) high level structure of their proof is as follows:

1. Fix shares $l_1, \ldots, l_{t-1}$ independent of the secret m. This can be done since l is shared using a t-out-of-n secret-sharing and $t \geq 3$. Shares $l_1, \ldots, l_{t-1}$ are hardcoded in the description of f and g.
2. The function g gets r as input and must output $\widetilde{r}$, the tampered version of r. Given r, g samples r_1, r_2 and hence now has $share_1 = (l_1, r_1)$ and $share_2 = (l_2, r_2)$ (since l_1 and l_2 are hardcoded). Use adversary A to compute $\widetilde{share}_1$ and $\widetilde{share}_2$, and hence, $\widetilde{r_1}$ and $\widetilde{r_2}$. Reconstruct $\widetilde{r}$ using $\widetilde{r_1}$ and $\widetilde{r_2}$ (recall r was shared using a 2-out-of-n scheme) and output it.
3. The function f gets l as input and must output $\widetilde{l}$. As the first step, f uses l to sample l_t *which is consistent* with the fixed shares $l_1, \ldots, l_{t-1}$. Next, f must run adversary A to compute tampered shares $\widetilde{share}_1, \ldots, \widetilde{share}_t$ which would allow for recovery of $\widetilde{l}_1, \ldots, \widetilde{l}_t$ and hence $\widetilde{l}$. However note that f does not have $(r_1, \ldots, r_t)$ and therefore cannot even compute $share_1$. In fact, it cannot have any two shares of r, as the tampering function f needs to be independent of r. Towards that end, [GK18] rely on the leakage resilience of the secret sharing scheme to compute $\widetilde{l_1}, \ldots, \widetilde{l_t}$.

Note that the above proof structure does not work when $t = 2$. For this case, they device a (completely separate) 2-out-of-n NMSS scheme by giving every pair an independent non-malleable encoding of the secret m.

Getting NMSS for general access structures. The natural starting point would be to replace the t-out-of-n secret sharing used to share l by the given secret sharing for the access structure in question. Instantiating this with various computational and information theoretic secret sharing schemes would presumably lead to NMSS for a variety of access structures including **monotone P** . However this idea fails because of the following two basic issues.

Firstly, we have to deal with authorized sets of size two ('pairs') in the given access structure (in case there are any). In case of [GK18], this was achieved by simply giving an entirely different construction (with a separate proof) for the case of $t = 2$. However in the setting of general access structures, the authorized set of size two may coexist with authorized sets of larger size. We solve this issue by efficiently constructing another access structure that has all authorized sets that contain an authorized subset of size two, in addition to the original access structure. Our hope would be to run NMSS for both these access structures in "parallel" for the same message. However this leads to additional difficulties in the proof of security related to composition: any authorized *pair* of parties will now have the same message encoded under two different schemes, and the split-state reduction to non-malleable codes fails.

Secondly, the construction in [GK18] heavily makes use of the fact that one can sample some of the shares without having knowledge of the secret at all. Then once the secret is available, you can "adjust" the remaining shares such that the resulting set of shares altogether is sampled from the correct distribution. As an example, see how the share l_t is sampled in step 3 (see the summary of [GK18] construction above). Indeed, such sampling is not just done once but at multiple steps in the [GK18] construction. In the computational case however, such an approach inherently breaks down. Since each share may have complete information about the secret (the secret may only be computationally hidden), one may not be able to sample a few shares independently of the secret and then "adjust" the rest so that overall, they come from the correct distribution. One could try to argue that even if the shares are sampled incorrectly, since the tampering function does not get all of them as input, it may anyway be indistinguishable to the tampering functions. However, such a guarantee is not sufficient for non-malleability. The tampering functions individually may not be able to distinguish correct shares from incorrect ones, and yet, the distribution of their joint output might change completely.

To solve these issues, we use two additional ideas to make our construction work.

1. Introduce "limited" information theoretic secrecy: We first compile the underlying statistical (resp. computational) secret sharing scheme into another which additionally guarantees that any two shares hide the secret *information theoretically* (even if the secret sharing scheme was computational to begin with). This not only solves the first issue, but also paves a way to the

solution of the second issue. For the first issue, this approach allows us to use non-malleable codes in a black-box way, as opposed to an alternative approach, where we could have strengthened the underlying split-state code to ensure non-malleability against "parallel" tamperings. For the second issue, we are now allowed to fix up to two shares of l even for computational schemes.

2. We use a secret sharing scheme with stronger leakage resilience properties: For any two secrets, suppose an adversary is given some valid shares of each of the secrets (potentially enough even to reconstruct the secret). Additionally, the adversary is given individual leakage from the rest of the shares of one secret. It should be statistically impossible for the adversary to identify whether the leakage corresponds to the first or the second secret. This property is significantly stronger than the one needed by Goyal and Kumar [GK18]. Unlike the proof of [GK18], this allows our reduction to generate t shares that are statistically quite far from any valid set of t shares, and still achieve statistical non-malleability.

Towards stronger tampering models. Let us try to construct n-out-of-n secret sharing schemes that are non-malleable against an adversary that arbitrarily partitions the n shares into two non-empty subsets and jointly tampers the shares in each of the these subsets independently.

First Attempt. Let us try to use a 2 split-state non-malleable code that encodes the message into two parts, say l and r. We let l be the first share, and obtain the last $n-1$ shares by secret sharing r using a traditional (n-1)-out-of-(n-1) secret sharing scheme. However, if the adversary tampers the first and last shares together, the tampered versions of last share (in particular r) may depend of the first share l and we will be not be able to obtain a split-state reduction to the underlying non-malleable code.

Second Attempt. What about a tree-based construction ? Consider, for example, a complete binary tree with 2^k leaves corresponding to 2^k parties. To share a secret, we put the secret at the root of this tree, and encode it using a non-malleable code to obtain the value of nodes at level 1 (children of root). We can recursively apply this process using several non-malleable codes to obtain the value of all the 2^k leaves, and these values correspond to the shares of 2^k parties. While this seems like a promising approach, the share size increases exponentially with the depth of the tree (as constant rate statistical split-state non-malleable codes are not yet constructed). Even more fundamentally, it is not clear how to prove that such a construction is secure against arbitrary joint tampering. As a concrete example, consider a simple depth 2 tree having 4 leaves. Suppose adversary tampers the first and the last leaf together, and independently tampers the second and the third leaf. It seems that stronger notions of non-malleable codes (while maintaining constant rate) are needed. Moreover, it appears that different properties might be needed for different choices of partitioning.

Third Attempt. Can we extend the techniques of [GK18] ? Unfortunately, when the two subsets are of equal cardinality, their technique of using different degree polynomials no longer seems to work.

Our construction : We take a step back and construct n-out-of-n scheme in a manner similar to the first attempt described above. Recall that we were struck while tyring to obtain a split-state reduction to the underlying non-malleable code. Nevertheless, we observe an underlying 'multiplicative structure' present in the code of Aggarwal et al. [ADL14] (hereby refered to as ADL construction) to achieve split-state reduction avoiding the problem mentioned in the first attempt.

We begin by recalling the elegant inner-product based ADL construction. They prove an amazing property of inner product, which roughly states that any independent tampering of left and right vector can be translated to an affine tampering of the output of inner product. This observation, reduces the problem to creating non-malleable codes against split-state arbitrary tampering functions to creating non-malleable codes against an affine function. To this end, they introduce affine evasive function, which ensures non-malleability against tampering by affine functions. Their proof relies on the linearity property satisifed by inner-product and is highly non-trivial relying on new results proved in additive combinatorics.

Given two equal length vectors over some finite field, the decoder of ADL computes inner-product and then applies the affine-evasive function to the output. Instead of viewing the first step as inner-product, we take a more fine-grained approach, and consider coordinate-wise multiplication of vectors to be the first step, followed by an addition of the coordinates. Our main observation is that the set of equal length vectors containing non-zero coordinates forms a finite abelian group under the operation of coordinate-wise multiplication of vectors. Next, we recall that Karnin et al. [KGH83] have shown how to use any abelian group to construct a n-out-of-n secret sharing scheme. The resulting scheme is quite simple, the reconstruction function will perform coordinate-wise multiplication of all the n vectors to obtain the secret vector, and we can proceed as in ADL, by computing sum of coordinates and then applying the affine evasive function to the sum.

We elaborated our scheme in the above fashion, instead of directly stating that we will use generalized inner-product instead of inner-product, because it is more insightful in conveying our proof ideas. In particular, we essentially use the associativity and commutativity of the mentioned abelian group (formed by coordinate-wise multiplication of non-zero field elements) to handle arbitrary partitions. Given any paritioning of n vectors into two subsets, we can use the commutativity of the abelian group to collect all the vectors of the first subset, and independently collect all the vectors of the second subset together. After which we can use the associativity of the same group to coordinate-wise multiply all the vectors in the first subset together, and independently coordinate-wise multiply all the vectors of the second subset. Notice, that now we are left with exactly two vectors corresponding to each of the two subsets, and we might be able to utilize the non-malleability of the ADL construction which works for two vectors. If we did not rely on this structure, we would have had to generalize the entire additive-combinatorics based proof of the ADL construction.

Paper organization. We define various primitives in section 2. We give our generic compiler in section 3. We give the construction of n-out-of-n schemes supporting joint-tampering in section 4.

Related Works. A number of works in the literature ensure that the correct secret is recovered even when some number of shares are arbitrarily corrupted. Concepts from error correcting codes have been useful in obtaining such schemes [Sha79, MS81]. In a seminal work [RBO89], Rabin and Ben-Or introduced verifiable secret sharing, which allowed the adversary to tamper almost half the shares, and still ensured that the adversary cannot cause the reconstruction procedure to output an incorrect message (except with exponentially small error probability). Cramer et. al. [CDF$^+$08], in a beautiful work introduced algebraic manipulation detection(AMD) codes and gave almost optimal constructions for them. These codes allow the adversary to "blindly" add any value to the codeword, and ensure that any such algebraic tampering will be detected with high probability. They used such codes to construct robust secret sharing schemes, which allowed adversary to tamper with any unauthorized subset of shares.

As already noted, 2 split state non-malleable codes can be seen as 2-out-of-2 non-malleable secret sharing schemes in which both the shares can be independently tampered. Though a brilliant line of works, such split-state non-malleable codes have been constructed [DPW10, LL12, DKO13, ADL14, CGL16, Li17]. [GK18] construct t-out-of-n non-malleable secret sharing schemes.

2 Definitions

We use capital letters to denote distributions and their support, and corresponding small letters to denote a sample from the distribution. Let $[m]$ denote the set $\{1, 2, \ldots, m\}$, and U_r denote the uniform distribution over $\{0, 1\}^r$. Unless otherwise stated, $\mathbb{F}_p$ is a finite field of order prime (power) p. For any set $B \in [n]$, let $\otimes_{i \in B} S_i$ denote the Cartesian product $S_{i_1} \times S_{i_2} \times \ldots \times S_{i_{|B|}}$, where $i_1, i_2 \ldots i_{|B|}$ are ordered elements of B, such that $i_j < i_{j+1}$.

Definition 1. ***(min-entropy)****. The min-entropy of a source X is defined as*

$$H_\infty(X) = \min_{x \in Support(X)} \left\{ \frac{1}{log(Pr[X = x])} \right\}$$

A (n, k)-source is a distribution on $\{0, 1\}^n$ with min-entropy k. A distribution D is flat if it is uniform over a set S.

Definition 2. ***(Statistical Distance)*** *Let D_1 and D_2 be two distributions on a set S. The statistical distance between D_1 and D_2 is defined to be :*

$$|D_1 - D_2| = \max_{T \subseteq S} |D_1(T) - D_2(T)| = \frac{1}{2} \sum_{s \in S} |Pr[D_1 = s] - Pr[D_2 = s]|$$

We say D_1 is ϵ-close to D_2 if $|D_1 - D_2| \leq \epsilon$. Sometimes we represent the same using $D_1 \approx_\epsilon D_2$.

2.1 Non-Malleable Codes

Definition 3. *(**Coding Schemes**)([ADL14]). A coding scheme consists of two functions : an encoding function (possibly randomized) $Enc : \mathcal{M} \to \mathcal{C}$, and a deterministic decoding function $Dec : \mathcal{C} \to \mathcal{M} \cup \{\perp\}$ such that , for each $m \in \mathcal{M}$, $Pr(Dec(Enc(m)) = m) = 1$ (over the randomness of the encoding function).*

Definition 4. *(**Non-Malleable Codes**) ([ADL14]). Let $\mathcal{F}$ be some family of tampering functions. For each $f \in \mathcal{F}$, and $m \in \mathcal{M}$, define the tampering experiment*

$$\mathbf{Tamper^f_m} = \left\{ \begin{matrix} c \leftarrow Enc(m) \\ \tilde{c} \leftarrow f(c) \\ \tilde{m} \leftarrow Dec(\tilde{c}) \\ Output : \tilde{m} \end{matrix} \right\}$$

*which is random variable over the randomness of the encoding function Enc. We say a coding scheme (Enc, Dec) is ϵ-**non-malleable** w.r.t $\mathcal{F}$ if for each $f \in \mathcal{F}$, there exists a distribution D^f (corresponding to the simulator) over $\mathcal{M} \cup \{same^*, \perp\}$ such that, for all $m \in \mathcal{M}$, we have that the statistical distance between $Tamper^f_m$ and*

$$\mathbf{Sim^f_m} = \left\{ \begin{matrix} \tilde{m} \leftarrow D^f \\ Output : m \; if \, \tilde{m} = same^*, or \; \tilde{m}, otherwise \end{matrix} \right\}$$

is at most ϵ. Additionally, D^f should be efficiently samplable given oracle access to $f(.)$.

2.2 Secret Sharing Schemes

The following definition is inspired from the survey [Bei11].

Definition 5. *(**Access Structure and Sharing function**) A collection $\mathcal{A}$ is called monotone if $B \in \mathcal{A}$ and $B \subseteq C$, then $C \in \mathcal{A}$. Let $[n] = \{1, 2, \ldots, n\}$ be a set of identities of n parties. An **access structure** is a monotone collection $\mathcal{A} \subseteq 2^{\{1,\ldots,n\}}$ of non-empty subsets of $[n]$. Sets in $\mathcal{A}$ are called **authorized**, and sets not in $\mathcal{A}$ are called **unauthorized**.*

*For any access structure $\mathcal{A}$, we define **minimal basis access structure** of $\mathcal{A}$, denoted by $\mathcal{A}^{min}$, as the the minimal subcollection of $\mathcal{A}$, such that for all authorized set $T \in \mathcal{A}$, there exists an authorized subset $B \subseteq T$ which is an element of $\mathcal{A}^{min}$.*

*Let $\mathcal{M}$ be the domain of secrets. A **sharing function** Share is a randomized mapping from $\mathcal{M}$ to $S_1 \times S_2 \times \ldots \times S_n$, where S_i is called the domain of shares* *of party with identity j. A dealer distributes a secret $m \in \mathcal{M}$ by computing the vector $Share(m) = (s_1, \ldots, s_n)$, and privately communicating each share s_j to the party j. For a set $S \subseteq \{p_1, \ldots, p_n\}$, we denote $Share(m)_S$ to be a restriction of $Share(m)$ to its S entries.*

Definition 6. *(**Secret Sharing Scheme** [Bei11]). Let $\mathcal{M}$ be a finite set of secrets, where $|\mathcal{M}| \geq 2$. A sharing function Share with domain of secrets $\mathcal{M}$ is a (n, ϵ)-**Secret Sharing Scheme** realizing an access structure $\mathcal{A}$ if the following two properties hold :*

1. ***Correctness**. The secret can be reconstructed by any authorized set of parties. That is, for any set $B \in \mathcal{A}$, where $B = \{i_1, \ldots, i_{|B|}\}$, there exists a deterministic reconstruction function $Rec^B : \otimes_{i \in B} S_i \to \mathcal{M}$ such that for every $m \in \mathcal{M}$,*

$$Pr[\mathbf{Rec}^B(\mathbf{Share}(m)_B) = m] = 1$$

(over the randomness of the Sharing function)

2. ***Statistical Privacy**. Any collusion of unauthorized parties should have "almost" no information about the underlying secret. More formally, for any unauthorized set $T \notin \mathcal{A}$, and for every pair of secrets $a, b \in \mathcal{M}$, for any distinguisher D with output in $\{0, 1\}$, the following holds :*

$$|Pr_{shares \leftarrow Share(a)}[D(shares_T) = 1] - Pr_{shares \leftarrow Share(b)}[D(shares_T) = 1]| \leq \epsilon$$

The special case of $\epsilon = 0$, is known as Perfect Privacy.

We use the definition of leakage-resilience from [GK18].

Definition 7. *(**Leakage-Resilient Secret Sharing Schemes**) Let $\mathcal{L}$ be some family of leakage functions. We say that the (n, ϵ)-secret sharing scheme, (Share, Rec), realizing access structure $\mathcal{A}$ is ϵ'-**leakage-resilient** w.r.t $\mathcal{L}$ if for each $f \in \mathcal{L}$, and for any two messages $a, b \in \mathcal{M}$, any distinguisher D with output in $\{0, 1\}$, the following holds :*

$$|Pr_{shares \leftarrow Share(a)}[D(f(shares)) = 1] - Pr_{shares \leftarrow Share(b)}[D(f(shares)) = 1]| \leq \epsilon'$$

We generalize the definition of non-malleable secret sharing schemes of [GK18] to general access structures.

Definition 8. *(**Non-Malleable Secret Sharing Schemes**) Let $\mathcal{A}$ be some access structure. Let $\mathcal{A}^{min}$ be its corresponding minimal basis access structure. Let $\mathcal{F}$ be some family of tampering functions. For each $f \in \mathcal{F}$, $m \in \mathcal{M}$ and authorized $T \in \mathcal{A}^{min}$, define the tampering experiment*

$$\mathbf{STamper_m^{f,T}} = \left\{ \begin{array}{c} shares \leftarrow Share(m) \\ \widetilde{shares} \leftarrow f(shares) \\ \tilde{m} \leftarrow Rec(\widetilde{shares}_T) \\ Output : \tilde{m} \end{array} \right\}$$

*which is a random variable over the randomness of the sharing function Share. We say that the (n, ϵ)-secret sharing scheme, (Share, Rec), realizing access structure $\mathcal{A}$ is ϵ'**-non-malleable** w.r.t $\mathcal{F}$ if for each $f \in \mathcal{F}$ and authorized $T \in \mathcal{A}^{min}$, there exists a distribution $SD^{f,T}$ (corresponding to the simulator) over $\mathcal{M} \cup \{same^*, \perp\}$ such that, for all $m \in \mathcal{M}$ and all authorized $T \in \mathcal{A}^{min}$, we have that the statistical distance between $STamper_m^{f,T}$ and*

$$\mathbf{SSim_m^{f,T}} = \left\{ \begin{array}{c} \tilde{m} \leftarrow SD^{f,T} \\ Output : m \; if \tilde{m} = same^*, or \; \tilde{m}, otherwise \end{array} \right\}$$

is at most ϵ'.

2.3 Threshold Access Structure $\mathcal{A}_n^t$

Apart from general access structure we will be interested in a special access structure which allows any t-out-of-n parties to pool their secret and reconstruct the secret. This threshold access structure can be formally represented as $\mathcal{A}_n^t = \{B \subseteq [n] : |B| \geq t\}$. We use the notation of (t, n, ϵ)-secret sharing sharing schemefor denoting (n, ϵ)-secret sharing scheme realizing access structure $\mathcal{A}_n^t$.

3 Non-Malleable Secret Sharing against Individual Tampering

In this section we show how to convert any secret sharing scheme into a non-malleable one against an adversary who arbitrarily tampers each of the shares independently. We begin recalling the tampering family from [GK18]:

Split-State Tampering Family $\mathcal{F}_n^{split}$:
Let **Share** be a sharing function that takes as input a message $m \in \mathcal{M}$ and outputs a shares $shares \in \otimes_{i \in [n]} \mathcal{S}_i$. Parse the output $shares$ into n blocks, namely $share_1, share_2, \ldots, share_n$ where each $share_i \in \mathcal{S}_i$. For each $i \in [n]$, let $\mathbf{f_i} : \mathcal{S}_i \rightarrow \mathcal{S}_i$ be an arbitrary tampering function, that takes as input $share_i$, the i^{th} share. Let $\mathcal{F}_n^{split}$ be a family of such n functions $(\mathbf{f_1}, \mathbf{f_2}, \ldots, \mathbf{f_n})$.

Note that above definition is written with respect to a sharing function. It is just for ease of presentation, we can use this family of tampering functions with respect to a coding scheme, by treating the encoding procedure as a sharing function. We also recall a lemma, which can be used to show that every 2 split-state non-malleable code is a 2-out-of-2 non-malleable secret sharing scheme.

Lemma 1. *([ADKO15]) Let* **Enc** $: \mathcal{M} \rightarrow \mathcal{C}^2$ *be the encoding function, and* **Dec** $: \mathcal{C}^2 \rightarrow \mathcal{M} \cup \{\perp\}$ *be a deterministic decoding function. If a coding scheme* $(\mathbf{Enc}, \mathbf{Dec})$ *is ϵ-non-malleable w.r.t $\mathcal{F}_2^{split}$ then* $(\mathbf{Enc}, \mathbf{Dec})$ *is also a $(2, 2\epsilon)$-secret sharing scheme that is ϵ-non-malleable w.r.t $\mathcal{F}_2^{split}$, where* **Enc** *acts as a sharing function.*

Access Structures Based Definitions. As our building blocks, we will use secret-sharing schemes that allow any authorized "pair" to reconstruct the secret. We formally define such "paired" access structures below, and construct these schemes in the full version.

Definition 9. *(**Paired Access Structures**) An access structure $\mathcal{A}$ is called a **paired access structure**, if each authorized set contains an authorized subset of size two. Formally, for all $B \in \mathcal{A}$, there exists a subset $C \subseteq B$ such that C is authorized and has cardinality two.*

Notice that, if $\mathcal{A}$ is a paired access structure then its corresponding minimal basis access structure $\mathcal{A}^{min}$ will only contain authorized sets of size two.

Definition 10. *(**Authorized Paired Access Structures**) For any access structure $\mathcal{A}$, we call a paired access structure $\mathcal{A}_{pairs}$ an **authorized paired access structure** corresponding to $\mathcal{A}$ if $\mathcal{A}_{pairs}$ is the maximal subcollection of $\mathcal{A}$. Formally,*

$$\mathcal{A}_{pairs} = \{B \in \mathcal{A} : \exists C \subseteq B, (C \in \mathcal{A}) \wedge (|C| = 2)\}$$

Notice that $\mathcal{A}^{min}_{pairs}$ will be equal to the set of all the authorized sets of size two in $\mathcal{A}$.

Leakage Family. We also use a 2-out-of-n leakage-resilient secret sharing scheme. While in [GK18] split state family of leakage-resilience was needed, we require leakage-resilience against the following stronger leakage family.

Leakage Family $\boldsymbol{\mathcal{L}^{pair}_{\mu}}$:
Let $(\mathbf{LRShare}, \mathbf{LRRec})$ be any $(2, n, \epsilon)$-secret sharing scheme with message space $\mathcal{M}$. For any $i, j \in [n]$, for each $k \in [n] \setminus \{i, j\}$, let f_k be an arbitrary function that takes $share_i$ as input and outputs μ bits of information about its input. For any collection of such functions, any pair of message $a^0, a^1 \in \mathcal{M}$, any independently chosen bit $b \in \{0, 1\}$, we define the leakage experiment as,

$$\mathbf{Leak}^{\mathbf{a^0,a^1}}_{\mathbf{b}} = \left\{ \begin{array}{c} a^0_1, \ldots, a^0_n \leftarrow \mathbf{LRShare}(a^0) \\ a^1_1, \ldots, a^1_n \leftarrow \mathbf{LRShare}(a^1) \\ Output : a^0_i, a^0_j, a^1_i, a^1_j, \otimes_{k \in [n] \setminus \{i,j\}} f_k(a^b_k) \end{array} \right\}$$

We say that the scheme $(\mathbf{LRShare}, \mathbf{LRRec})$ is ϵ-leakage-resilient w.r.t. $\mathcal{L}^{pair}_{\mu}$if for every pair of message $a^0, a^1 \in \mathcal{M}$, we have that

$$\mathbf{Leak}^{\mathbf{a^0,a^1}}_{\mathbf{0}} \approx_{\epsilon} \mathbf{Leak}^{\mathbf{a^0,a^1}}_{\mathbf{1}}$$

In full version, we prove that the construction of [GK18] is in fact leakage-resilient against $\mathcal{L}^{pair}_{\mu}$.

Building blocks. In our constructions for general access structure, we need a method to find a minimal authorized set, when given any authorized set. For any access structure $\mathcal{A}$ not containing singletons, we define a deterministic procedure $\mathbf{FindMinSet} : \mathcal{A} \rightarrow \mathcal{A}^{min}$, which takes an authorized set and outputs a minimal authorized set contained in that set. The description follows :

Procedure $\boldsymbol{FindMinSet^{\mathcal{A}}(S)}$:
On input an authorized set S for an access structure $\mathcal{A}$, if there exists an $i \in S$ and $j \in S$ such that $i \neq j$ and $\{i, j\} \in \mathcal{A}$, then return the lexicographical smallest pair $\{i, j\}$ satisfying these conditions, otherwise initialize $T \leftarrow D$ and execute the following loop : let T be an ordered set of t elements $i_1, i_2, \ldots, i_t$. For $j \in [t]$, check if $T \setminus \{i_j\}$ belongs to $\mathcal{A}$, in which case set $T \leftarrow T \setminus \{i_j\}$ and go the beginning of the loop. If no such j exists, then break from the loop and output T.

The runtime of the above procedure is $O(n^2)$, because in each step of the loop it removes one element from the set T, whose size is upper bounded by the number of parties n. Note that, we assumed a membership query oracle, which decides whether the given set is authorized or not.

Pruning Compiler. As a building block towards our generic compiler, we need another compiler that given any statistical (resp. computational) secret sharing scheme realizing any access structure, outputs another secret sharing scheme that deauthorizes all authorized pairs while preserving the underlying statistical/computational secrecy. That is, it additionally guarantees that any two shares perfectly hide the secret.

Lemma 2. *For any efficient statistical (resp. computational) secret sharing scheme* $(\mathbf{AShare}, \mathbf{ARec})$ *realizing access structure* $\mathcal{A}$ *that does not contain singletons, there exists another efficient statistical (resp. computational) secret sharing scheme* $(\mathbf{APShare}, \mathbf{APRec})$ *which satisfies the following properties.*

1. $(\mathbf{APShare}, \mathbf{APRec})$ *realizes the access structure* $\mathcal{A}$ *with authorized pairs removed. The statistical error remains the same if the input is a statistical scheme.*
2. $(\mathbf{APShare}, \mathbf{APRec})$ *ensures that given any two shares, the secret is perfectly hidden.*

Proof. We give the construction of $(\mathbf{APShare}, \mathbf{APRec})$, deferring the proof to full version. Let n be the number of parties, and $\mathbb{F}$ be the secret space. Let $\mathbf{AShare}$ share an element of $\mathbb{F}$ into n elements of field $\mathbb{F}_1$. Let $(\mathbf{TShare^3_n}, \mathbf{TRec^3_n})$ and $(\mathbf{TShare^2_2}, \mathbf{TRec^2_2})$ be two threshold secret sharing scheme instantiated with Shamir's Secret Sharing scheme [Sha79] mapping an element of $\mathbb{F}_1$ into shares in $\mathbb{F}_1$ having theshold 2 and 3 respectively.

- **Sharing function APShare**. On input $m \in F$, share m using $\mathbf{AShare}$ to obtain $m_1, \ldots, m_n \leftarrow \mathbf{AShare}(m)$. For each $i \in [n]$, share m_i using $\mathbf{TShare^2_2}$ to obtain $l_i, r_i \leftarrow \mathbf{TShare^2_2}(m_i)$ and share r_i using $\mathbf{TShare^3_n}$ to obtain $r_i^1, \ldots, r_i^n \leftarrow \mathbf{TShare^3_n}\ (r_i)$. For each $i \in [n]$ construct $share_i$ as l_i, $r_1^i, \ldots, r_n^i$. Output $share_1, \ldots, share_n$.

- **Reconstruction Function APRec**. On input the shares $\otimes_{i \in T} share_i$ corresponding to authorized set $T \in \mathcal{A}$ with $|T| \geq 3$, for each $i \in T$, parse $share_i$ as $l_i, r_1^i, \ldots, r_n^i$. For each $i \in [n]$, reconstruct $r_i \leftarrow \mathbf{TRec^3_n}\ (\otimes_{i \in T} r_i)$. For each $i \in T$, reconstruct $m_i \leftarrow \mathbf{TRec^2_2}(l_i, r_i)$. Reconstruct $m \leftarrow \mathbf{ARec}(\otimes_{i \in T} m_i)$. Output m.

As our compiler also works with computational schemes, we first define them. Please refer to the book by Goldreich [Gol07] for definition of computaional indistinguishability.

Definition 11. *(**Computational Secret Sharing**). Let $\mathcal{M}$ be a finite set of secrets, where $|\mathcal{M}| \geq 2$. An efficient sharing function Share with domain of secrets $\mathcal{M}$ is a **Computational Secret Sharing Scheme** realizing an access structure $\mathcal{A}$ if the following two properties hold :*

1. ***Correctness**. The secret m can be reconstructed by any authorized set of parties. That is, for any set $B \in \mathcal{A}$(where $B = \{p_{i_1}, \ldots, p_{i_{|B|}}\}$), there exists an efficient deterministic reconstruction function Reconstruct^B : $S_{i_1} \times S_{i_2} \times \ldots S_{i_{|B|}} \rightarrow \mathcal{M}$ such that for every $m \in \mathcal{M}$,*

$$Pr[Reconstruct^B(Share(m)_B) = m] = 1$$

(over the randomness of the Sharing function)

2. ***Computational Privacy**. An unauthorized set of parties should be unable to distinguish whether the hidden secret is m_0 or m_1 for all $m_0, m_1 \in \mathcal{M}$. More formally, for any set $T \notin \mathcal{A}$, for every two secrets $a, b \in \mathcal{M}$, any PPT adversary should not be able to distinguish between,*

$$Share(a)_T \approx Share(b)_T$$

where the two distributions are computationally indistinguishable.

Main Result for General Access Structures. We are now in position to give our main result.

Theorem 3. *For any number of parties n, and any access structure $\mathcal{A}$ that does not contain singletons. If we have the following primitives :*

1. *For any $\epsilon_1 \geq 0$, let $(\mathbf{NMEnc}, \mathbf{NMDec})$ be any coding scheme that is ϵ_1-non-malleable wrt $\mathcal{F}_2^{split}$, which encodes an element of the set $\mathbb{F}_0$ into two elements of the field $\mathbb{F}_1$.*
2. *For any $\epsilon_2 \geq 0$, let $(\mathbf{AShare}, \mathbf{ARec})$ be any (n, ϵ_2)-secret sharing scheme (resp. computational) realizing access structure $\mathcal{A}$, which shares an element of field $\mathbb{F}_1$ into n elements of the field $\mathbb{F}_2$.*
3. *Let $\mu \leftarrow \log |\mathbb{F}_2|$. For any $\epsilon_3 \geq 0$, let $(\mathbf{LRShare}, \mathbf{LRRec})$, be any $(2, n, \epsilon_3)$-secret sharing scheme that is ϵ_3-leakage-resilient w.r.t. $\mathcal{L}_\mu^{pair}$, which shares an element of the field $\mathbb{F}_1$ into n elements of the field $\mathbb{F}_3$.*
4. *For any $\epsilon_4 \geq 0$, let $(\mathbf{PNMShare}, \mathbf{PNMRec})$, be any (n, ϵ_4)-secret sharing scheme realizing the authorized paired access structure $\mathcal{A}_{pairs}$ that is ϵ_4-non-malleable wrt $\mathcal{F}_n^{split}$, which shares an element of the set $\mathbb{F}_0$ into n elements of the field $\mathbb{F}_4$.*

then there exists $(n, 2\epsilon_1 + \epsilon_2 + \epsilon_4)$-secret sharing scheme (resp. computational) realizing access structure $\mathcal{A}$ that is $(2\epsilon_1 + \epsilon_2 + \epsilon_3 + \epsilon_4)$-non-malleable w.r.t $\mathcal{F}_n^{split}$. The resulting scheme, $(\mathbf{NMShare}, \mathbf{NMRec})$, shares an element of the set $\mathbb{F}_0$ into n shares where each share is an element of $(\mathbb{F}_2 \times \mathbb{F}_3 \times \mathbb{F}_4)$. Further, if the *four primitives have efficient construction (polynomial time sharing and reconstruction functions), then the constructed scheme is also efficient.*

Proof. We begin with the construction of the desired non-malleable secret sharing scheme. Apply Lemma 2 to the computational secret sharing scheme (**AShare**, **ARec**) to obtain a pruned secret sharing scheme (**APShare**, **APRec**) .

- **Sharing function NMShare**: Encode the secret input $m \in \mathbb{F}_1$ using the encoding function of the non-malleable code. Let $l, r \leftarrow \mathbf{NMEnc}(m)$. Share l using a **APShare** to obtain $l_1, \ldots, l_n \leftarrow \mathbf{APShare}(l)$. Share r using a 2-out-of-n leakage-resilient secret sharing scheme. Let $r_1, \ldots, r_n \leftarrow \mathbf{LRRec}(r)$. Use the sharing procedure **PNMShare** to share m. Let $(p_1, \ldots, p_n) \leftarrow \mathbf{PNMShare(m)}$. Then for each $i \in [n]$, construct $share_i$ as l_i, r_i, p_i.

- **Reconstruction function NMRec**: On input the shares $\otimes_{i \in D} share_i$ corresponding to authorized set D, for each $i \in D$, parse $share_i$ as (l_i, r_i, p_i). Find the minimal authorized set $T \in \mathcal{A}^{min}$ by running the procedure **findMinSet** with input D. Let T be a set containing t indices $\{i_1, i_2, \ldots, i_t\}$ such that $i_j < i_{j+1}$ for each $j \in [t-1]$. If $D \in \mathcal{A}_{pairs}$, use the decoding procedure $\mathbf{PNMRec}^{\{\mathbf{i_1},\mathbf{i_2}\}}$ to obtain the hidden secret $m \leftarrow \mathbf{PNMRec}^{\{\mathbf{i_1},\mathbf{i_2}\}}(p_{i_1}, p_{i_2})$. Otherwise, run the reconstruction procedure **APRec** on t shares of l, to obtain $l \leftarrow \mathbf{APRec}(\otimes_{i \in T} l_i)$. Run the reconstruction procedure of the leakage-resilient secret sharing scheme on the first 2 shares of r, to obtain $r \leftarrow \mathbf{LRRec}^{\{\mathbf{i_1},\mathbf{i_2}\}}(r_{i_1}, r_{i_2})$. Decode l and r using decoding process of underlying non-malleable code to obtain : $m \leftarrow \mathbf{NMDec}(\mathbf{l}, \mathbf{r})$. Output m.

Correctness and Efficiency : Trivially follows from the construction.

Statistical (resp. Computational Privacy) : We prove statistical privacy using hybrid argument. For ease of understanding, let $share_i$ be of the form al_i, ar_i, ap_i when the secret a is encoded by the sharing procedure **NMShare**. Similarly, let $share_i$ be of the form bl_i, br_i, bp_i when the secret b is encoded. Let T be an unauthorized set containing t indices $\{i_1, i_2, \ldots, i_t\}$ such that $i_j < i_{j+1}$ for each $j \in [t-1]$. We describe the hybrids below :

1. $\mathbf{Hybrid_1}$: for each $i \in T$, $share_i$ is of the form al_i, ar_i, ap_i. The distribution of these t shares is identical to distribution obtained on running the **NMShare** on input a. Output $\otimes_{i \in T} share_i$.
2. $\mathbf{Hybrid_2}$: Sample the shares as in $\mathbf{Hybrid_1}$, the previous hybrid. For each $i \in T$, replace al_i with bl_i to obtain share of the form bl_i, ar_i, ap_i. Output $\otimes_{i \in T} share_i$.
3. $\mathbf{Hybrid_3}$: Sample the shares as in $\mathbf{Hybrid_2}$, the previous hybrid. For each $i \in T$, replace ar_i with br_i to obtain share of the form bl_i, br_i, ap_i. Output $\otimes_{i \in T} share_i$.
4. $\mathbf{Hybrid_4}$: Sample the shares as in $\mathbf{Hybrid_3}$, the previous hybrid. For each $i \in T$, replace ap_i with bp_i to obtain share of the form bl_i, br_i, bp_i. Output $\otimes_{i \in T} share_i$. The distribution of these t shares is identical to distribution obtained on running the **NMShare** on input b. Output $\otimes_{i \in T} share_i$.

Claim: For any pair of secrets $a, b \in F_0$, any unauthorized $T \notin \mathcal{A}$, the statistical distance between $\mathbf{Hybrid_1}$ and $\mathbf{Hybrid_2}$ is at most ϵ_2 (resp. $\mathbf{Hybrid_1}$ and $\mathbf{Hybrid_2}$ are computationally indistinguishable).
Proof: The two hybrids only differ in the shares of l. As T is unauthorized in $\mathcal{A}$, the claim follows from the statistical (resp. computational) privacy of the secret scheme $(\mathbf{AShare}, \mathbf{ARec})$. ■

Claim: For any pair of secrets $a, b \in F_0$, any unauthorized $T \notin \mathcal{A}$, the statistical distance between $\mathbf{Hybrid_2}$ and $\mathbf{Hybrid_3}$ is at most $2\epsilon_1$.
Proof: As in [GK18], the two hybrids are statistically indistinguishable by the $(2, 2\epsilon_1)$-secrecy satisfied by the non-malleable code $(\mathbf{NMEnc}, \mathbf{NMDec})$ (as in lemma 1), by utilizing that fact knowing only r reveals nothing about the underlying message m. ■

Claim: For any pair of secrets $a, b \in F_0$, any unauthorized $T \notin \mathcal{A}$, the statistical distance between $\mathbf{Hybrid_3}$ and $\mathbf{Hybrid_4}$ is at most ϵ_4.
Proof: $T \notin \mathcal{A}$, implies that $T \notin \mathcal{A}_{pairs}$. The two hybrids only differ in the shares corresponding to output of **PNMShare**. The claim follows from the statistical privacy of $(\mathbf{PNMShare}, \mathbf{PNMRec})$. ■

By repeated application of triangle inequality, we get that for any $a, b \in F_0$, any unauthorized $T \notin \mathcal{A}$, the statistical distance between $\mathbf{Hybrid_1}$ and $\mathbf{Hybrid_4}$ is at most $2\epsilon_1 + \epsilon_2 + \epsilon_4$ (resp. the hybrids $\mathbf{Hybrid_1}$ and $\mathbf{Hybrid_4}$ are computationally indistinguishable). This proves the statistical (resp. computational) privacy of our scheme.

Statistical Non Malleability : To prove non-malleability of the current secret sharing scheme, we give a simulator for every admissible tampering attack on our scheme by using the simulator of the underlying non-malleable code after we have given an equivalent split-state tampering attack.

Let us begin with the intuition for the procedure **FindMinSet**. Notice that for general access structures, it is possible that the given authorized set has an authorized subset of size two, and another disjoint (minimal) authorized set of size three. Moreover, in our construction different schemes are being used to encode for these subsets. In case our output depends on all these five shares, we cannot hope to achieve a reduction to the underlying non-malleable code (because by definition, non-malleability holds only when the adversary is given one encoding of the message, and it tampers to produce only one encoding. In the present case it gets two encodings of the same message). We solve such an issue by giving the procedure **FindMinSet** in subsection 3, which prunes the given authorized set efficiently and ensures that no proper subset of the output (minimal) authorized set is authorized. It is easy to see that this procedure needs to be deterministic for us to be able to argue that share reconstructed in real experiment is equal to the one in reduction. Given this observation, without loss of generality we can assume that adversary chooses an authorized set $T \in \mathcal{A}^{min}$

to be used for reconstruction of the secret, as otherwise we can use the function **FindMinSet** to compute $T \in \mathcal{A}^{min}$ from any $D \in \mathcal{A}$. As the adversary belongs to $\mathcal{F}_n^{split}$, it also specifies a set of n tampering functions $\{\mathbf{f_i} : i \in [n]\}$. All these functions act on their respective shares independently of the other shares, i.e. every $\mathbf{f_i}$ takes $share_i$ as input and outputs the tampered $\widetilde{share_i}$. We can also assume without loss of generality that all these tampering functions are deterministic, as the computationally unbounded adversary can compute the optimal randomness. Unlike [GK18], depending on the cardinality of T, we use these tampering functions to create explicit split-state function to tamper with either non-malleable code or paired non-malleable secret-sharing. :

Case 1 ($|T| = 2$) :
Let i_1 and i_2 be the two indices of T such that $i_1 < i_2$. In this case, we use the tampering functions $\mathbf{f_{i_1}}$ and $\mathbf{f_{i_2}}$ for the scheme $(\mathbf{NMShare}, \mathbf{NMRec})$ to create explicit tampering functions $\mathbf{F}_{i_1}$ and $\mathbf{F}_{i_2}$ for the underlying scheme $(\mathbf{PNMShare}, \mathbf{PNMRec})$. The reduction is described below :

1. **(Initial Setup)** : Randomly choose a message $m_\$ \in \mathcal{M}$, and run the sharing function **NMShare** with input $m_\$$ to obtain temporary shares. That is, $(tShare_1, \ldots, tShare_n) \leftarrow \mathbf{NMShare}(m_\$)$. For each $i \in [n]$, parse $tShare_i$ as tl_i, tr_i, tp_i.
2. The **tampering function** $\mathbf{F}_{i_1}$ is defined as follows : On input $p_{i_1} \in \mathbb{F}_4$, replace tp_{i_1} by p_{i_1} in $tShare_{i_1}$ to obtain $share_{i_1}$. Run $\mathbf{f_{i_1}}$ on $share_{i_1}$ to obtain $\widetilde{share_{i_1}}$. Parse $\widetilde{share_{i_1}}$ as $\widetilde{l_{i_1}}, \widetilde{r_{i_1}}, \widetilde{p_{i_1}}$. Output $\widetilde{p_{i_1}}$.
3. The **tampering function** $\mathbf{F}_{i_2}$ is defined as follows : On input $p_{i_2} \in \mathbb{F}_4$, replace tp_{i_2} by p_{i_2} in $tShare_{i_2}$ to obtain $share_{i_2}$. Run $\mathbf{f_{i_2}}$ on $share_{i_2}$ to obtain $\widetilde{share_{i_2}}$. Parse $\widetilde{share_{i_2}}$ as $\widetilde{l_{i_2}}, \widetilde{r_{i_2}}, \widetilde{p_{i_2}}$. Output $\widetilde{p_{i_2}}$.

The functions $\mathbf{F}_{i_1}$ and $\mathbf{F}_{i_2}$ have been defined in this way to ensure that the secret hidden by the shares l_{i_1} and l_{i_2} of the scheme $(\mathbf{PNMShare}, \mathbf{PNMRec})$ is the same as the secret hidden by $share_{i_1}$ and $share_{i_2}$ of the scheme (**NMShare**, **NMRec**). We also need to argue that the reduction generates $share_{i_1}$ and $share_2$ from the right distribution, as otherwise the functions $\mathbf{f_{i_1}}$ and $\mathbf{f_{i_2}}$ may detect the change in distribution and stop working. Similar to the proof of statistical privacy, we can use hybrid argument to show that, for any p_{i_1} and p_{i_2} encoding message $m \leftarrow \mathbf{PNMRec}^{\{i_1,i_2\}}(p_{i_1}, p_{i_2})$, the statistical distance between the distribution of $share_{i_1}, share_{i_2}$ generated while executing $\mathbf{NMShare}(m)$ and the two shares generated by the reduction is at most $2\epsilon_1$. We rely on 2-out-of-2 secrecy property satisfied by non-malleable codes to show that even after learning r from the two shares, we learn nothing about the underlying secret. We also relied on the fact that two shares of l reveal nothing about l by the property of the pruning compiler (as in lemma 2). Note that here we relied on the pruning compiler to ensure that any authorized pair will only get the encoding of the message under the pair-wise scheme $(\mathbf{PNMShare}, \mathbf{PNMRec})$ and not the other scheme.

For all $i \in [n] \setminus \{i_1, i_2\}$, let $\mathbf{F}_i$ be the identity function. The created set of functions $\{F_i : i \in [n]\}$ belongs to $\mathcal{F}_n^{split}$. Therefore, the tampering experiments of the two non-malleable secret-sharing scheme (see definition 8) are statistically indistinguishable, specifically,

$$\mathbf{STamper}_{\mathbf{m}}^{\mathbf{f},\mathbf{T}} \approx_{2\epsilon_1} \mathbf{STamper}_{\mathbf{m}}^{\mathbf{F},\mathbf{T}}$$

By the ϵ_4-non malleability of the scheme $(\mathbf{PNMShare}, \mathbf{PNMRec})$, there exists a simulator $\mathbf{SSim}_{\mathbf{m}}^{\mathbf{F},\mathbf{T}}$ such that $\mathbf{STamper}_{\mathbf{m}}^{\mathbf{F},\mathbf{T}} \approx_{\epsilon_4} \mathbf{SSim}_{\mathbf{m}}^{\mathbf{F},\mathbf{T}}$. We use the underlying simulator as our simulator, and let $\mathbf{SSim}_{\mathbf{m}}^{\mathbf{f},\mathbf{T}} \equiv \mathbf{SSim}_{\mathbf{m}}^{\mathbf{F},\mathbf{T}}$. Applying triangle inequality to the above relations we prove the statistical non malleability for this case.

$$\mathbf{STamper}_{\mathbf{m}}^{\mathbf{f},\mathbf{T}} \approx_{2\epsilon_1+\epsilon_4} \mathbf{SSim}_{\mathbf{m}}^{\mathbf{f},\mathbf{T}}$$

Case 2 ($|T| \geq 3$) :
Let $T = \{i_1, i_2 \dots i_t\}$ be an ordered set of t indices, such that $i_j < i_{j+1}$. In this case, we use the tampering functions $\{\mathbf{f_i} : i \in T\}$ that tamper the shares of the scheme $(\mathbf{NMShare}, \mathbf{NMRec})$ to create explicit tampering functions $\mathbf{F}$ and $\mathbf{G}$ which tamper the two parts of non-malleable code. Note that as $\mathcal{F}_2^{split}$ allows arbitrary computation, the functions $\mathbf{F}$ and $\mathbf{G}$ are allowed to brute force over any finite subset. The reduction giving explicit $(\mathbf{F}, \mathbf{G}) \in \mathcal{F}_2^{split}$ is described below.

1. **(Initial Setup)** : Fix an arbitrary $m_\$$ and let $l_\$, r_\$ \leftarrow \mathbf{NMEnc}(m_\$)$. Run the sharing function **APShare** with input $l_\$$ to obtain $\otimes_{i\in[n]} tl_i$. Run the sharing function $\mathbf{LRShare}_n^2(r_\$)$ to obtain $\otimes_{i\in[n]} tr_i$. Run the sharing function $\mathbf{PNMShare}(m_\$)$ to obtain $\otimes_{i\in[n]} tp_i$. For each $i \in [n]$, create $tshare_i$ as tl_i, tr_i, tp_i. For all $i \in T$, **fix** $p_i \leftarrow tp_i$. For each $i \in \{i_1, i_2\}$, run f_i on $tShare_i$ to obtain $\widetilde{tShare_i} \leftarrow f_i(tShare_i)$. Parse $\widetilde{tshare_i}$ as $\widetilde{tl_i}, \widetilde{tr_i}, \widetilde{tp_i}$. **Fix** $l_i \leftarrow tl_i$ and $\widetilde{l_i} \leftarrow \widetilde{tl_i}$. For $i \in \{i_3, \dots, i_t\}$, fix $r_i \leftarrow tr_i$. (Note that, here we rely on our pruning compiler for a different purpose: fixing l_{i_1}, l_{i_2} is allowed by property 2 of lemma 2. We would not have been able to do the same with a computational secret sharing directly. Also note that we depart significantly from initial step of [GK18], where $t-1$ shares of l and only the last share of r was fixed. This was allowed because any $t-1$ shares (resp. one share) does not reveal anything about the underlying l (resp. r). We on the other hand have fixed $t-2$ shares of r, which encode a random value of $r_\$$).
2. The **tampering function F** is defined as follows : On input l, sample the value of $l_{i_3}, \dots, l_{i_t}$ such that the shares $\{l_i : i \in T\}$ hide the secret l under $(\mathbf{APShare}, \mathbf{APRec})$ and the distribution of sampled l_{i_t} is identical to the distribution produced on running **APShare** with input l conditioned on fixing $\{l_i : i \in \{i_1, i_2\}\}$. In case such a sampling is not possible, then `abort`. Otherwise, for each $i \in T \setminus \{i_1, i_2\}$, construct $share_i$ as l_i, r_i, p_i using the fixed values of r_i and p_i. Run the tampering function f_i on $share_i$ to obtain tampered $\widetilde{share_i}$. Parse $\widetilde{share_i}$ as $\widetilde{l_i}, \widetilde{r_i}, \widetilde{p_i}$. Run the reconstruction function **APRec** with input $\otimes_{i\in T}\widetilde{l_i}$ to obtain $\widetilde{l}$. Output $\widetilde{l}$. (Note that unlike [GK18] we invoked the tampering functions with 'incorrect' shares of r).

3. The **tampering function G** is defined as follows : On input r, sample the values of first two shares of r, namely $\{r_{i_1}, r_{i_2}\}$ satisfying the following constraints :-
   - The two shares $\{r_{i_1}, r_{i_2}\}$ encode the secret r under the (**LRShare**,**LRRec**). Moreover, the two shares should be distributed according to the output distribution of scheme (**LRShare**,**LRRec**).
   - For each $i \in \{i_1, i_2\}$, let $Share_i$ be l_i, r_i, p_i, run f_i on $share_i$ to obtain $\widetilde{share_i}$. Parse $\widetilde{share_i}$ as $\widetilde{nl_i}, \widetilde{nr_i}, \widetilde{np_i}$. The value of $\widetilde{nl_i}$ should be equal to $\widetilde{l_i}$ (the value that was fixed in the initial step of reduction). This can be achieved via brute force over the all the possibilities.

   In case such a sampling is not possible, then `abort`. Otherwise, run the reconstruction procedure of the leakage-resilient scheme to obtain $\widetilde{r}$, using the tampered values of first 2 shares of r. That is $\widetilde{r} \leftarrow \mathbf{LRRec}^{\{\mathbf{i_1},\mathbf{i_2}\}}(\widetilde{nr_{i_1}}, \widetilde{nr_{i_2}})$. Output $\widetilde{r}$. (Unlike [GK18], we now only ensure that the first two shares are from the correct distribution.)

The reduction given above creates t shares corresponding to indices in T. Unlike the proof of [GK18], here the distribution of the t shares is not close to the distribution of the t shares during actual sharing (in fact statistically it is quite far). Nevertheless, we show that an adversary cannot notice this change without violating the leakage resilience of the (**LRShare**,**LRRec**).

We achieve this using hybrid argument, however, instead of outputting t shares $\otimes_{i \in T} share_i$ as in [GK18], we output $\mathbf{NMRec}(\otimes_{i \in T} f_i(share_i))$, the output of the tampering experiment. For ease of understanding, let $share_i$ be of the form al_i, ar_i, ap_i when the shares are produced by the reduction on input l and r, with the fixing of $l_\$$ and $r_\$$. Similarly, let $share_i$ be of the form bl_i, br_i, bp_i when the secret m is encoded by the sharing procedure **NMShare** conditioned on output of $\mathbf{NMEnc}(m)$ being l, r.

1. $\mathbf{Hybrid_1}$: for each $i \in T$, $share_i$ is of the form al_i, ar_i, ap_i. The distribution of these t shares is identical to distribution of the shares produced by the reduction on input l and r, with the fixing of $l_\$$ and $r_\$$. Output $\mathbf{NMRec}(\otimes_{i \in T} f_i(share_i))$.
2. $\mathbf{Hybrid_2}$: In the initial setup phase of the reduction, for each $i \in T$, fix bp_i instead of ap_i. Proceed with the reduction to create t shares of the form al_i, ar_i, bp_i. Output $\mathbf{NMRec}(\otimes_{i \in T} f_i(share_i))$.
3. $\mathbf{Hybrid_3}$: Fix $l_\$ \leftarrow l$ in the initial setup phase. Fix shares of p like $\mathbf{Hybrid_2}$. Output $\mathbf{NMRec}(\otimes_{i \in T} f_i(share_i))$.
4. $\mathbf{Hybrid_4}$: Fix $l_\$ \leftarrow l$ and fix $r_\$ \leftarrow r$ in the initial setup phase. Fix the shares of p as in previous hybrid $\mathbf{Hybrid_3}$. Proceed with the reduction to create the t shares. Output $\mathbf{NMRec}(\otimes_{i \in T} f_i(share_i))$.
5. $\mathbf{Hybrid_5}$: For each $i \in [n]$, let $share_i$ be of the form bl_i, br_i, bp_i. The distribution of these t shares is identical to distribution obtained on running the **NMShare** conditioned on output of $\mathbf{NMEnc}(m)$ being l, r. Output $\mathbf{NMRec}(\otimes_{i \in T} f_i(share_i))$.

Claim: For any authorized $T \in \mathcal{A}^{min}$ with cardinality greater than 2, the statistical distance between $\mathbf{Hybrid_1}$ and $\mathbf{Hybrid_2}$ is at most ϵ_4.
Proof: As $|T| \geq 3$, T does not belong to $\mathcal{A}_{pairs}$. The two hybrids only differ in the shares corresponding to output of **PNMShare**. The claim follows from the statistical privacy of (**PNMShare**, **PNMRec**). ■

Claim: For any $l, l_\$$, any authorized $T \in \mathcal{A}^{min}$, $\mathbf{Hybrid_2}$ is identical to $\mathbf{Hybrid_3}$.

Proof: The two hybrids differ in the intial setup phase. In $\mathbf{Hybrid_2}$, 2 shares of $l_\$$ are fixed, while in $\mathbf{Hybrid_3}$ 2 shares of l are fixed. Lemma 2 ensures that the secret is perfectly hidden even when two shares of **APShare** are revealed. ■

The above also shows that the function **F** in the reduction never aborts.
Claim: For any $r, r_\$$, any authorized $T \in \mathcal{A}^{min}$ with cardinality greater than 2, the statistical distance between $\mathbf{Hybrid_3}$ and $\mathbf{Hybrid_4}$ is at most ϵ_3.
Proof: Assume towards contradiction that there exists $r, r_\$ \in \mathbb{F}_1$, $T \in \mathcal{A}^{min}$ and a distinguisher D that is successful in distinguishing $\mathbf{Hybrid_3}$ and $\mathbf{Hybrid_4}$ with probability greater than ϵ_3. We use distinguisher D to construct another distinguisher D_1 and a leak function $g \in \mathcal{L}_\mu^{pair}$which violates the property of leakage-resilience satisfied by the scheme ($\mathbf{LRShare_n^2}$, $\mathbf{LRRec_n^2}$) for the secrets $r, r_\$$. The reduction is described below :

1. (**Initial Setup**) : Run the sharing function **APRec** with input l to obtain $\otimes_{i\in[n]} tl_i$. Run the sharing function **PNMShare**(m) to obtain $\otimes_{i\in[n]} tp_i$. For all $i \in T$, **fix** $p_i \leftarrow tp_i$ and $l_i \leftarrow tl_i$.
   Give $r, r_\$$ to the adversary, who then specifies $r_1, r_2, r_1^\$, r_2^\$$. Use l_1, r_1, p_1 and l_2, r_2, p_2 to create the first two shares $share_1$ and $share_2$. Tamper the shares using f_1 and f_2 to obtain $\tilde{l_1}, \tilde{r_1}, \tilde{p_1}$ and $\tilde{l_2}, \tilde{r_2}, \tilde{p_2}$. Compute $\tilde{r} \leftarrow$ **LRRec**$(\tilde{r_1}, \tilde{r_2})$. Fix $\tilde{l_1}, \tilde{l_2}$.
2. (**Leak function g**) : We define a specific leakage function $g = \{g_i : i \in T \setminus \{i_1, i_2\}\}$ which leaks μ bits independently from each of the $t-2$ shares.
   - For each $i \in T \setminus \{i_1, i_2\}$, define g_i as the following function which takes r_i as input. Create $tShare_i$ as l_i, r_i, p_i. Run f_i on $tShare_i$ to obtain $\widetilde{tShare_i} \leftarrow f_i(tShare_i)$. Parse $\widetilde{tshare_i}$ as $\widetilde{tl_i}, \widetilde{tr_i}, \widetilde{tp_i}$. Output $\widetilde{tl_i}$.

   As $\widetilde{tl_i}$ is an element of $\mathbb{F}_2$, it can be represented by at most $\log |\mathbb{F}_2|$ bits, which is equal to μ. This shows that the above leak function g belongs to the class $\mathcal{L}_\mu^{pair}$.

3. (**Distinguisher** D_1) : The distinguisher D_1 is defined as follows : On input $g(r_3, \ldots, r_t)$, parse it as $\widetilde{tl_{i_3}}, \ldots, \widetilde{tl_{i_t}}$. Compute $\tilde{l} \leftarrow$ **APRec**$(\tilde{l_1}, \ldots, \tilde{l_t})$. Compute $\tilde{m} \leftarrow$ **NMDec**$(\tilde{l}, \tilde{r})$. Invoke the distinguisher D with $\tilde{m}$ and output its output.

Notice, in the case the secret hidden by the leakage-resilient scheme was $r_\$$, D will be invoked with input distributed according to $\mathbf{Hybrid_2}$. In the other case, in which r was hidden, D will be invoked with distributed according to $\mathbf{Hybrid_3}$. Therefore the success probability of D_1 will be equal to the advantage of D in distinguishing these two hybrids, which is greater than ϵ_3 by assumption. Hence, we have arrived at a contradiction to statistical leakage-resilience property of the scheme (**LRShare**,**LRRec**). ■

The above also shows that the function **G** in the reduction aborts with probability less than ϵ_3.

Claim: For any l, r, $\mathbf{Hybrid_4}$ is identical to $\mathbf{Hybrid_5}$.

Proof: In $\mathbf{Hybrid_4}$, the shares of $r_\$$ (resp. $l_\$$) that are sampled in the initial setup already encode the value r (resp. l). Therefore, all the t shares created in $\mathbf{Hybrid_4}$ will be identically distributed to the ones produced while executing **NMShare** with the output of **NMEnc** being (l, r). ■

By repeated application of triangle inequality, we get that for any $a, b \in \mathbb{F}_0$, the statistical distance between $\mathbf{Hybrid_1}$ and $\mathbf{Hybrid_5}$ is at most $\epsilon_2 + \epsilon_3 + \epsilon_4$. This proves that the set of shares created by our reduction is statistically close the set of shares created during the real sharing by the scheme, and thus the tampering functions $\mathbf{f} = \{\mathbf{f_i} : i \in T\}$ can be successfully invoked.

From our construction of **F** and **G**, it is clear that for any l and r, if the reduction is successful in creating the t shares, then the secret hidden is these t shares is the same as the message encoded by l and r (under non-malleable code). That is,

$$\mathbf{NMRec}(\{share_i : i \in T\}) = \mathbf{NMDec}(l, r)$$

Similarly, we can say that the secret hidden is the t tampered shares is the same as the message encoded by tampered $\tilde{l}$ and tampered $\tilde{r}$. That is,

$$\mathbf{NMRec}(\{\mathbf{f_i}(share_i) : i \in T\}) = \mathbf{NMDec}(\mathbf{F}(l), \mathbf{G}(r))$$

Therefore, the tampering experiments of non-malleable codes (see definition 4) and non-malleable secret-sharing schemes (see definition 8) are statistically indistinguishable, specifically,

$$\mathbf{STamper_m^{f,T}} \approx_{\epsilon_2+\epsilon_3+\epsilon_4} \mathbf{Tamper_m^{F,G}}$$

By the ϵ_1-non malleability of the scheme $(\mathbf{NMEnc}, \mathbf{NMDec})$, there exists a simulator $\mathbf{Sim_m^{F,G}}$ such that $\mathbf{Tamper_m^{F,G}} \approx_{\epsilon_1} \mathbf{Sim_m^{F,G}}$. We use the underlying simulator as our simulator and let $\mathbf{SSim_m^{f,T}} \equiv \mathbf{Sim_m^{F,G}}$. Applying triangle inequality to the above relations we prove the statistical non malleability.

$$\mathbf{STamper_m^{f,T}} \approx_{\epsilon_1+\epsilon_2+\epsilon_3+\epsilon_4} \mathbf{SSim_m^{f,T}}$$

As the the statistical distances between $\mathbf{STamper_m^{f,T}}$ and $\mathbf{SSim_m^{f,T}}$ in the two cases are $(2\epsilon_1 + \epsilon_4)$ and $(\epsilon_1 + \epsilon_2 + \epsilon_3 + \epsilon_4)$, we take $(2\epsilon_1 + \epsilon_2 + \epsilon_3 + \epsilon_4)$ as the worst case statistical error of our scheme $(\mathbf{NMShare}, \mathbf{NMRec})$.

□

4 n-out-of-n NMSS against Joint Tampering

Tampering family. We now formally define the supported tampering family, in which, we allow the tampered value of each share to depend on all the n shares in a restricted fashion.

Tampering Family $\boldsymbol{\mathcal{F}_n^{general}}$:
Assume that input shares are of equal length vectors over some finite field of prime order. The adversary specifies four subsets of $[n]$, namely $B_f^{in}, B_f^{out}, B_g^{in}$, B_g^{out} and also specifies four arbitrary tampering functions f_1, g_1, f_2, g_2 such that

$$f_1 : \{share_i : i \in B_f^{in}\} \rightarrow \{\widetilde{fshare}_i : i \in B_f^{out}\}$$
$$g_1 : \{share_i : i \in B_g^{in}\} \rightarrow \{\widetilde{gshare}_i : i \in B_f^{out}\}$$
$$f_2 : \{share_i : i \in B_f^{in}\} \rightarrow \{\widetilde{fshare}_i : i \in B_g^{out}\}$$
$$g_2 : \{share_i : i \in B_g^{in}\} \rightarrow \{\widetilde{gshare}_i : i \in B_g^{out}\}$$

such that for all $i \in [n]$, the final tampered share is of the form

$$\widetilde{share_i} \leftarrow \widetilde{fshare}_i \odot \widetilde{gshare}_i$$

where $\odot$ represents element wise multiplication of the two vectors over the given finite field. Here $B_f^{in} \subset [n]$ denotes the set of identities of parties whose shares are available as input to function f_1 and f_2. Similarly, B_f^{out} denotes the set of identities of parties whose tampered shares are produced by functions f_1 and g_1. B_g^{in} and B_f^{out} are analogous. The four subsets can be arbitrarily chosen by the adversary as long as they satisfy the following natural constraints :

- The input to tampering function f_1 contains atleast one share, which does not occur as the input of the tampering function g_1 and vice versa. That is, $|B_f^{in} \setminus B_g^{in}| \geq 1$ and $|B_g^{in} \setminus B_f^{in}| \geq 1$.
- The output sets B_f^{out} and B_g^{out} are disjoint. For the sake of simplicity, we further assume w.l.o.g that $B_f^{out} \cup B_g^{out} = [n]$.

Construction of [ADL14]. As we use the construction of Aggarwal et al. [ADL14] in a non-black-box way, we recall it for convenience :

Definition 12. *([ADL14]) **Affine Evasive Function** : A surjective function $h : F_p \rightarrow \mathcal{M} \cup \{\perp\}$ is called (γ, δ)-affine-evasive if for any $a, b \in F_p$ such that $a \neq 0$, and $(a, b) \neq (1, 0)$, and for any $m \in \mathcal{M}$,*

- *$Pr(h(aU + b) \neq \perp) \leq \gamma$*
- *$Pr(h(aU + b) \neq \perp | h(U) = m) \leq \delta$*
- *A uniformly random X such that $h(X) = m$ is efficiently samplable.*

Using these affine evasive functions, they arrive at the construction of split-state non-malleable codes by composing it with inner product. For $L, R \in F_p^\lambda$, let $\langle L, R\rangle$ represent the inner product $\langle L, R\rangle = \sum_{i=1}^{\lambda} L[i] \times R[i]$. Their scheme is as follows :

- The **decoding function ADLDec** : $F_p^\lambda \times F_p^\lambda \to \mathcal{M} \cup \{\perp\}$ is defined using affine evasive function h as follows : $\mathbf{ADLDec(L, R)} := \mathbf{h}(\langle \mathbf{L}, \mathbf{R} \rangle)$
- The **encoding function ADLEnc** : $\mathcal{M} \to F_p^\lambda \times F_p^\lambda$ is defined as $\mathbf{ADLEnc}(m) = (L, R)$ where L, R are chosen uniformly at random from $F_p^\lambda \times F_p^\lambda$ conditioned on the fact that $\mathbf{ADLDec}(L, R) = m$.

Theorem 4. *([ADL14]) Let $\mathcal{M} = \{1, 2, \dots, K\}$ and let $p \geq (\frac{4K}{\epsilon})^{\rho \log\log(4K/\epsilon)}$ be a prime. Let λ be $(\lceil \frac{2\log p}{c} \rceil)^6$. Let $\mathbf{ADLEnc} : \mathcal{M} \to F_p^\lambda \times F_p^\lambda, \mathbf{ADLDec} : F_p^\lambda \times F_p^\lambda \to \mathcal{M} \cup \{\perp\}$ be as defined above. Then the scheme $(\mathbf{ADLEnc}, \mathbf{ADLDec})$ is ϵ-non-malleable w.r.t $\mathcal{F}_2^{split}$.*

Multiplicative secret sharing scheme of [KGH83]. We recall the result of Karnin et al. [KGH83], in which they construct $(n, 0)$-secret sharing scheme realizing access structure $\mathcal{A}_n^n$ over arbitrary Abelian group. Let $(\mathbb{F}_p, +, \times)$ be a finite field. Let $\mathbb{F}_p^*$ be the set of non zero elements of the field $\mathbb{F}_p$, and this set along with the operation $\times$ forms an abelian group.

- $\mathbf{MultShare_n}$: Let $MultShare_n : \mathbb{F}_p^* \to \otimes_{i \in [n]} \mathbb{F}_p^*$ be a randomized sharing function. On input a secret $s \in \mathbb{F}_p^*$, sample the first $n-1$ shares, namely $s_1, s_2, \dots, s_{n-1}$, randomly from $\mathbb{F}_p^*$. Compute the last share using the secret s and the sampled shares as $s_n \leftarrow x / \prod_{i=1}^{n-1} s_i$ Output $s_1, \dots, s_n$.
- $\mathbf{MultRec_n}$: Let $MultRec_n : \otimes_{i \in [n]} \mathbb{F}_p^* \to \mathbb{F}_p^*$ be a deterministic function for reconstruction. On input n shares, namely $s_1, s_2, \dots, s_n$, compute $s \leftarrow \prod_{i=1}^n s_i$ and output the result s.

Theorem 5. *([KGH83]) $\mathbf{MultShare_n}, \mathbf{MultRec_n}$ is an $(n, 0)$-secret sharing scheme realizing access structure $\mathcal{A}_n^n$.*

Our n-out-of-n non-malleable secret sharing scheme :

Theorem 6. *Let the message space $\mathcal{M}$, prime p and vector length λ be as in the construction of $(\mathbf{ADLEnc}, \mathbf{ADLDec})$, the coding scheme of Aggarwal et al. [ADL14] that is ϵ-non-malleable against $\mathcal{F}_n^{split}$. Then for any number of parties $n \geq 2$, there exists an efficient construction of $\big(n, n, 2\epsilon + \frac{2\lambda}{p})\big)$-secret sharing scheme that is $\big(\epsilon + \frac{2\lambda}{p}\big)$-non-malleable w.r.t $\mathcal{F}_n^{general}$*

Corollary 7. *The coding scheme of Aggarwal et al. [ADL14] is also statistically-non-malleable w.r.t. $\mathcal{F}_2^{general}$. (which allows the tampering of left share to partially depend on the right share)*

Proof. (of theorem)

We begin with the description of our secret sharing scheme :

- The **reconstruction function** $\mathbf{JNMRec_n}$: $F_p^{\lambda \times n} \to \mathcal{M} \cup \{\perp\}$ is defined using affine evasive function h as follows :

$$\mathbf{JNMRec_n(sh_1, sh_2 \dots sh_n)} := \mathbf{h}(\langle \mathbf{sh_1, sh_2 \dots sh_n} \rangle)$$

where $\langle a_1, a_2 \dots a_n \rangle = \sum_{i=1}^{\lambda} \prod_{j=1}^{n} a_j[i]$ is the generalized inner product function.

- The **sharing function** $\mathbf{JNMShare_n} : \mathcal{M} \rightarrow F_p^{\lambda \times n}$ is defined as follows. On input m, output $(sh_1, sh_2 \dots sh_n)$ where $sh_1, sh_2 \dots sh_n$ are chosen uniformly at random from $F_p^{\lambda \times n}$ conditioned on the fact that $\mathbf{JNMRec_n}(sh_1, sh_2 \dots sh_n) = m$.

Correctness, Efficiency, and Statistical Privacy :Correctness and efficiency trivially follows from the construction. Statistical privacy follows from the non-malleability proved below (in a manner similar to lemma 1).

Statistical Non Malleability : We transform an attack on our scheme to an attack on the underlying split-state non-malleable code. Let the adversary choose a four tampering functions (f_1, g_1, f_2, g_2) and corresponding four subsets $(B_f^{in}, B_f^{out}, B_g^{out}, B_g^{in})$ from the allowed tampering class $\mathcal{F}_n^{general}$. Let $n_f \leftarrow |B_f^{in}|$ and $n_g \leftarrow |B_g^{in}|$ denote the cardinality of input set of indices of function f and g respectively. Similarly, let $n_f^{out} \leftarrow |B_f^{out}|$ and $n_g^{out} \leftarrow |B_g^{out}|$ denote the cardinality of output set of indices of function f and g respectively. Using these tampering functions, we give explicit pair of tampering function $(F, G) \in \mathcal{F}_2^{split}$. The description of the reduction follows :

- (**Initial Setup**) : Start with fixing the shares which occurs as input of both the tampering functions. Let $B_{fix} \leftarrow B_f^{in} \cap B_g^{in}$. Let $n_{fix} \leftarrow |B_{fix}|$ denote the cardinality of this common set. For each $i \in B_{fix}$, fix $share_i \leftarrow a_1^i, a_2^i, \dots, a_\lambda^i$ randomly such that each $a_i^j \in \mathbb{F}_p^*$.

- The **tampering function** $\mathbf{F(l)}$
  - On input a vector $l \in F_p^\lambda$, parse it as $l_1, l_2 \dots l_\lambda$ such that $l_i \in F_p$ for each $i \in [\lambda]$. If there exists an $i \in [\lambda]$ such that $l_i = 0$, then `abort`. Otherwise, for each $i \in [\lambda]$, calculate $prod_i \leftarrow (l_i / (\prod_{j \in B_f^{out} \cap B_{fix}} a_i^j))$ and use multiplicative sharing to share $prod_i$ into $n_f - n_{fix}$ shares. That is, let $\{a_i^j : j \in B_f^{in} \setminus B_{fix}\} \leftarrow \mathbf{MultShare_{n_f - n_{fix}}}(prod_i)$. Construct $share_i$ as $a_1^i, a_2^i, \dots, a_\lambda^i$ for each $i \in B_f^{in} \setminus B_{fix}$. (We are excluding shares in B_{fix} as they have already been fixed earlier)
  - Tamper the shares by executing the adversary specified function f_1.

    $$\{\widetilde{fshare}_j : j \in B_f^{out}\} \leftarrow \mathbf{f_1}(\{share_j : j \in B_f^{in}\})$$

    Similarly, compute the tampered shares using f_2.

    $$\{\widetilde{fshare}_j : j \in B_g^{out}\} \leftarrow \mathbf{f_2}(\{share_j : j \in B_f^{in}\})$$

  - Parse the tampered shares as $\widetilde{fshare}_i = (\widetilde{a}_1^i, \widetilde{a}_2^i \dots \widetilde{a}_\lambda^i)$ for each $i \in [n]$ (recall $[n] = B_f^{in} \cup B_f^{out}$ by assumption). Reconstruct the tampered value of l_i for each $i \in [\lambda]$ using the reconstruction function of multiplicative sharing. Let $\widetilde{l}_i \leftarrow \mathbf{MultRec_{n_f^{out}}}(\{\widetilde{a}_i^j : j \in [n]\})$.

  - Then construct the tampered vector $\widetilde{l}$ as $(\widetilde{l}_1, \widetilde{l}_2, \ldots, \widetilde{l}_\lambda)$ and output $\widetilde{l}$.

- The **tampering function** $\mathbf{G}(\mathbf{r})$
  - On input a vector $r \in F_p^\lambda$, parse it as $r_1, r_2 \ldots r_\lambda$ such that $r_i \in F_p$ for each $i \in [\lambda]$. If there exists an $i \in [\lambda]$ such that $r_i = 0$, then `abort`. Otherwise, for each $i \in [\lambda]$, calculate $prod_i \leftarrow (r_i/(\prod_{j\in B_f^{out}\cap B_{fix}} a_i^j))$ and use multiplicative sharing to share $prod_i$ into $n_g - n_{fix}$ shares. That is, let $\{a_i^j : j \in B_g^{in} \setminus B_{fix}\} \leftarrow \mathbf{MultShare_{n_g - n_{fix}}}(prod_i)$. Construct $share_i$ as $a_1^i, a_2^i, \ldots, a_\lambda^i$ for each $i \in B_g^{in} \setminus B_{fix}$. (We are excluding shares in B_{fix} as they have already been fixed earlier)
  - Tamper the shares by executing the adversary specified function g_1.

$$\{\widetilde{gshare}_j : j \in B_g^{out}\} \leftarrow \mathbf{g_1}(\{share_j : j \in B_g^{in}\})$$

    Similarly, compute the tampered shares using g_2.

$$\{\widetilde{gshare}_j : j \in B_f^{out}\} \leftarrow \mathbf{g_2}(\{share_j : j \in B_g^{in}\})$$

  - Parse the tampered shares as $\widetilde{gshare}_i = \widetilde{b}_1^i, \widetilde{b}_2^i \ldots \widetilde{b}_\lambda^i$ for each $i \in [n]$. Reconstruct the tampered value of r_i for each $i \in [\lambda]$ using the reconstruction function of multiplicative sharing. Let $\widetilde{r}_i \leftarrow \mathbf{MultRec_{n_g^{out}}}(\{\widetilde{b}_i^j : j \in [n]\})$.
  - Then construct the tampered vector $\widetilde{r}$ as $(\widetilde{r}_1, \widetilde{r}_2, \ldots, \widetilde{r}_\lambda)$ and output $\widetilde{r}$.

It is easy to see that the reduction does not terminate with probability at least $\left(1 - \frac{2\lambda}{p}\right)$.

Claim: For any l and r, if the reduction is successful in creating the n shares, then the secret hidden is these n shares is the same as the message encoded by l and r.
Proof: The reduction constructs an instance of the secret sharing scheme using l and r in a split-state manner. Basically , for all $i \in [\lambda]$, it creates parts of shares such that $\left(\prod_{j\in B_f^{out}} a_i^j\right) = l_i$ and $\left(\prod_{j\in B_g^{out}} a_i^j\right) = r_i$. In this way, it is ensured that the secret hidden by n shares is the same as the message encoded by challenge shares l and r of the underlying non-malleable code. This can be seen by the following calculation :

$$\begin{aligned}
\mathbf{JNMRec_n}\big(\{share_i : i \in [n]\}\big) &= h\big(\sum_{i=1}^{\lambda} \prod_{j=1}^{n} a_i^j\big) \\
&= h\Big(\sum_{i=1}^{\lambda} \big(\prod_{j\in B_f^{out}} a_i^j\big) \times \big(\prod_{j\in B_g^{out}} a_i^j\big)\Big) \\
&= h\big(\sum_{i=1}^{\lambda} l_i \times r_i\big) = h(\langle l, r\rangle)
\end{aligned}$$

$$= \mathbf{ADLDec}(l, r)$$

■

Claim: For any l and r, if the reduction is successful in creating the t shares, then the secret hidden is the t tampered shares is the same as the message encoded by the tampered l and the tampered r.

Proof: Let $\{\widetilde{share}_i : i \in [n]\}$ be the disjoint union of outputs of two tampering functions $\mathbf{f}(\{share_i : i \in B_f^{in}\})$ and $\mathbf{g}(\{share_i : i \in B_g^{in}\})$. Now the reduction transforms the tampered shares back to two tampered parts of non-malleable code. Let $(\mathbf{F}, \mathbf{G})$ be as defined in the reduction.

$$\begin{aligned}
\mathbf{ADLDec}\big(\mathbf{F}(l), \mathbf{G}(r)\big) &= h\big(\langle \mathbf{F}(l), \mathbf{G}(r) \rangle\big) \\
&= h\big(\langle \widetilde{l}, \widetilde{r} \rangle\big) \\
&= h\big(\sum_{i=1}^{\lambda} \widetilde{l}_i \times \widetilde{r}_i\big) \\
&= h\Big(\sum_{i=1}^{\lambda} \big(\prod_{j \in [n]} \widetilde{a}_i^j\big) \times \big(\prod_{j \in [n]} \widetilde{b}_i^j\big)\Big) \\
&= h\big(\sum_{i=1}^{\lambda} \prod_{j=1}^{n} (\widetilde{a}_i^j \times \widetilde{b}_i^j)\big) \\
&= h\big(\sum_{i=1}^{\lambda} \prod_{j=1}^{n} (\widetilde{share}_j[i])\big) \\
&= \mathbf{JNMRec_n}\big(\{\widetilde{share}_j : j \in [n]\}\big)
\end{aligned}$$

■

By design the tampering functions $\mathbf{F}$ and $\mathbf{G}$ belongs to $\mathcal{F}_2^{split}$. By the ϵ-non malleability of the scheme $(\mathbf{ADLEnc}, \mathbf{ADLDec})$, we know that there exists a distribution $D^{F,G}$ such that

$$\mathbf{Sim_m^{F,G}} \approx_{\epsilon} \mathbf{Tamper_m^{F,G}}$$

Using the observation about the equivalence of tampering, and assuming that the adversary succeeds in case the reduction terminates by executing `abort`, we get that

$$\mathbf{STamper_m^{f,T}} \approx_{\epsilon + \frac{2\lambda}{p}} \mathbf{SSim_m^{f,T}}$$

This proves the non malleability of our scheme. □

Acknowledgments

We thank the anonymous reviewers, as their detailed and insightful reviews significantly helped in improving the presentation of this article.

The first author is supported by a grant from Northrop Grumman.

A part of this work was done while the second author was at Microsoft Research, India. Work done at UCLA is supported in part from NSF grant 1619348, NSF frontier award 1413955, US-Israel BSF grants 2012366, 2012378, and by the Defense Advanced Research Projects Agency (DAPRA) SAFEWARE program through the ARL under Contract W911NF-15-C-0205 and through a subcontract with Galois, inc. The views expressed are those of the authors and do not reflect the official policy or position of the Department of Defense, the National Science Foundation, or the U.S. Government.

References

ADKO15. Divesh Aggarwal, Stefan Dziembowski, Tomasz Kazana, and Maciej Obremski. Leakage-resilient non-malleable codes. In *Twelfth IACR Theory of Cryptography Conference (TCC 2015)*, 2015.

ADL14. Divesh Aggarwal, Yevgeniy Dodis, and Shachar Lovett. Non-malleable codes from additive combinatorics. In *Proceedings of the 46th Annual ACM Symposium on Theory of Computing*, pages 774–783. ACM, 2014.

Bei. Amos Beimel. *Secure schemes for secret sharing and key distribution, PhD Thesis.*

Bei11. Amos Beimel. Secret-sharing schemes: a survey. In *International Conference on Coding and Cryptology*, pages 11–46. Springer Berlin Heidelberg, 2011.

Bla79. G. R. Blakley. Safeguarding cryptographic keys. In *AFIPS National Computer Conference (NCC '79)*, pages 313–317, Los Alamitos, CA, USA, 1979. IEEE Computer Society.

BOGW88. Michael Ben-Or, Shafi Goldwasser, and Avi Wigderson. Completeness theorems for non-cryptographic fault-tolerant distributed computation. In *Proceedings of the twentieth annual ACM symposium on Theory of computing*, pages 1–10. ACM, 1988.

CDF$^+$08. Ronald Cramer, Yevgeniy Dodis, Serge Fehr, Carles Padró, and Daniel Wichs. Detection of algebraic manipulation with applications to robust secret sharing and fuzzy extractors. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 471–488. Springer, 2008.

CDTV16. Sandro Coretti, Yevgeniy Dodis, Björn Tackmann, and Daniele Venturi. Non-malleable encryption: simpler, shorter, stronger. In *Theory of Cryptography Conference*, pages 306–335. Springer, 2016.

CGL16. Eshan Chattopadhyay, Vipul Goyal, and Xin Li. Non-malleable extractors and codes, with their many tampered extensions. In *STOC*, 2016.

DKO13. Stefan Dziembowski, Tomasz Kazana, and Maciej Obremski. Non-malleable codes from two-source extractors. In *Advances in Cryptology–CRYPTO 2013*, pages 239–257. Springer, 2013.

DPW10. Stefan Dziembowski, Krzysztof Pietrzak, and Daniel Wichs. Non-malleable codes. In *Innovations in Computer Science - ICS 2010, Tsinghua University, Beijing, China, January 5-7, 2010. Proceedings*, pages 434–452, 2010.

GGSW13. Sanjam Garg, Craig Gentry, Amit Sahai, and Brent Waters. Witness encryption and its applications. In *Proceedings of the forty-fifth annual ACM symposium on Theory of computing*, pages 467–476. ACM, 2013.

GJK15. Vipul Goyal, Aayush Jain, and Dakshita Khurana. Non-malleable multi-prover interactive proofs and witness signatures. Cryptology ePrint Archive, Report 2015/1095, 2015. `http://eprint.iacr.org/2015/1095`.

GK18. Vipul Goyal and Ashutosh Kumar. Non-malleable secret sharing. In *To Appear in Proceedings of the fiftieth ACM STOC*. ACM, 2018.

Gol07. Oded Goldreich. *Foundations of cryptography: volume 1, basic tools*. Cambridge university press, 2007.

GPR16. Vipul Goyal, Omkant Pandey, and Silas Richelson. Textbook non-malleable commitments. In *Proceedings of the 48th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2016, Cambridge, MA, USA, June 18-21, 2016*, pages 1128–1141, 2016.

ISN89. Mitsuru Ito, Akira Saito, and Takao Nishizeki. Secret sharing scheme realizing general access structure. *Electronics and Communications in Japan (Part III: Fundamental Electronic Science)*, 72(9):56–64, 1989.

KGH83. Ehud Karnin, Jonathan Greene, and Martin Hellman. On secret sharing systems. *IEEE Transactions on Information Theory*, 29(1):35–41, 1983.

KNY14. Ilan Komargodski, Moni Naor, and Eylon Yogev. Secret-sharing for np. In *International Conference on the Theory and Application of Cryptology and Information Security*, pages 254–273. Springer, 2014.

KW93. Mauricio Karchmer and Avi Wigderson. On span programs. In *Structure in Complexity Theory Conference, 1993., Proceedings of the Eighth Annual*, pages 102–111. IEEE, 1993.

Li17. Xin Li. Improved non-malleable extractors, non-malleable codes and independent source extractors. In *STOC*. ACM, 2017.

LL12. Feng-Hao Liu and Anna Lysyanskaya. Tamper and leakage resilience in the split-state model. In *CRYPTO*, pages 517–532, New York, NY, USA, 2012. Springer-Verlag New York, Inc.

MS81. Robert J. McEliece and Dilip V. Sarwate. On sharing secrets and reed-solomon codes. *Communications of the ACM*, 24(9):583–584, 1981.

RBO89. T. Rabin and M. Ben-Or. Verifiable secret sharing and multiparty protocols with honest majority. In *STOC*, STOC '89, pages 73–85, New York, NY, USA, 1989. ACM.

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