Non-Malleable Codes for Decision Trees

Marshall Ball[1], Siyao Guo[2], and Daniel Wichs[3]

[1] Columbia University, New York, USA
marshall@cs.columbia.edu
[2] New York University Shanghai, Shanghai, China
siyao.guo@nyu.edu
[3] Northeastern University, Boston, USA
wichs@ccs.neu.edu

Abstract. We construct efficient, unconditional non-malleable codes that are secure against tampering functions computed by decision trees of depth $d = n^{1/4-o(1)}$. In particular, each bit of the tampered codeword is set arbitrarily after adaptively reading up to d arbitrary locations within the original codeword. Prior to this work, no efficient unconditional non-malleable codes were known for decision trees beyond depth $O(\log^2 n)$.

Our result also yields efficient, unconditional non-malleable codes that are $\exp(-n^{\Omega(1)})$-secure against constant-depth circuits of $\exp(n^{\Omega(1)})$-size. Prior work of Chattopadhyay and Li (STOC 2017) and Ball et al. (FOCS 2018) only provide protection against $\exp(O(\log^2 n))$-size circuits with $\exp(-O(\log^2 n))$-security.

We achieve our result through simple non-malleable reductions of decision tree tampering to split-state tampering. As an intermediary, we give a simple and generic reduction of leakage-resilient split-state tampering to split-state tampering with improved parameters. Prior work of Aggarwal et al. (TCC 2015) only provides a reduction to split-state non-malleable codes with decoders that exhibit particular properties.

1 Introduction

Motivated by applications in tamper-resilience, non-malleable codes were first introduced by Dziembowski, Pietrzak, and Wichs [DPW10] as an extension of error-correcting codes that give meaningful guarantees even when every bit of a codeword may be altered. To define the non-malleability of an encoding scheme (Enc, Dec) for a class of tampering functions $\mathcal{F}$, consider the following experiment for any $f \in \mathcal{F}$: (1) encode a message x via Enc, (2) tamper the resulting codeword with f, and (3) decode the tampered codeword with Dec. Roughly, (Enc, Dec) is non-malleable if $\tilde{x} = \mathrm{Dec}(f(\mathrm{Enc}(x)))$ is either completely unrelated to the original message x or identical to x, for any x. In particular, the outcome of the experiment should be simulatable without any knowledge of x, up to allowing the simulator to output a special symbol "same" to indicate that the message is unchanged.

The initial work of Dziembowski et al. [DPW10] observed that achieving non-malleability against arbitrary tampering is strictly impossible. The goal, as in many coding tasks, is to construct a code against as large a class of tampering functions (channels) as possible with best information rate and strongest security guarantees. And, since 2010 a flurry of work has done just that, studying this object in a variety of models. Of particular interest, both at large and in this work, is explicit constructions of statistically secure non-malleable codes for specific families of tampering functions and the efficiency of these constructions (in both information rate and computational complexity).

Much of the work on explicit constructions has focused on split-state tampering, where the codeword is broken into blocks according to some (fixed a priori) partition and each block may be tampered with independently of all other blocks [DKO13,CG14,CZ14,ADL14,Agg15,CGL16,AB16,KOS17,Li17,Li18]. However, a recent strand of work has focused on tampering functions that aren't restricted to fixed partitions. Since initial work on tampering via permutations and bit-flipping [AGM$^+$15] many of these works have looked at tampering functions that are restricted under some measure of computational complexity.

In 2016, Ball et al. [BDKM16] constructed non-malleable codes for ℓ-local tampering functions (functions where each output only depends on ℓ inputs) for $\ell = o(n/\log n)$ with rate proportional to ℓ restriction and negligible error. This class contains NC^0, functions computed by circuits composed from a constant depth of constant fan-in gates. In 2017, Chattopadhyay and Li [CL17] constructed non-malleable codes for AC^0 (polynomial size circuits of constant depth, unbounded fan-in AND and OR gates, as well as NOT gates). Their construction achieved a negligible error, but had exponentially larger codewords of length $n = 2^{\sqrt{k}}$ for messages of length k.[4] In 2018, Ball et al. [BDG$^+$18] constructed non-malleable codes for AC^0 with negligible error and codewords of almost linear length ($n = k^{1+o(1)}$).[5]

However, none of the known constructions for small-depth circuits can support tampering via *large size* small-depth circuits nor can they provide security guarantees beyond $\exp(-\omega(\log^2 n))$ even for AC^0 itself. This leads to the following problem which we address in this work:

Construct non-malleable codes for AC^0 with error bounds $\exp(-\omega(\log^2 n))$.

In fact, the techniques of [BDG$^+$18] immediately suggest a path to improving non-malleable codes for small-depth circuits, by resolving another open question which is interesting, independent of the above motivation:

[4] [CL17] also gave a construction for local functions with polynomial length codewords and sub-exponential error.

[5] Actually, the construction of [BDG$^+$18] can handle a slightly wider range of parameters including polynomial size circuits of depth $o(\log n/\log\log n)$ and constant depth circuits of size $n^{O(\log n)}$. Note that depth d decision trees are also a strict subclass of 2^d-local functions. Accordingly, Ball et al.'s codes for $n^{1-\varepsilon}$-local tampering handle decision tree tampering of depth up to $(1-\varepsilon)\log n$.

Construct non-malleable codes for decision trees of depth $d = \omega(\log^2 n)$, or ideally $d = n^{\Omega(1)}$.

Decision trees of depth d capture tampering where each output bit is set arbitrarily after adaptively reading d locations of the input, where the choice of which input location to read next at any point in time can depend on the values of all the previous locations read. While tampering via decision trees has been studied in the Common Reference String (CRS) model under cryptographic assumptions [BDKM18], prior to the present work no efficient or explicit information-theoretic codes were known for decision trees of $\omega(\log^2 n)$ depth.[6]

1.1 Our Results

Theorem 1 (Informal). *There exists an explicit, efficient, information theoretic non-malleable code for decision trees of depth* $n^{1/4-o(1)}$.

Given the above, the following theorem is a straightforward corollary to a lemma of [BDG+18] that reduces small-depth circuit tampering to tampering functions that may adaptively leak limited information from the codeword before selecting local functions to modify the codeword with. It is easy to observe that this class is subsumed by decision tree tampering of sufficient depth (See Lemma 8 in Section 5 for more details).

Theorem 2 (Informal). *For* $d \leq c_1 \log n / \log\log n$*, there exists an explicit, efficient, information theoretic non-malleable code for* d*-depth circuits (of unbounded fan-in) of size* $\exp(n^{c_2/d})$ *with error* $\exp(-n^{\Omega(1/d)})$ *and encoding length* $n = k^{1+c}$*, where* $c, c_1, c_2 \in (0,1)$ *are constants.*

In particular, for constant-depth circuits, our result yields efficient, unconditional non-malleable codes that are $\exp(-n^{\Omega(1)})$-secure against $\exp(n^{\Omega(1)})$-size circuits. Prior work [BDG+18,CL17] only provides protection against circuits of size $\exp(O(\log^2 n))$ with $\exp(-O(\log^2 n))$ security.[7]

It is easy to observe that a non-malleable code $(\mathrm{Enc}, \mathrm{Dec})$ with error ε for most classes $\mathcal{C}$ implies a strong average-case lower bound on the power of that class – in particular, Dec is ε-hard to compute on average via $\mathcal{C}$ with respect to the distribution $\mathrm{Enc}(b)$ where b is uniformly chosen.[8] For this reason, explicit constructions of non-malleable codes (with good error) for even a slightly larger class $\mathsf{AC}^0[2]$ (AC^0 with the addition of mod 2 gates) would necessitate a better structural understanding of this class than is currently known.

[6] Note that any decision tree of depth d can also be represented by a 2^d-local function or as a DNF with 2^d clauses of width d.

[7] Note that if security $2^{-\lambda}$ is required, these codes will no longer be efficient. In particular, the codeword lengths in both cases will be super-polynomial in λ.

[8] For tampering functions such that each output bit is in the class $\mathcal{C}$, the implications follows so long as $\mathcal{C}$ contains the constant functions and is closed under negation.

Along the way, we also construct new leakage-resistant split-state codes. Our construction allows up to a 1/4-fraction of the codeword to be leaked which, to our knowledge, is the best known.[9]

Theorem 3 (Informal). *There is an explicit split-state non-malleable code supporting leakage of up to a* $1/4$*-fraction of the bits with rate* $\Omega(\log\log n/\log n)$ *and error* $\exp(-\Omega(n\log\log n/\log n))$.

In particular, our construction allows one to reduce leakage-resilient split-state tampering to *any* split-state tampering with just a constant factor increase in codeword size. Previously, such reductions were only known with worse parameters and if the underlying split-state code had decoder with certain properties [ADKO15b]. On the other hand, [CL18] show that certain split-state seedless non-malleable extractors *are*, in fact, leakage-resilient, yielding codes with rate comparable to that of the present work (but the leakage bound is left unspecified). Unlike either of these previous construction, our reduction would yield improved leakage-resilient split-state non-malleable codes from *any* future improvement in the rate of explicit split-state non-malleable codes. Finally, our analysis is much simpler than that of [ADKO15b].

1.2 Our Techniques

We construct our codes for decision trees by constructing a *non-malleable reduction* from decision tree tampering to split-state tampering. A non-malleable reduction, first defined in [ADKO15a], from $\mathcal{F}$ to $\mathcal{G}$ is simply an encoding scheme, $(\mathrm{Enc}, \mathrm{Dec})$ such that for any $f \in \mathcal{F}$ and any message x, the value $\mathrm{Dec}(f(\mathrm{Enc}(x)))$ is statistically close to $g(x)$ for g chose from some distribution over functions in $\mathcal{G}$. In particular a non-malleable code for $\mathcal{F}$ is a non-malleable reduction from $\mathcal{F}$ to the family $\mathcal{G}$ consisting of the identity function and constant functions.

Both the reduction and its analysis are surprisingly simple. We also note that our reduction can be seen as distillation of the core ideas behind the construction for local tampering from [BDKM16,BDG$^+$18]. One can view the reductions in these papers as, in fact, performing two reductions: local tampering to leaky split-state tampering, and leaky split-state tampering to split-state tampering. Viewed this way, these works implicitly construct a very weak form a leakage-resilient split-state code. It is weak in that it can only handle the leakage of a few bits chosen in advance, in particular it is not robust enough to handle even adaptive choices of the bits. While leakage-resilient split-state codes are already known [ADKO15b,CL18], this alone is not enough. In particular, the implicit reduction from local tampering to leaky split-state tampering in [BDKM16,BDG$^+$18] does not seem to hold for decision trees. However, this modular perspective simplifies the analysis tremendously, even for this original case of local tampering.

[9] [CL18] does not give an explicit bound on leakage and [ADKO15b] allows 1/12-fraction leakage (or 1/6 in a more restricted model where the leakage amount from each side has to be the same).

We will outline our new reduction for decision tree tampering. The key idea here, as in [BDKM16,BDG$^+$18], is to exploit size differences. Our encoder and decoder will work independently on the left and right pieces of the message, so we will in turn think of having left and right encoders, decoders, codewords, and tampering functions (corresponding to the respective outputs).

First, let us suppose that the right piece of the message (corresponding to the right split-state codeword) is much longer than that of the left. Then, suppose both the right and left encoders and decoders are simply the identity function. Then, all the left tampering functions together will make a number of queries to the right codeword that is below our leakage threshold.

However, because the right is much longer than the left, the above analysis won't help in simulating tampering on the right with low leakage from the left. Instead, we modify the left encoder/decoder to make it much longer than the right, but while retaining the property that the left can be decoded from just a few decision trees. To do so, we sample a random small set, whose size is that of the message, in a much larger array. We plant the message in these locations and zero everything else out. Then, we bit-wise secret share a description of the small set (i.e., its seed) such that the secrecy threshold is relatively large. To decode, we can simply extract the seed and output what is in the corresponding locations of the array.

Now, note that decoding the left still only requires at most relatively few queries to the right: decision tree depth times both encoded seed length plus message length. But we can't make the encoded seed too long or we will be dead again. Instead, we critically use the fact that tampering is by a *forest* of decision trees. In particular, for any small set of tampering functions on the right, the seed remains uniformly chosen regardless of what queries the set makes, so we expect only a small fraction of any queries made to the array to actually hit the message locations. Strong concentrations bounds guarantee that this is more or less what actually happens. Finally we simply union bound over all such subsets to guarantee that collectively the right tampering function makes few queries to the left with overwhelming probability.

Finally, we apply the same style of encoding used on the left to the right side to fix the syntactic mismatch and reduce to the case where the right and left messages are the same size.

As mentioned above, non-malleable codes for leakage-resilient split-state are known from prior work [ADKO15b,CL18]. However, we give a new reduction from leakage-resilient split-state tampering to split-state tampering that, when combined with the state-of-the-art split-state non-malleable codes [Li18], improves on the (explicit) parameters of what was known before.

Our leakage-resilient reduction is quite intuitive. We show that given a statistically-secure leakage-resilient encryption scheme (where an adversary can receive bounded leakage from both ciphertext *and* the key used to encrypt it) it suffices to simply encrypt the left and right split-state codewords independently (with their own keys) and place the keys in the opposite state. By the strong security property of the encryption scheme, the ciphertext hides each underlying split-state codeword piece, whatever (bounded amount) is leaked from the key in the opposite state.

To complete the reduction, we construct such a statistically-secure leakage-resilient encryption scheme from extractors. Our notion of leakage-resilience essentially combines the notions of "forward-secure storage" [Dzi06] and "leakage-resilient storage" [DDV10] to get the best of both worlds, and our construction essentially combines the ideas behind the constructions of the above two objects.

Related Work in Other Models. The aforementioned work of [BDKM18] constructs a generic framework for converting correlation bounds into non-malleable codes. They instantiate their framework to construct non-malleable codes both for decision trees and large small-depth circuits. However, this work (a) requires a Common Reference String, and (b) is secure in a game-based model against efficient adversaries, assuming unproven cryptographic assumptions. In fact, one of the assumptions, public key encryption with decryption in AC^0, necessarily limits their error term to be at most $\exp(-\log^{O(1)}(n))$.

A very recent follow-up work, [BDSK+18], improves upon [BDKM18] showing how to remove the Common Reference String, but still does not achieve unconditional or statistical guarantees.

2 Preliminaries

For a positive integer n, we use $[n]$ to denote $\{1, \ldots, n\}$. For $x = (x_1, \ldots, x_n) \in \{0,1\}^n$ and $i \le j \in [n]$, we define $x_{i:j} := (x_i, \ldots, x_j)$. For a set $S \subseteq [n]$ or a string $S \in \{0,1\}^n$, x_S denotes the projection of x to S. For $x, y \in \{0,1\}^n$, if they disagree on at least $\varepsilon \cdot n$ indices, we say they are ε-far, otherwise, they are ε-close to each other.

For a set Σ, we use Σ^Σ to denote the set of all functions from Σ to Σ. Given a distribution $\mathcal{D}$, $z \leftarrow \mathcal{D}$ denotes sample z according to $\mathcal{D}$. For two distributions $\mathcal{D}_1$,$\mathcal{D}_2$ over Σ, their statistical distance is defined as $\Delta(\mathcal{D}_1, \mathcal{D}_2) := \frac{1}{2} \sum_{z \in \Sigma} |\mathcal{D}_1(z) - \mathcal{D}_2(z)|$.

2.1 Non-malleable Reductions and Codes

Non-malleable codes were first defined in [DPW10]. Here we use a simpler, but equivalent, definition based on the following notion of non-malleable reduction by Aggarwal et al. [ADKO15a].

Definition 1 (Non-Malleable Reduction [ADKO15a]). *Let $\mathcal{F} \subset A^A$ and $\mathcal{G} \subset B^B$ be some classes of functions. We say $\mathcal{F}$* reduces to *$\mathcal{G}$, $(\mathcal{F} \Rightarrow \mathcal{G}, \varepsilon)$, if there exists an efficient (randomized) encoding function* Enc : $B \to A$, *and an efficient decoding function* Dec : $A \to B$, *such that*

(a) $\forall x \in B, \Pr[\mathrm{Dec}(\mathrm{Enc}(x)) = x] = 1$ *(over the randomness of* E*).*
(b) $\forall f \in \mathcal{F}, \exists G$ *s.t.* $\forall x \in B$, $\Delta(\mathrm{Dec}(f(\mathrm{Enc}(x))); G(x)) \le \varepsilon$, *where G is a distribution over $\mathcal{G}$ and $G(x)$ denotes the distribution $g(x)$, where $g \leftarrow G$.*

If the above holds, then (Enc, Dec) *is an* $(\mathcal{F}, \mathcal{G}, \varepsilon)$-non-malleable reduction*.*

Definition 2 (Non-Malleable Code [ADKO15a]). *Let* NM_k *denote the set of* trivial manipulation functions *on* k*-bit strings, consisting of the identity function* $id(x) = x$ *and all constant functions* $f_c(x) = c$*, where* $c \in \{0,1\}^k$*.* (Enc, Dec) *defines an* $(\mathcal{F}_{n(k)}, k, \varepsilon)$-non-malleable code*, if it defines an* $(\mathcal{F}_{n(k)}, \mathrm{NM}_k, \varepsilon)$*-non-malleable reduction. Moreover, the rate of such a code is taken to be* $k/n(k)$*.*

The following useful theorem allows us to compose non-malleable reductions.

Theorem 4 (Composition [ADKO15a]). *If* $(\mathcal{F} \Rightarrow \mathcal{G}, \varepsilon_1)$ *and* $(\mathcal{G} \Rightarrow \mathcal{H}, \varepsilon_2)$*, then* $(\mathcal{F} \Rightarrow \mathcal{H}, \varepsilon_1 + \varepsilon_2)$*.*

2.2 Tampering Function Classes

Definition 3 (Split-State Model [DPW10]). *The* split-state model*,* SS_k*, denotes the set of all functions:*

$$\{f = (f_1, f_2) : \ f(x) = (f_1(x_{1:k}) \in \{0,1\}^k, f_2(x_{k+1:2k}) \in \{0,1\}^k), x \in \{0,1\}^{2k}\}.$$

One natural extension of split-state model is leaky/bounded-communication split-state functions considered by Aggarwal et al. [ADKO15b] and Chattopadhyay et al. [CL18].

Definition 4 (Leaky/Bounded-Communication Split-State Model). *Let* $\alpha \in [0,1]$ *be a parameter. We say* $f \in \{0,1\}^{2k} \to \{0,1\}^{2k}$ *is in* α-leaky split-state model*,* $\alpha - \mathrm{SS}_k$ *if there exists a communication protocol between Alice and Bob such that for* $x = (x_{1:k}, x_{k+1:2k}) \in \{0,1\}^{2k}$*,* $f(x)$ *can be computed by a communication protocol with parameter* α *between Alice and Bob where Alice has access to* $x_{1:k}$*, Bob has access to* $x_{k+1:2k}$*. Alice and Bob send information back and forth depending on their own inputs and the current transcript of the communication so far. Overall the total communication is at most* αk *bits and finally Alice outputs* $f(x)_{1:k}$ *and Bob outputs* $f(x)_{k+1:2k}$*.*

Definition 5 (Decision Trees). *A decision tree with* n *input bits is a binary tree whose internal nodes have labels from* $x_1, \ldots, x_n$ *and whose leaves have labels from* $\{0,1\}$*. If a node has label* x_i *then the test performed at that node is to examine the* i*-th bit of the input. If the result is* 0*, one descends into the left subtree , whereas if the result is* 1*, one descends into the right subtree. The label of the leaf so reached is the output value on that particular input. The* depth *of a decision tree is the number of edges in a longest path from the root to a leaf. Let* $\mathrm{DT}(t)$ *denote decision trees with depth at most* t*.*

Definition 6 (Small Depth Circuits). *A Boolean circuit with* n *input bits is a directed acyclic graph in which every node (also called gate) is either an input node of in-degree* 0 *labeled by one of the* n *input bits, an AND gate, an OR gate, or a NOT gate. One of these gates is designated as the output gate. The* size *of a circuit is its number of gates and the* depth *of a circuit is the length of its longest path from an input gate to the output gate. Let* $\mathrm{AC}_d(S)$ *denote depth* d *circuits of size at most* S *with unbounded fan-in.*

For a set of boolean circuits $\mathcal{C}$ (respectively decision trees), we say that a boolean function f is in $\mathcal{C}$, if there exists a $C \in \mathcal{C}$ which agrees with f on every input. We say that **a multiple output function** $f = (f_1, \ldots, f_m)$ **is in** $\mathcal{C}$ if $f_i \in \mathcal{C}$ for any $i \in [m]$.

2.3 Pseudorandom Objects

Extractors

Definition 7 (Weak Random Sources). *The min-entropy of a distribution* X *over* $\{0,1\}^n$ *is* $\mathrm{H}_\infty(X) := -\log(\max_{x\in\{0,1\}^n} \Pr[X = x])$. *A distribution* X *over* $\{0,1\}^n$ *is called an* (n,k) *source if* $\mathrm{H}_\infty(X) \geq k$.

Definition 8 (Strong Extractors [NZ96]). *A function* $\mathsf{Ext} : \{0,1\}^{n+d} \to \{0,1\}^m$ *is a* (k,ε) *extractor if for every* (n,k) *source* X, $\mathrm{Y}, \mathsf{Ext}(\mathrm{X},\mathrm{Y})$ *is* ε*-close to* Y, U_m *where* Y *is uniformly distributed over* $\{0,1\}^d$ *and* U_m *is uniformly distributed over* $\{0,1\}^m$. *An extractor is explicit if it is computable in polynomial time.*

Theorem 5 (Explicit Strong Extractors [GUV09]). *For every constant* $\alpha > 0$, *and all positive integers* n, k *and all* $\varepsilon > 0$, *there is an explicit construction of a* (k,ε) *extractor* $\mathsf{Ext} : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ *with* $d = O(\log n + \log(1/\varepsilon))$ *and* $m \geq (1-\alpha)k$.

Definition 9 (Two Source Extractors [CG88]). *A function* $2\mathsf{Ext} : \{0,1\}^n \times \{0,1\}^n \to \{0,1\}^m$ *is a* (k,ε) *two source extractor if for independent source* X *and sources* Y *such that* $\mathrm{H}_\infty(X) + \mathrm{H}_\infty(Y) \geq k$, $(\mathrm{Y}, 2\mathsf{Ext}(\mathrm{X},\mathrm{Y}))$ *is* ε*-close to* $(\mathrm{Y}, \mathrm{U}_m)$ *and* $(\mathrm{X}, 2\mathsf{Ext}(\mathrm{X},\mathrm{Y}))$ *is* ε*-close to* $(\mathrm{X}, \mathrm{U}_m)$ *where* U_m *is uniformly distributed over* $\{0,1\}^m$. *An extractor is explicit if it is computable in polynomial time.*

Theorem 6 (Explicit Two Source Extractors [CG88]). *For all positive integers* m, n *such that* n *is a multiple of* m *and for all* $\varepsilon \geq 0$, *there exists an efficient* $(n + m + 2\log 1/\varepsilon, \varepsilon)$ *2-source extractor with* n*-bit sources and* m*-bit output.*[10]

Binary Ramp Secret Sharing Encoding Schemes

Definition 10 (Binary Ramp Secret Sharing Encoding Schemes). *We say* $(\mathrm{Enc}, \mathrm{Dec})$ *is a* binary ramp secret sharing encoding scheme *with parameters* (k, n, c_{sec}), *where* $k, n \in \mathbb{N}$, $0 \leq c_{sec} < 1$, *if it satisfies the following properties:*

1. ***Reconstruction.*** $\mathrm{Enc}\colon \{0,1\}^k \to \{0,1\}^n$ *is an efficient probabilistic procedure, which maps a message* $x \in \{0,1\}^k$ *to a distribution over* $\{0,1\}^n$, *and* $\mathrm{Dec}\colon \{0,1\}^n \to \{0,1\}^k$ *is an efficient procedure. For any* $x \in \{0,1\}^k$, *it holds that* $\Pr[\mathrm{Dec}(\mathrm{Enc}(x)) = x] = 1$.

[10] [CG88] implies this theorem and the parameters have been taken from [ADKO15b].

2. ***Secrecy of partial views.*** *For any* $x \in \{0,1\}^k$ *and any non-empty set* $S \subset [n]$ *of size* $\leq \lfloor c_{sec} \cdot n \rfloor$, $\mathrm{Enc}(x)_S$ *is identically distributed to the uniform distribution over* $\{0,1\}^{|S|}$.

As observed by Ball et al. [BDG+18] (cf. Lemma 2), such a coding scheme can be constructed efficiently from any linear error correcting code. We reproduce their construction here for convenience and refer the reader to [BDG+18] for further discussion.

Lemma 1 ([BDG+18]). *Suppose there exists a binary linear error correcting code with parameters* (k, n, d), *then there is a binary ramp secret sharing scheme with parameters* $(k, n, (d-1)/n)$.

Proof. For a linear error correcting code with (k, n, d), let A denote its encoding matrix, H denote its parity check matrix. Let B be a matrix so that $BA = I$ where I is the $k \times k$ identity matrix (such B exists because A has rank k and can be found efficiently). By property of parity check matrix, $HA = \mathbf{0}$ and $Hs \neq 0$ for any $0 < ||s||_0 < d$ where $\mathbf{0}$ is the $(n-k) \times k$ all 0 matrix.

We define $(\mathrm{Enc}, \mathrm{Dec})$ as follows: for $x \in \{0,1\}^k$ and randomness $r \in \{0,1\}^{n-k}$, $\mathrm{Enc}(x; r) := B^T x + H^T r$, for $c \in \{0,1\}^n$; $\mathrm{Dec}(c) := A^T c$.

$(\mathrm{Enc}, \mathrm{Dec})$ is an encoding scheme because $\mathrm{Dec} \circ \mathrm{Enc} = A^T B^T = I^T = I$. For secrecy property, note that for any non-empty $S \subseteq [n]$ of size at most $d-1$, $(Hr)_S$ is distributed uniformly over $\{0,1\}^{|S|}$, because for any $a \in \{0,1\}^{|S|}$,

$$\begin{aligned}\Pr_r[(H^T r)_S = a] &= \mathrm{E}[\mathit{\Pi}_{i\in S} \frac{1 + (-1)^{(H^T r)_i + a_i}}{2}] \\ &= 2^{-|S|} \sum_{S' \subseteq S} \mathrm{E}[\mathit{\Pi}_{i \in S'} (-1)^{(H^T r)_i + a_i}] = 2^{-|S|},\end{aligned}$$

where the last equality is because the only surviving term is $S' = \emptyset$ and for other S', $\sum_{i \in S'} H_i^T \neq 0$ so $\mathrm{E}[\mathit{\Pi}_{i \in S'} (-1)^{(H^T r)_i}] = 0$. It implies $\mathrm{Enc}(x)_S$ is also distributed uniformly over $\{0,1\}^S$. Hence $(\mathrm{Enc}, \mathrm{Dec})$ is a binary ramp secret sharing encoding scheme with parameters $(k, n, (d-1)/n)$.

The following lemma is an immediate consequence of the former and any construction of a good code, such as a Justesen code.

Lemma 2. *For any* $k \in \mathbb{N}$, *there exist constants* $0 < c_{rate}, c_{sec} < 1$ *such that there is a binary RSS scheme with parameters* $(k, c_{rate}k, c_{sec})$.

To achieve longer encoding lengths n, with the same c_{sec} parameter, one can simply pad the message to an appropriate length.

2.4 Concentration Inequalities

Theorem 7 (Generalized Chernoff Bound [PS97]). *Let* $X_1, \ldots, X_n$ *be boolean random variables such that, for some* $0 \leq \delta \leq 1$, *we have that, for every subset* $S \subseteq [n]$, $\mathrm{E}[\mathit{\Pi}_{i \in S} X_i] \leq \delta^{|S|}$. *Then* $\Pr[\sum_{i=1}^n X_i \geq 2\delta n] \leq \exp(-n\delta/3)$.

3 Decision Trees to Leaky Split-State Model

In this section, we give a non-malleable reduction from decision tree tampering to leaky split-state tampering.

Lemma 3. *For any constant $\alpha \in (0, 1)$ and $t = O(n^{1/4}/\log^{3/2} n)$, there is a $(\mathrm{DT}(t) \Rightarrow \alpha - \mathrm{SS}_k, \varepsilon)$-non-malleable reduction with rate $\Omega(1/t^2 \log^3 n)$ where $\varepsilon \le \exp(-\Omega(n/t^4 \log^5 n))$.*

A single decision tree from $\{0, 1\}^n$ to $\{0, 1\}$ of depth t can be viewed as an adversary (or computation) that first adaptively queries at most t input coordinates, and then outputs a bit. Given a tampering function f from $\{0, 1\}^n$ to $\{0, 1\}^n$ such that each output bit is computed by a decision tree of depth t (i.e., each output is the result of adaptively querying at most t coordinates from the input, here a codeword), we will non-malleably reduce f to a special subclass of leaky split-state functions.

Recall that an α-leaky split-state function $g = (g_L, g_R)$ from $\{0, 1\}^{2k}$ to $\{0, 1\}^{2k}$ can be computed by a communication protocol with parameter α between Alice and Bob where Alice has access to $x_L := x_{1:k}$ and Bob has access to $x_R := x_{k+1:2k}$. Alice and Bob send information back and forth depending on their own inputs and the current transcript of the communication so far. And finally, Alice outputs $g_L(x_L, x_R)$ and Bob outputs $g_R(x_L, x_R)$. We consider a special subclass of α-leaky split-state function where Alice and Bob simply make a bounded number of adaptive queries to each other's input. In particular, when Alice (resp. Bob) makes a query to x_R (resp. x_L), Alice (resp. Bob) sends location $i \in [k]$ and ask Bob (resp. Alice) to send back the ith coordinate of x_R (resp. x_L).

Note that the communication cost for each query (and answer) is $\lceil \log k \rceil + 1$ bits. (For the remainder of this section, we will assume that both k and n are powers of 2.) So if both Alice and Bob make at most $\alpha k/(2(\log k + 1))$ queries to each other's input, the total communication is at most αk and g is an α-leaky split-state function. We will show that our reduction reduces decision tree tampering functions (of appropriate depth) to this subclass of leaky split-state functions.

Our reduction relies on a ramp secret sharing scheme over binary alphabet. For parameter m, let $(\mathrm{Enc}_{RSS}, \mathrm{Dec}_{RSS})$ be an efficient coding scheme such that Enc_{RSS} maps an m-bit to a (random) $4m$-bit string so that for any $\zeta \in \{0, 1\}^m$ and subset $S \neq \emptyset$ of size at most m, $\mathrm{Enc}_{RSS}(\zeta)_S$ distributes uniformly and randomly over $\{0, 1\}^{|S|}$. As observed by Ball et al. [BDG+18] (see Lemma 2), such a coding scheme can be constructed efficiently from any linear error correcting code from m bits to $4m$ bits with minimal distance $m + 1$. We choose constant 4 to simplify the presentation.

Based on $(\mathrm{Enc}_{RSS}, \mathrm{Dec}_{RSS})$, we define a coding scheme that hides a message in random locations. Let $G_{k,n}$ be the function that given a short "seed," ζ, of k distinct indices in $[n]$ expands it to the corresponding n-bit string with hamming weight k (where the ones are in locations indexed by ζ). Let $D_{k,n}$ be

Given $f = (f_L, f_R) \in \mathrm{DT}(t)$, we sample an α-split state g from the distribution G_f as follows: sample and hardwire the randomness required for $\mathrm{Enc}_L, \mathrm{Enc}_R$. Let ζ_L and ζ_R be the respective seeds used for G in Enc_L and Enc_R. Then, $g = (g_L, g_R)$ is defined as follows: on input $x = (x_L, x_R)$

- $g_L(x)$ simulates $\mathrm{Dec}_L(f_L(\mathrm{Enc}_L(x_L), \mathrm{Enc}_R(x_R)))$ with at most $\alpha k/2(\log k + 1)$ queries to x_R (*)
  1. Decode the tampered seed $\widetilde{\zeta_L}$ by computing the first $4m_L$ outputs of $f_L(\mathrm{Enc}_L(x_L), \mathrm{Enc}_R(x_R))$ then applying Dec_{RSS}.
     (To evaluate the necessary portion f_L the simulator evaluates the corresponding decision trees. When a bit is queried in $\mathrm{Enc}_R(x_R)$ we have three cases (recall that these encodings consist of two parts, $\mathrm{Enc}_{RSS}(\zeta_R)$ and c_R): (a) if the index corresponds to a location in $\mathrm{Enc}_{RSS}(\zeta_R)$, it is hardwired already in g_L; (b) if the index corresponds to c_R and a location specified by ζ_R, query/request the relevant bit from x_R; (c) if the index corresponds to c_R and a location *not* specified by ζ_R, simply use 0.)
  2. Compute the output bits of $f_L(\mathrm{Enc}_L(x_L), \mathrm{Enc}_R(x_R))$ indexed by set $G(\widetilde{\zeta_L})$.
     (The decision trees corresponding to indices specified by $\widetilde{\zeta_L}$ are evaluated identically to the preceeding step.)

  (*) whenever $g_L(x)$ makes more than $\alpha k/(2(\log k + 1))$ queries to x_R, abort and output 0^k.
- $g_R(x)$ simulates $\mathrm{Dec}_R(f_R(\mathrm{Enc}_L(x_L), \mathrm{Enc}_R(x_R)))$ with at most $\alpha k/2(\log k + 1)$ queries to x_L (*)
  1. Compute $\widetilde{y_R} = f_R(\mathrm{Enc}_L(x_L), \mathrm{Enc}_R(x_R))$.
     (The corresponding decision trees in f_R are evaluated in a symmetric manner to those needed for g_L.)
  2. Compute $\mathrm{Dec}_R(\widetilde{y_R})$.

  (*) whenever $g_R(x)$ makes more than $\alpha k/(2(\log k + 1))$ queries to x_L, abort and output 0^k.

Fig. 1. Simulator for $(\mathrm{Enc}, \mathrm{Dec})$

a distribution such that $G_{k,n}(D_{k,n})$ is uniform over n-bit strings with hamming weight k. Note that $G_{k,n}$ can be computed efficiently and $D_{k,n}$ can be sampled efficiently by simply sampling k locations from $[n]$ without replacement. For k, n and $m \geq k \log n$, we define an encoding $\mathrm{Enc}^*_{n,k,m} : \{0,1\}^k \to \{0,1\}^{4m} \times \{0,1\}^n$, as follows: on a k-bit string x, sample a random seed $\zeta \leftarrow D_{k,n}$ for $G_{k,n}$, and output $(\mathrm{Enc}_{RSS}(\zeta), c)$ so that c is 0 everywhere except $c_{G(\zeta)} = x$. $\mathrm{Dec}^*_{n,k,m}$ is defined in the straightforward way (it first decodes ζ using Dec_{RSS}, then outputs $c_{G(\zeta)}$).

The high level idea of our reduction is to use two copies of Enc^* to hide inputs x_L and x_R independently inside two long messages (with different length). Let n_L, m_L, n_R, m_R be parameters to be determined later. We define Enc as $\mathrm{Enc}(x_L, x_R) = (\mathrm{Enc}_L(x_L), \mathrm{Enc}_R(x_R))$ where $\mathrm{Enc}_L = \mathrm{Enc}^*_{n_L,k,m_L}$ and $\mathrm{Enc}_R =$

$\mathrm{Enc}^*_{n_R,k,m_R}$. And we define Dec as $\mathrm{Dec}(y_L, y_R) = (\mathrm{Dec}_L(y_L), \mathrm{Dec}_R(y_R))$ where $\mathrm{Dec}_L = \mathrm{Dec}^*_{n_L,k,m_L}$ and $\mathrm{Dec}_R = \mathrm{Dec}^*_{n_R,k,m_R}$.

Now we show $(\mathrm{Enc}, \mathrm{Dec})$ non-malleably reduces $\mathrm{DT}(t)$ to α-SS_k and prove Lemma 3. First observe that $\mathrm{Dec} \circ \mathrm{Enc}$ is the identity function due to the correctness of $(\mathrm{Enc}_L, \mathrm{Dec}_L)$ and $(\mathrm{Enc}_R, \mathrm{Dec}_R)$. It remains to show that for any $f\colon \{0,1\}^n \to \{0,1\}^n \in \mathrm{DT}(t)$, $\mathrm{Dec} \circ f \circ \mathrm{Enc}$ becomes α-split state functions. In fact, in Figure 1, we reduce decision trees to a simpler subclass of α-SS_k where Alice and Bob, in parallel, make at most $\alpha k/(2(\log k + 1))$-bounded number of adaptive queries to locations of the other party's inputs, then output tampered values.

By the special condition $(*)$ in g_L, g_R, $g = (g_L, g_R)$ is indeed α-split state function because Alice and Bob communicates at most $\log k + 1$ bits per query (including the answer). Moreover, for any x, $G_f(x)$ distributes identically to $\mathrm{Dec} \circ f \circ \mathrm{Enc}(x)$ conditioning on that $(*)$ doesn't happen. Therefore ε, the difference between the simulation and the real experiment, is at most the probability that $(*)$ happens. To bound the event that $(*)$ happens, we begin by proving the following more general proposition.

Proposition 1. *For integers $n, k, m \geq k \log n$. Let A be an arbitrary algorithm that makes at most m adaptive queries to $(\mathrm{Enc}_{RSS}(\zeta), G(\zeta))$. Let Y denote the number of distinct 1's in $G(\zeta)$ which are queried by A. It holds that over the randomness of ζ and Enc_{RSS},*

$$\Pr[Y \geq 2mk/n] \leq \exp(-mk/3n).$$

Proof. Note that, for any fixed ζ, any A that makes at most m adaptive queries cannot distinguish $(\mathrm{Enc}_{RSS}(\zeta), G(\zeta))$ and $(U, G(\zeta))$ where U is uniformly distributed over $\{0,1\}^{4m}$. That's because $(U, G(\zeta))$ generates any possible transcript $(i_1, b_1, \ldots, i_m, b_m)$, $(\mathrm{Enc}_{RSS}(\zeta), G(\zeta))$ with exactly the same probability due to the secrecy property of Enc_{RSS}. Because U and ζ are independent, it suffices to bound the probability that $A^{U,G(\zeta)}$ queries more than $2mk/n$ number of 1's for an arbitrary fixed choice of U and a random ζ.

Without loss of generality, we assume A queries m distinct locations of $G(\zeta)$ because any algorithm can be made into one which sees more ones from $G(\zeta)$ by querying distinct locations. Let $Y_1, \ldots, Y_m$ be indicators that $G(\zeta)$ returns 1 for these m queries made by A. Note that $Y = Y_1 + \cdots + Y_m$ and $\mathrm{E}[Y] \leq mk/n$. In addition, observe that for any $b_1, \ldots, b_m \in \{0,1\}$, $\Pr[\forall i \in [m], Y_i = b_i] = \binom{n-m}{k-|b|_0}/\binom{n}{k}$. It follows that for any set $S \subseteq [m]$, $\mathrm{E}[\mathit{\Pi}_{i \in S} Y_i] = \binom{n-|S|}{k-|S|}/\binom{n}{k} \leq (k/n)^{|S|}$. By the generalized Chernoff bound by Theorem 7 with $\delta = k/n$, we obtain the desired conclusion.

We then apply proposition 1 in following two claims to bound (*), the probability that the number of bits required from the opposite side exceeds some threshold for either half of the simulated tampering function. We will handle each side separately. In particular, these claims will bound the number of queries or probes made to bits on the opposite side that depend on the input (if we fix the randomness of encoding). Because in the simulated tampering, both Alice and Bob jointly know the randomness of encoding, if a bit on the opposite side is not dependent on the input, then both Alice and Bob know this and do not need to request its value. So, in order to complete the proof we only need to bound queries the simulator makes to the opposite side that additionally correspond to locations specified by the respective ζ (in the respective Enc^*).

Claim. Suppose $m_R \geq (4m_L + k)t$, then for any $x \in \{0,1\}^{2k}$, the event that g_L makes more than $2(4m_L + k)tk/n_R$ queries to x_R happens with probability at most $\exp(-(4m_L + k)tk/3n_R)$.

Proof. Fix any x_L, x_R and the randomness for $\mathrm{Enc}_L(x_L)$. Note that Dec_L reads at most $4m_L + k$ coordinates from its input, $4m_L$ to reconstruct the tampered "seed" $\tilde{\zeta}$ and then the at most k locations specified by $G(\tilde{\zeta})$. Each decision tree tampering one of these bits makes at most t queries $\mathrm{Enc}_R(x_R)$ (it makes at most t queries total). Therefore g_L, in order to simulate the tampering of the bits Dec_L requires, makes at most $(4m_L+k)t$ queries to $\mathrm{Enc}_R(x_R) = (\mathrm{Enc}_{RSS}(\zeta_R), G(\zeta_R))$. By Proposition 1, g_L queries more than $2(4m_L+k)tk/n_R$ locations happens with probability at most $\exp(-(4m_L + k)tk/3n_R)$.

Claim. Suppose $m_L \geq t$, then for any $x \in \{0,1\}^{2k}$, the event that g_R makes more than $2(4m_R + n_R)tk/n_L$ queries to x_L happens with probability at most $(n_R + 4m_R)t/m_L \cdot \exp(-m_L k/3n_L)$.

Proof. Fix any x_L, x_R and the randomness for $\mathrm{Enc}_R(x_R)$. Note that any subset of m_L/t outputs of f_R makes at most m_L queries to $\mathrm{Enc}_L(x_L)$. By Proposition 1, the probability that $2m_L k/n_L$ ones in $G(\zeta_L)$ are queried is at most $\exp(-m_L k/3n_L)$. We partition the output bits of f_R into $(n_R + 4m_R)t/m_L$ disjoint subsets of size m_L/t. By a union bound over these subsets, the even that f_R makes more than $2(4m_R + n_R)tk/n_L$ queries to x_L happens with probability at most $(n_R + 4m_R)t/m_L \cdot \exp(-m_L k/3n_L)$. The number of queries made by g_R to x_L is at most the queries made by f_R and the desired conclusion follows.

Then for fixed α, there exists constants c_1, c_2, c_3 (only dependent on α) such that if we set $m_L = c_1 k \log n, n_R = m_R = c_2 tk \log n \log k$ and $n_L = c_3 t^2 k \log n \log^2 k$, then (*) (the event that the number of queries to either opposing side exceeds $\alpha k/(2(\log k + 1))$ happens with probability at most $(t^2 \log k) \cdot \exp(-\Omega(k/t^2 \log^2 k))$. Note that it follows that the rate is $k/n = \Omega(1/t^2 \log^3 n)$ and for $t = O(n^{1/4}/\log^{3/2} n)$, the error can be simplified to $\exp(-\Omega(n/t^4 \log^5 n))$ because $\Omega(n/t^4 \log^5 n) = \Omega(\log n)$ and $t^2 \log k = \exp(O(\log n))$.

4 Leaky Split-State to Split-State Model

In this section, we show how to reduce leaky split-state to split-state non-malleability. In other words, we show how to add leakage-resilience generically to any split-state non-malleable code. Our construction handles up to $\frac{1}{4}$ fraction leakage. Concretely, we prove the following lemma, where the tampering classes SS_k (split-state) and α-SS_n (leaky split-state) are defined in Definition 3 and 4 respectively.

Lemma 4. *For any constant $\alpha \in [0, 1/4)$, α-SS_n non-malleably reduces to SS_k with loss $\exp(-\Omega(n)))$ and constant rate.*

Our main tool is new notion of (information-theoretic, one-time) leakage-resilient encryption defined below.

Definition 11 (Leakage-Resilient Encryption). *We consider a (randomized) encryption scheme* $(\mathsf{Encrypt}, \mathsf{Decrypt})$ *which encrypts message x of length $|x| = k$ using a* key *of size* $|\mathsf{key}| = m$. *For some message $x \in \{0,1\}^k$ we consider the following randomized experiment* $\mathsf{Game}^{\mathsf{LRENC}}(x)$*:*

- *Choose* $\mathsf{key} \leftarrow \{0,1\}^m$, $\mathsf{ct} \leftarrow \mathsf{Encrypt}(\mathsf{key}, x)$.
- *Alice gets* ct *and Bob gets* key. *They can run an arbitrary protocol with each other subject to the total communication being at most ℓ_1 bits. Let* $\mathsf{trans} \in \{0,1\}^{\ell_1}$ *be the transcript.*
- *At the end of the protocol, Alice also outputs an additional value* $\mathsf{aux} \in \{0,1\}^{\ell_2}$.
- *The output of the game is* $\mathsf{key}, \mathsf{trans}, \mathsf{aux}$.

We say that an encryption scheme is $(\ell_1, \ell_2, \varepsilon)$-leakage-resilient if for any adversarial strategy of Alice and Bob and for any x_0, x_1 the outputs of $\mathsf{Game}^{\mathsf{LRENC}}(x_0)$ *and* $\mathsf{Game}^{\mathsf{LRENC}}(x_1)$ *have statistical distance at most ε.*

The above definition is similar to the "forward-secure storage" of Dziembowski [Dzi06], which corresponds to our notion with $\ell_1 = 0$ (there is only leakage on the ciphertext; it is completely independent of the key but can be much larger than the key). It is also similar to the notion of "leakage-resilient storage" of Davi, Dziembowski and Venturi [DDV10], which corresponds to our notion with $\ell_2 = 0$ (there is back-and-forth leakage on the ciphertext and the key but the total leakage is smaller than either the ciphertext or the key). Our definition combines the two notions. We will crucially rely on a setting of parameters where, if the key size is m and the message size is k then we need $\ell_1 < m \leq \ell_2 < k$. In other words, we allow ℓ_1 bits of back-and-forth leakage between the ciphertext and the key where ℓ_1 is smaller than either component, but then allow and additional ℓ_2 bits of leakage on the ciphertext where ℓ_2 is larger than the key.

Reduction via Leakage-Resilient Encryption. We first show how to use leakage-resilient encryption as defined above to construct a reduction from leaky split-state to split-state tampering. We do so by encrypting the two states x_L, x_R using leakage-resilient encryption and storing the key with the other state (i.e., the key used to encrypt the left state is stored on the right sides and vice versa). Intuitively, the leakage-resilient encryption ensures that the leakage is independent of the actual states x_L, x_R. However, we face the challenge that, by tampering the key on the right side we can influence how the left side is decrypted and vice versa. We get around this by thinking of the tampered keys as additional leakage (aux).

Let $\mathcal{E} = (\mathsf{Encrypt}, \mathsf{Decrypt})$ be a leakage-resilient encryption with message size k and key length m. We define our reduction $(\mathrm{Enc}, \mathrm{Dec})$ below:

- $\mathrm{Enc}(x_L, x_R)$: Sample $\mathsf{key}_L \leftarrow \{0,1\}^m, \mathsf{ct}_L \leftarrow \mathsf{Encrypt}(x_L)$, $\mathsf{key}_R \leftarrow \{0,1\}^m$, $\mathsf{ct}_R \leftarrow \mathsf{Encrypt}(x_R)$. Output (y_L, y_R) where $y_L = (\mathsf{ct}_L, \mathsf{key}_R), y_R = (\mathsf{ct}_R, \mathsf{key}_L)$.
- $\mathrm{Dec}(y_L, y_R)$: Parse $y_L = (\mathsf{ct}_L, \mathsf{key}_R), y_R = (\mathsf{ct}_R, \mathsf{key}_L)$ and output $x_L = \mathsf{Decrypt}(\mathsf{key}_L, \mathsf{ct}_L)$ and $x_R = \mathsf{Decrypt}(\mathsf{key}_R, \mathsf{ct}_R)$.

Lemma 5. *Assume $\mathcal{E}$ is an $(\ell_1, \ell_2, \varepsilon)$-leakage-resilient encryption with message length k, key length $m \leq \ell_2$ and ciphertext length c. Then* $(\mathrm{Enc}, \mathrm{Dec})$ *defined above is a 2ε-non-malleable reduction from $(\ell_1/(c+m))$-leaky split-state to split-state. For a messages (x_L, x_R) of length $2k$, the resulting codeword* $\mathrm{Enc}(x_L, x_R)$ *has length* $2(c+m)$

Proof. Consider the following game: $\mathsf{Game}^{\mathsf{NM}}(x_L, x_R)$:

1. Compute $(y_L = (\mathsf{ct}_L, \mathsf{key}_R), y_R = (\mathsf{ct}_R, \mathsf{key}_L)) \leftarrow \mathrm{Enc}(x_L, y_L)$
2. Give Alice y_L and Bob y_R. They can run an arbitrary protocol with each other subject to the total communication being at most ℓ_1 bits. Let $\mathsf{trans} \in \{0,1\}^{\ell_1}$ be the transcript.
3. At the end of the protocol Alice outputs $y'_L = (\mathsf{ct}'_L, \mathsf{key}'_R)$ and Bob outputs $y'_R = (\mathsf{ct}'_R, \mathsf{key}'_L)$.
4. The output of the game $(x'_L, x'_R) = \mathrm{Dec}(y'_L, y'_R)$,
so that $x'_L = \mathsf{Decrypt}(\mathsf{key}'_L, \mathsf{ct}'_L)$ and $x'_R = \mathsf{Decrypt}(\mathsf{key}'_R, \mathsf{ct}'_R)$.

To prove the Lemma, we fix an arbitrary strategy of Alice and Bob and need to show that there exist some distribution G over functions (g_L, g_R) such that for every x_L, x_R the output of $\mathsf{Game}^{\mathsf{NM}}(x_L, x_R)$ is 2ε-statistically close to $g_L(x_L), g_R(x_R)$ where $(g_L, g_R) \leftarrow G$.

Let us define the distribution $z \leftarrow D(x_L, x_R)$ to be the distribution of the values

$$z = (\mathsf{key}_L, \mathsf{key}_R, \mathsf{trans}, \mathsf{key}'_L, \mathsf{key}'_R)$$

in the context of $\mathsf{Game}^{\mathsf{NM}}(x_L, x_R)$. We make two observations, which we then combine to prove our lemma.

Observation 1. In $\mathsf{Game}^{\mathsf{NM}}(x_L, x_R)$, if we condition on some particular choice of $z \leftarrow D(x_L, x_R)$, then the distribution of x'_L a is completely independent of (x_R, x'_R) and similarly x'_R is independent of (x_L, x'_L). In particular, we can define the randomized process g^z_L which has z hard-coded and samples $x'_L \leftarrow g^z_L(x_L)$ by first sampling Alice's view in the game conditioned on z and x_L, computing her output ct'_L and setting $x'_L = \mathsf{Decrypt}(\mathsf{key}'_L, \mathsf{ct}'_L)$. We can define the randomized process $x'_R \leftarrow g^z_R(x_R)$ analogously. It is easy to see that the distribution of $\mathsf{Game}^{\mathsf{NM}}(x_L, x_R)$ is then identical to sampling $z \leftarrow D(x_L, x_R)$ and outputting $x'_L \leftarrow g^z_L(x_L), x'_R \leftarrow g^z_R(x_R)$.

Observation 2. For any x_L, x_R the distribution of $D(x_L, x_R)$ is 2ε-statistically close to $D(0^k, 0^k)$. We argue that this holds via two steps. We first argue that $D(x_L, x_R)$ is ε-close to $D(0^k, x_R)$ and then argue that $D(0^k, x_R)$ is ε-close to $D(0^k, 0^k)$. For the first step, we can fix any worst case choice of $\mathsf{key}_R, \mathsf{ct}_R$ and use the security of leakage-resilient encryption to argue that the joint distribution of $\mathsf{key}_L, \mathsf{trans}, \mathsf{key}'_R$ is ε-close between $D(x_L, x_R)$ and $D(0^k, x_R)$; we set $\mathsf{aux} = \mathsf{key}'_R$ in this argument and use the fact that $|\mathsf{key}'_R| = m \leq \ell_2$. We then note that key'_L is just a function of $\mathsf{ct}_R, \mathsf{key}_L$ and trans and therefore the total distribution of $(\mathsf{key}_L, \mathsf{key}_R, \mathsf{trans}, \mathsf{key}'_L, \mathsf{key}'_R)$ is ε close between $D(x_L, x_R)$ and $D(0^k, x_R)$. The argument that $D(0^k, x_R)$ is ε-close to $D(0^k, 0^k)$ is identical.

By combining observations 1 and 2, we see the distribution of $\mathsf{Game}^{\mathsf{NM}}(x_L, x_R)$ is 2ε statistically close to sampling $z \leftarrow D(0^k, 0^k)$ and outputting $x'_L \leftarrow g^z_L(x_L)$, $x'_R \leftarrow g^z_R(x_R)$. This concludes our reduction as desired; we define the distribution G over functions (g_L, g_R) by sampling $z \leftarrow D(0^k, 0^k)$ and setting $g_L = g^z_L$ and $g_R = g^z_R$. The output of $\mathsf{Game}^{\mathsf{NM}}(x_L, x_R)$ is 2ε-statistically close to $g_L(x_L), g_R(x_R)$ where $(g_L, g_R) \leftarrow G$.

Construction of Leakage-Resilient Encryption. Let Ext be a seeded strong extractor with r-bit source, d-bit seed and output size k which is $(r - \ell_1 - \ell_2, \varepsilon_1)$-secure. Let $2\mathsf{Ext}$ be a strong two-source extractor with m-bit sources and d-bit output which is $(2m - \ell_1, \varepsilon_2)$-secure.

Define the scheme $(\mathsf{Encrypt}, \mathsf{Decrypt})$ as follows:

- $\mathsf{Encrypt}(\mathsf{key}, x)$:
  Choose $u \leftarrow \{0,1\}^r, y \leftarrow \{0,1\}^m, s = 2\mathsf{Ext}(\mathsf{key}, y), z = \mathsf{Ext}(u; s) \oplus x$.
  Output $\mathsf{ct} = (u, y, z)$.
- $\mathsf{Decrypt}(\mathsf{key}, \mathsf{ct} = (u, y, z))$:
  Compute $s = 2\mathsf{Ext}(\mathsf{key}, y)$ and output $z \oplus \mathsf{Ext}(u; s)$.

Claim. Consider a variant of the leakage-resilient encryption game, which we denote "weak leakage resilience", where Alice does not get the z part of the ciphertext during the game but the output of the game is $\mathsf{key}, \mathsf{trans}, \mathsf{aux}, z$. If the scheme is $(\ell_1, \ell_2, \varepsilon)$-"weak leakage resilient" then it also satisfies $(\ell_1, \ell_2, \varepsilon \cdot 2^k)$-leakage resilience.

Proof. Assume there exists some (Alice, Bob, Distinguisher) strategy in the original game such that the Distinguisher has an $\varepsilon 2^k$ advantage in distinguishing the game with x_0 and x_1. We convert this into an (Alice', Bob, Distinguisher') strategy for the weak game by guessing a random value $v \leftarrow \{0,1\}^k$ at the beginning of the game and having Alice' run Alice with v in place of z. Then Distinguisher' gets z and if $v = z$ it runs the original Distinguisher else outputs 0. It's easy to see that the advantage of Distinguisher' is the same as that of Distinguisher when $v = z$, which happens with probability 2^{-k}, and 0 otherwise. Therefore, Distinguisher' has advantage 2^{-k} smaller than Distinguisher which proves the claim.

Lemma 6. $(\mathsf{Encrypt}, \mathsf{Decrypt})$ *is* $(\ell_1, \ell_2, (\varepsilon_1 + \varepsilon_2)2^{k+1})$*-leakage-resilient.*

Proof. It suffices to show that the scheme is $(\ell_1, \ell_2, 2(\varepsilon_1 + \varepsilon_2))$ weak leakage resilient and the rest follows by the preceding claim. We use a statistical hybird argument.

Hybrid 1: This is the weak leakage resilient game with message x_0. Recall that in the game Alice gets (u, y) and Bob gets key. They run a protocol with ℓ_1 bits of communication resulting in transcript trans. At the termination of the protocol, Bob also outputs an additional ℓ_2-bit value aux. The output of the protocol is $\mathsf{key}, \mathsf{trans}, \mathsf{aux}, z$ where $z = \mathsf{Ext}(u; s) \oplus x_0$ and $s = 2\mathsf{Ext}(\mathsf{key}, y)$.

Hybrid 2: Note that, in Hybrid 1, conditioned on the random variable $V_1 = (s, y, \mathsf{trans})$ the random variables $V_2 = (u, \mathsf{aux}, z)$ and $V_3 = \mathsf{key}$ are independent. Therefore, we can define Hybrid 2 to run the same game as Hybrid 1, which defines (V_1, V_2, V_3), but then, instead of key, output a freshly re-sampled key' from the correct distribution of V_3 conditioned on V_1. This is distributed identically to Hybrid 1.

Hybrid 3: In this hybrid, we choose s uniformly at random instead of $s = 2\mathsf{Ext}(\mathsf{key}, y)$ and set $z = \mathsf{Ext}(r; s) \oplus x_0$. We still sample key' from the same distribution of V_3 conditioned on V_1 at the end of the game, just like in Hybrid 2.
The statistical distance between Hybrid 2 and Hybrid 3 is ε_2. We rely on the fact that $2\mathsf{Ext}$ is a strong extractor and that trans amounts to ℓ_1 bits of entropy loss from key to argue that, even given (u, y, trans) the value s is ε_2-close to uniform.

Hybrid 4: In this hybrid, we set z to uniform instead of $z = \mathsf{Ext}(u; s) \oplus x_0$.
The statistical distance between Hybrid 3 and Hybrid 4 is ε_1. This follows from the strong-extractor property of Ext and the fact that $\mathsf{trans}, \mathsf{aux}$ gives $\ell_2 + \ell_1$ bits of leakage on u.

Hybrid 5,6,7: Are the same as 3,2,1 with x_0 replaced by x_1. In particular Hybrid 7 is the weak leakage resilience game with message x_1.
Hybrids 4,5 are ε_1 close (same argument as Hybrids 3,4), Hybrids 5,6 are ε_2 close (same argument as Hybrids 2,3) and Hybrids 6,7 are identical (same argument as 1,2).

Combining the above we get a total distance of $2(\varepsilon_1 + \varepsilon_2)$ between Hybrids 1 and 7 as we wanted to show.

We can now plug in parameters using the inner-product two-source extractor [CG88], and the strong extractor [GUV09] to prove the main Lemma of this section:

Proof (Proof of Lemma 4). For $\varepsilon \in (0, 1)$, let $\varepsilon_1 = \varepsilon_2 = \varepsilon/2^{k+3}$ and $\ell_2 = m$. By Theorem 5, there exists some constant $c_1, c_2 \geq 1$ and explicit Ext such that for $r - \ell_1 - \ell_2 \geq c_1 \cdot k$, Ext can extract k bits from $(r, r - \ell_1 - \ell_2)$ source with $d = c_2 \log(r/\varepsilon_1)$-bit seed and error ε_1. By Theorem 6, for $2m - \ell_1 \geq m + d + 2\log(1/\varepsilon_2)$, there exists explicit $2\mathsf{Ext}$ that extracts d bits with error ε_2 from m-bit sources with entropy $2m - \ell_1$. Plugging in Ext and $2\mathsf{Ext}$, by Lemma 6 and Lemma 5, we obtain $(\ell_1, \ell_2, (\varepsilon_1+\varepsilon_2)2^{k+1})$ leakage-resilient encryption and a $(\varepsilon_1+\varepsilon_2)2^{k+2}$-non-malleable reduction from (ℓ_1/n)-split state to split state with $n = r + 2m + k$. By setting $r = \ell_1 + m + c_2 k$, $m = \ell_1 + d + 2\log(1/\varepsilon_2)$ and $d = c_2(\log(r/\varepsilon_1))$, we obtain that $n \le 4\ell_1 + c_3(k + \log 1/\varepsilon)$ for some constant $c_3 \ge 1$. Therefore for any $\alpha < 1/4$ and $\ell_1 = \alpha n$, we can set $n = \Theta(\frac{k+\log(1/\varepsilon)}{1/4-\alpha})$. The desired conclusion follows from setting $\varepsilon = \exp(-\Omega(k))$ and $n = \Theta(k)$.

5 Putting things Together

Combining Lemma 3 and Lemma 4, we obtain a non-malleable reduction from decision trees to split-state model.

Lemma 7. *For $t = O(n^{1/4}/\log^{3/2} n)$, there is a $(\mathrm{DT}(t) \Rightarrow \mathrm{SS}_k, \varepsilon)$ non-malleable reduction with rate $\Omega(1/t^2 \log^3 n)$ where $\varepsilon \le \exp(-\Omega(n/t^4 \log^5 n))$.*

Plugging in the construction of non-malleable codes for split state model [Li18] with rate $\Omega(\log\log n/\log n)$ and error $\exp(-\Omega(n \log\log n/\log n))$, we obtain our main theorem.

Theorem 8. *For any $t = O(n^{1/4}/\log^{3/2} n)$, there is an explicit and efficient non-malleable code that is unconditionally secure against depth-t decision trees with codeword length $n = O(kt^2 \log^4 n/\log\log n)$ and error $\exp(-\Omega(n/t^4 \log^5 n))$ for a k-bit message.*

Ball et al. [BDG+18] gave a non-malleable reduction from small-depth circuits to a leaky variant of decision trees, $\mathrm{LL}^{d,m,n}[\mathrm{DT}(t)]$ (See definition 12).

Lemma 8. *[BDG+18] For $S, d, n, t \in \mathbb{N}, p, \delta \in (0,1)$, there exist*

$$\sigma = \mathrm{poly}(t, \log(2^t S), \log(1/\delta), \log(1/p))^{11}$$

and $m = O(\sigma \log n)$ such that, for any $2m \le k \le n(p/4)^d$,

$$(\mathrm{AC}_d(S) \implies \mathrm{LL}^{d,m,n}[\mathrm{DT}(t)], d\varepsilon)$$

where

$$\varepsilon = nS\left(2^{2t+1}(5pt)^t + \delta\right) + \exp\left(-\tfrac{\sigma}{2\log(1/p)}\right).$$

Ball et al. [BDG+18] used the fact that for $t < \log n$, leaky depth-t decision trees is a subclass of leaky 2^t-local functions and gave a non-malleable code for leaky local functions based on a construction of Ball et al. [BDKM16]. Their approach only works for $t < \log n$. This limits the error of the composed code to be $n^{-O(\log n)}$ which, in turns, requires $S = n^{O(\log n)}$. (The same restrictions also appear in [CL17], but for other reasons.)

[11] The exponent of this polynomial is a fixed absolute constant independent of all other parameters.

We note that the "leakage" is simply a restricted form of dm adaptive queries to depth-t decision trees. Thus, $\mathrm{LL}^{d,m,n}[\mathrm{DT}(t)] \subseteq \mathrm{DT}(dmt)$. Therefore a non-malleable code for large depth decision trees immediately yields a new non-malleable code for small depth circuits (with improved error). In particular, $\mathrm{LL}^{d,m,n}[\mathrm{DT}(t)]$ gives decision trees that are identical excepting (up to) the last t queries before output (and that the last t-queries must be consistent with one of n depth-t decision trees). Combining Lemma 8 and our new non-malleable reduction from decision trees to split-state functions, we obtain an improved non-malleable reduction from small-depth circuits to split-state functions.

Lemma 9. *For $S, d, n, t \in \mathbb{N}, p, \delta \in (0,1)$, there exist*

$$\sigma = \mathrm{poly}(t, \log(2^t S), \log(1/\delta), \log(1/p))$$

and $m = O(\sigma \log n)$ such that, for $t' = dmt = O(n^{1/4}/\log^{3/2} n)$, $k \geq O(\sigma \log n)$ and $k = \Omega(n(p/4)^d/(t')^2 \log^3 n)$,

$$\left(\mathrm{AC}_d(S) \implies \mathrm{SS}_k, d\varepsilon + \exp(-\Omega(n/(t')^4 \log^5 n))\right)$$

where

$$\varepsilon = nS\left(2^{2t'+1}(5pt')^{t'} + \delta\right) + \exp(-\frac{\sigma}{2\log(1/p)}).$$

For constant-depth polynomial-size circuits (i.e.,AC^0), by setting $p = n^{-O(1/d)}$ (such as $n^{-1/100d}$), $t' = 1/40p$ and $\delta = \exp(n^{-\Omega(1/d)})$, we improve the error of the non-malleable reduction from AC^0 to split-state from $n^{-\log n}$ to $\exp(-n^{\Omega(1)})$.

Corollary 1. $\left(\mathsf{AC}^0 \implies \mathrm{SS}_k, \exp(-n^{\Omega(1)})\right)$ *for $n = k^{1+c}$ for a constant $0 < c < 1$.*

The same setting of parameters lead to non-malleable reduction for circuits of depth as large as $\Theta(\log(n)/\log\log(n))$ and size $S = \exp(n^{O(1/d)})$ with error $\exp(-n^{\Omega(1/d)})$. Combining the non-malleable code for split state from [Li18] with rate $\Omega(\log\log n/\log n)$ and error $\exp(-\Omega(n \log\log n/\log n))$, we obtain our main theorem.

Theorem 9. *For any constant $c \in (0,1)$, there exist constants $c_1, c_2 \in (0,1)$ such that for any $d \leq c_1 \log n/\log\log n$ and $S = \exp(n^{c_2/d})$ there is an explicit, efficient, information theoretic non-malleable code for depth d size S circuits with error $\exp(-n^{\Omega(1/d)})$ and encoding length $n = k^{1+c}$.*

Acknowledgements

We would like to thank Dana Dachman-Soled, Tal Malkin, and Li Yang Tan for many insightful conversations and helping to pose the initial question and its connections to small depth circuits. We would like to additionally thank Justin Holmgren and Ron Rothblum for stimulating discussions. The first author is supported by an IBM Research PhD Fellowship, NSF grant CCF1423306, and the Leona M. & Harry B. Helmsley Charitable Trust. Part of this work was completed while the author was visiting IDC Herzilya. The second author is supported by NSF grants CNS1314722 and CNS-1413964. The third author is supported by NSF grants CNS-1314722, CNS-1413964, CNS-1750795 and the Alfred P. Sloan Research Fellowship.

References

AB16. Divesh Aggarwal and Jop Briët. Revisiting the sanders-bogolyubov-ruzsa theorem in fp^{n} and its application to non-malleable codes. In *IEEE International Symposium on Information Theory, ISIT 2016, Barcelona, Spain, July 10-15, 2016*, pages 1322–1326, 2016.

ADKO15a. Divesh Aggarwal, Yevgeniy Dodis, Tomasz Kazana, and Maciej Obremski. Non-malleable reductions and applications. In Rocco A. Servedio and Ronitt Rubinfeld, editors, *Proceedings of the Forty-Seventh Annual ACM on Symposium on Theory of Computing, STOC 2015, Portland, OR, USA, June 14-17, 2015*, pages 459–468. ACM, 2015.

ADKO15b. Divesh Aggarwal, Stefan Dziembowski, Tomasz Kazana, and Maciej Obremski. Leakage-resilient non-malleable codes. In *Theory of Cryptography - 12th Theory of Cryptography Conference, TCC 2015, Warsaw, Poland, March 23-25, 2015, Proceedings, Part I*, pages 398–426, 2015.

ADL14. Divesh Aggarwal, Yevgeniy Dodis, and Shachar Lovett. Non-malleable codes from additive combinatorics. In David B. Shmoys, editor, *Symposium on Theory of Computing, STOC 2014, New York, NY, USA, May 31 - June 03, 2014*, pages 774–783. ACM, 2014.

Agg15. Divesh Aggarwal. Affine-evasive sets modulo a prime. *Inf. Process. Lett.*, 115(2):382–385, 2015.

AGM$^+$15. Shashank Agrawal, Divya Gupta, Hemanta K. Maji, Omkant Pandey, and Manoj Prabhakaran. Explicit non-malleable codes against bit-wise tampering and permutations. In *Advances in Cryptology - CRYPTO 2015 - 35th Annual Cryptology Conference, Santa Barbara, CA, USA, August 16-20, 2015, Proceedings, Part I*, pages 538–557, 2015.

BDG$^+$18. Marshall Ball, Dana Dachman-Soled, Siyao Guo, Tal Malkin, and Li-Yang Tan. Non-malleable codes for small-depth circuits. *IACR Cryptology ePrint Archive*, 2018:207, 2018.

BDKM16. Marshall Ball, Dana Dachman-Soled, Mukul Kulkarni, and Tal Malkin. Non-malleable codes for bounded depth, bounded fan-in circuits. In *Advances in Cryptology - EUROCRYPT 2016 - 35th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Vienna, Austria, May 8-12, 2016, Proceedings, Part II*, pages 881–908, 2016.

BDKM18. Marshall Ball, Dana Dachman-Soled, Mukul Kulkarni, and Tal Malkin. Non-malleable codes from average-case hardness: AC0, decision trees, and streaming space-bounded tampering. In Jesper Buus Nielsen and Vincent Rijmen, editors, *Advances in Cryptology - EUROCRYPT 2018 - 37th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Tel Aviv, Israel, April 29 - May 3, 2018 Proceedings, Part III*, volume 10822 of *Lecture Notes in Computer Science*, pages 618–650. Springer, 2018.

BDSK$^+$18. Marshall Ball, Dana Dachman-Soled, Mukul Kulkarni, Huijia Lin, and Tal Malkin. Non-malleable codes against bounded polynomial time tampering. Cryptology ePrint Archive, Report 2018/1015, 2018. `https://eprint.iacr.org/2018/1015`.

CG88. Benny Chor and Oded Goldreich. Unbiased bits from sources of weak randomness and probabilistic communication complexity. *SIAM J. Comput.*, 17(2):230–261, 1988.

CG14. Mahdi Cheraghchi and Venkatesan Guruswami. Non-malleable coding against bit-wise and split-state tampering. In Yehuda Lindell, editor, *Theory of Cryptography - 11th Theory of Cryptography Conference, TCC 2014, San Diego, CA, USA, February 24-26, 2014. Proceedings*, volume 8349 of *Lecture Notes in Computer Science*, pages 440–464. Springer, 2014.

CGL16. Eshan Chattopadhyay, Vipul Goyal, and Xin Li. Non-malleable extractors and codes, with their many tampered extensions. In *Proceedings of the 48th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2016, Cambridge, MA, USA, June 18-21, 2016*, pages 285–298, 2016.

CL17. Eshan Chattopadhyay and Xin Li. Non-malleable codes and extractors for small-depth circuits, and affine functions. In *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2017, Montreal, QC, Canada, June 19-23, 2017*, pages 1171–1184, 2017.

CL18. Eshan Chattopadhyay and Xin Li. Non-malleable extractors and codes in the interleaved split-state model and more. *CoRR*, abs/1804.05228, 2018.

CZ14. Eshan Chattopadhyay and David Zuckerman. Non-malleable codes against constant split-state tampering. In *55th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2014, Philadelphia, PA, USA, October 18-21, 2014*, pages 306–315. IEEE Computer Society, 2014.

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

DKO13. Stefan Dziembowski, Tomasz Kazana, and Maciej Obremski. Non-malleable codes from two-source extractors. In Ran Canetti and Juan A. Garay, editors, *Advances in Cryptology - CRYPTO 2013 - 33rd Annual Cryptology Conference, Santa Barbara, CA, USA, August 18-22, 2013. Proceedings, Part II*, volume 8043 of *Lecture Notes in Computer Science*, pages 239–257. Springer, 2013.

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

Dzi06. Stefan Dziembowski. On forward-secure storage. In *Advances in Cryptology - CRYPTO 2006, 26th Annual International Cryptology Conference, Santa Barbara, California, USA, August 20-24, 2006, Proceedings*, pages 251–270, 2006.

GUV09. Venkatesan Guruswami, Christopher Umans, and Salil P. Vadhan. Unbalanced expanders and randomness extractors from parvaresh-vardy codes. *J. ACM*, 56(4):20:1–20:34, 2009.

KOS17. Bhavana Kanukurthi, Sai Lakshmi Bhavana Obbattu, and Sruthi Sekar. Four-state non-malleable codes with explicit constant rate. In *Theory of*

Cryptography - 15th International Conference, TCC 2017, Baltimore, MD, USA, November 12-15, 2017, Proceedings, Part II, pages 344–375, 2017.

Li17. Xin Li. Improved non-malleable extractors, non-malleable codes and independent source extractors. In Hamed Hatami, Pierre McKenzie, and Valerie King, editors, *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2017, Montreal, QC, Canada, June 19-23, 2017*, pages 1144–1156. ACM, 2017.

Li18. Xin Li. Pseudorandom correlation breakers, independence preserving mergers and their applications. *Electronic Colloquium on Computational Complexity (ECCC)*, 25:28, 2018.

NZ96. Noam Nisan and David Zuckerman. Randomness is linear in space. *J. Comput. Syst. Sci.*, 52(1):43–52, 1996.

PS97. Alessandro Panconesi and Aravind Srinivasan. Randomized distributed edge coloring via an extension of the chernoff-hoeffding bounds. *SIAM J. Comput.*, 26(2):350–368, 1997.

A Leaky Function Classes

Ball et al. [BDG$^+$18] considered a leaky variant of a given tampering class $\mathcal{C}$.

Definition 12 (Leaky Function Families). *[BDG$^+$18] Let $\mathrm{LL}^{i,m,N}[\mathcal{C}]$ denote tampering functions generated via the following game:*

1. *The adversary first commits to N functions from a class $\mathcal{C}$, $F_1, \ldots, F_N = \boldsymbol{F}$. (Note: $F_j : \{0,1\}^N \to \{0,1\}$ for all $j \in [N]$.)*
2. *The adversary then has i-adaptive rounds of leakage. In each round $j \in [i]$,*
   - *the adversary selects s indices from $[N]$, denoted S_j,*
   - *the adversary receives $\boldsymbol{F}(x)_{S_j}$.*

   Formally, we take $h_j : \{0,1\}^{m(j-1)} \to [N]^m$ to be the selection function such that
$$h_j(F(X)_{S_1}, \ldots, F(X)_{S_{j-1}}) = S_j.$$
   Let h_1 be the constant function that outputs S_1.
3. *Finally, selects a sequence of n functions $(F_{t_1}, \ldots, F_{t_n})$ ($T = \{t_1, \ldots, t_n\} \subseteq [N]$ such that $t_1 < t_2 < \cdots < t_n$) to tamper with.*
   Formally, we take $h : \{0,1\}^{mi} \to [N]^n$ such that $h(F(X)_{S_1}, \ldots, F(X)_{S_i}) = T$.

Thus, any $\tau \in \mathrm{LL}^{i,m,N}[\mathcal{C}]$ can be described as $(\boldsymbol{F}, h_1, \cdots, h_i, h)$ and denote the tampering function described above via $\tau = \mathrm{Eval}(\boldsymbol{F}, h_1, \cdots, h_i, h)$.