Leakage-resilient coin tossing

Leakage-Resilient Coin Tossing

Elette Boyle* MIT
Shafi Goldwasser† MIT and Weizmann
Yael Tauman Kalai MSR New England

June 21, 2011

Abstract

The ability to collectively toss a common coin among n parties in the presence of faults is an important primitive in the arsenal of randomized distributed protocols. In the case of dishonest majority, it was shown to be impossible to achieve less than $\frac{1}{r}$ bias in $O(r)$ rounds (Cleve STOC '86). In the case of honest majority, in contrast, unconditionally secure $O(1)$-round protocols for generating common perfectly *unbiased* coins follow from general completeness theorems on multi-party secure protocols in the perfectly secure channels model (e.g., BGW, CCD STOC '88).

However, in the multi-party protocols with faulty minority, parties need to generate and hold local secret values which are assumed to be *perfectly hidden* from malicious parties: an assumption which is crucial to proving the resulting common coin is unbiased. This assumption unfortunately does not seem to hold in practice, as attackers can launch side-channel attacks on the local state of honest parties and leak information on their secrets.

In this work, we present an $O(1)$-round protocol for collectively generating an unbiased common coin, in the presence of leakage on the local state of the honest parties. We tolerate $t \leq (\frac{1}{3} - \epsilon)n$ computationally-unbounded Byzantine faults and in addition a $\Omega(1)$-fraction leakage on each (honest) party's secret state. Our results hold in the memory leakage model (of Akavia, Goldwasser, Vaikuntanathan '08) adapted to the distributed setting.

Another contribution of our work is a tool we use to achieve collective coin flipping – *leakage-resilient verifiable secret sharing*. Informally, this is a variant of ordinary VSS in which secrecy guarantees are maintained even if information is leaked on individual shares of the secret.

*Supported by NDSEG graduate fellowship.

†This work was supported in part by Trustworthy Computing: NSF CCF-1018064 and PROCEED: DARPA FA8750-11-2-0225.

1 Introduction

Randomization, and the the ability to keep your local randomness and local state private, are fundamental ingredients at the disposal of fault tolerant distributed algorithms. This was realized originating with the work of Rabin [Rab83] introducing the power of a shared global common coin to obtain a dramatic reduction in round complexity with respect to Ben-Or's asynchronous randomized consensus algorithm [Ben83][1]; and continued to be utilized in many beautiful distributed algorithms to this day in various network models: synchronous and asynchronous, faulty majority and faulty minority, private channels and full information, and networks with and without broadcast channels.

The assumption that a party's local state – including its local randomness and the values of its secret cryptographic keys – is perfectly hidden from an adversary, is an assumption that has undergone much scrutiny in the past few years in the cryptographic community. This is in light of accumulating evidence which shows that in practice physical measurements (so called side-channel attacks) can be made on honest parties' devices, which result in leakage from their local state that can completely compromise the security of the cryptographic algorithm. Indeed, a considerable amount of effort in the cryptographic community is devoted today to develop new cryptographic schemes which are resistant to leakage (e.g., [Riv97, CDH$^+$00, ISW03, MR04, DP08, AGV09, NS09, KV09, BKKV10, DHLW10] and many more). Several models of leakage have been considered. The one most relevant to this work is that an adversary can adaptively choose polynomial time length-shrinking *leakage* functions and receive the value of the leakage functions on the secret state of the device [AGV09].

We propose to mirror this line of work in the regime of distributed fault tolerant algorithms. Namely, to address the question of how leakage from the local state of non-faulty parties affects the correctness of fault-tolerant distributed protocols. Here, in addition to the fact that some of the parties are faulty and fully compromised, the adversary who is coordinating the action of the faulty parties can obtain partial information in the form of leakage on the local state of all honest parties. This potentially may enable the adversary to alter the course of the protocol's execution. We note that in this context, the coordinating adversary can adaptively choose the leakage function, depending on the history of communication it sees thus far.

Randomized distributed algorithms address many different tasks. In some tasks, such as Byzantine agreement, leader election, and collective coin tossing, the parties have no secret inputs and the emphasis is on getting the *correct distribution over the outputs* rather than on input privacy. In other tasks such as secure distributed function evaluation, both perfect input privacy and the correctness of output distribution are required.

In this paper, we focus on the output correctness aspect of distributed protocols in the presence of leakage attacks. In particular, we provide a fault-tolerant and leakage-resilient protocol for collective unbiased coin tossing among n parties.

The problem of collective coin tossing in a distributed setting has received a lot of prior attention, starting with the work of Rabin [Rab83] on distributed consensus. When there is no honest majority of parties, results from the two-party setting by [Cle86] showed that a bias of $\frac{1}{r}$ must be incurred by any $O(r)$-round protocol (this was recently shown optimal in a work of Moran et al. [MNS09]). Loosely speaking, the problem is that a dishonest party can bias the output by doing the following: At the last round, before sending his final message, he can compute the outcome, and abort if he does not favor this outcome, thus biasing the output. When there is an honest majority of parties, this attack can be prevented using *verifiable secret sharing* (VSS), a notion defined by Chor, Goldwasser, Micali, and Awerbuch [CGMA85]. Verifiable secret sharing allows each of the n parties to toss a coin locally and share it among the n parties. After all the local coins have been shared via a VSS, the parties reconstruct the coin. The output coin is set to be the result of simply taking the exclusive or of the local coins. The works of ([BGW88, CCD88]) on secure multi-party computation show how to achieve VSS in expected $O(1)$ rounds, and thus how to construct an unbiased coin tossing protocol that runs in $O(1)$ rounds. These results ([BGW88, CCD88]) hold unconditionally in the synchronous network

[1] Ben-Or's ingenious protocol does not require the local coin outcomes to remain ever private. All that is required of the coin is to be random. Alas, the number of rounds is exponential.

model with less than a third Byzantine faults, assuming perfectly secure channels of communication between pairs of users and the use of a broadcast channel.

However, each of these protocols require the parties to generate and hold secret values, and security is guaranteed only under the assumption that these secrets are *completely* hidden from the adversarial view. It is easy to check that these protocols are no longer secure if the adversary obtains some partial information about these secrets. This is the starting point of our work.

Before we go on to state our results, we remark that the more general problem of multi-party secure function evaluation (SFE), which addresses both correctness and privacy of parties' inputs, runs into immediate definitional problems in the presence of leakage attacks. Since leakage on the private inputs is available to the adversary, it is impossible to meet the SFE problem specification as is, since they require the inputs of honest parties to remain private beyond what is revealed by the SFE output value. Possible ways out of this conundrum may be to relax the attack model to allow some form of a leak-free pre-processing phase, or to relax the security guarantees of an SFE. We do not explore these relaxations here.

1.1 Our Results

We construct a leakage-resilient collective coin-tossing protocol in synchronous point-to-point networks with a broadcast channel and secure communication channels between pairs of parties.

We allow up to one third colluding statically corrupted malicious parties. Namely, a computationally unbounded rushing adversary sees the internal state of all corrupted parties and can set the messages of these parties at any round, as a function of all honest parties' messages up to (and including) this round. In addition, the adversary can make *leakage queries* at every round in the form of specifying a leakage function, and obtain the result of the leakage function applied to each honest parties internal state.

We allow the adversary to leak *arbitrary* functions of each party's secret state, as long as the total number of bits leaked from each party is at most some (pre-specified) λ fraction of its entire secret state.[2] Each leakage query is applied to the secret state of a single party. Since participants of a distributed protocol typically run on different physical hardware (and reside in different locations), we believe that it is reasonable to assume each leakage query modeling a physical measurement reveals information about each party's execution separately. To maximize generality within this setting, we allow the leakage queries on different parties' secret states to be interleaved (i.e., leak some from party i then some from party j, and then some more from party i), and the choice of leakage queries to be adaptively selected as a function of prior leakage. We remark that this distributed leakage model is similar to the earlier model used by Akavia et al. [AGH10] in their work on public-key encryption in which the secret key of the decryption algorithm is distributed among two parties.

Informally, let us call a protocol (t, λ)-leakage-resilient if it achieves its desired functionality in presence of an adversary who can control up to t parties and can request leakage queries for functions which output up to a λ-fraction of the internal secret state of each honest party (as above). We can now state our main theorem, to be formally stated within the paper.

Theorem For any constants $\delta, \epsilon > 0$, any $\lambda \leq \frac{\delta(1-\epsilon)}{10+6\delta}$, any $n \geq (3+\delta)t$, and any m, there exists a (t, λ)-leakage-resilient n-party distributed protocol that generates a value $v \in \{0,1\}^m$, and terminates in $O(1)$ rounds satisfying:

- **Agreement:** At the conclusion of the protocol, each party outputs a value $v_i \in \{0,1\}^m$. For all honest parties P_i, P_j, it holds that $v_i = v_j$.
- **Randomness:** With all but negligible probability (in n), the distribution of the honest output value v is exponentially close to uniform in $\{0,1\}^m$.

A few remarks are in order.

[2] Our methods extend to also tolerate the Naor and Segev [NS09] leakage model which allows leakage functions which are not necessarily shrinking but leave the internal local state with enough min-entropy.

Fairness: We emphasize that our protocol achieves fairness, in the sense that even if the dishonest parties abort prematurely, the honest parties will output a random string.

Strings versus Bits: We emphasize that the output of our coin tossing protocol can be a long random string, as opposed to just a single bit. In the leak-free setting, this point is not worth emphasizing, since the coin-tossing protocol can be run in parallel to output as many bits as desired. However, in the leaky setting, if we run many protocols in parallel, our leakage bounds may deteriorate very quickly: if we run k protocols, where each protocol can tolerate leakage of up to a λ fraction, then in the resulting parallel execution, the leakage rate becomes only $\frac{\lambda}{k}$. Thus, to maintain our leakage bounds we would need to run the protocol sequentially, resulting in many rounds of communication. Our protocol has the property that it can output as many bits as desired in constant rounds with constant leakage rate.

Weakening the Secure Channels Assumption: Note that our leakage model immediately implies we can tolerate leakage of information from the secure communication channels, as parties' messages are computed as a function of public information and their personal secret state. To remove the secure channels assumption altogether, we would need to send the messages between honest parties using encryption, which would necessitate a computational assumption supporting the strength of the encryption algorithm. Furthermore, one would have to consider whether leakage from the secret keys of the decryption algorithm and the randomness used by the encryption algorithm can be tolerated.[3]

Relation to Using Imperfect Random Sources in Distributed Computing: The question of achieving $O(1)$-round Byzantine Agreement and multi-party computation when parties do not have access to perfect local randomness, but rather to independent imperfect random sources such as min-entropy sources [GSV05, KLRZ08, KLR09], seems strongly related to our work here. Indeed, one may naturally view a random secret with leakage as a secret a-priori drawn from a min-entropy source. The crucial difference between these works and our own is that our leakage model allows the adversary to leak adaptively *during* the protocol, as opposed to non-adaptively before the protocol begins. More specifically, the approach taken in [GSV05, KLRZ08, KLR09] is to first generate *truly* random strings from the weak random sources, and then to use these random strings in the underlying protocol execution. This approach will not work in our setting, since the adversary can simply choose to leak on the newly generated random strings.

On the other hand, we note that the works of [GSV05, KLRZ08, KLR09] consider randomness coming from an *arbitrary* min-entropy distribution, whereas our model considers perfect randomness that is being leaked so as to leave min-entropy in the distribution.

Coin Flipping versus Byzantine Agreement: Achieving a weak form of collective coin tossing was in itself an important building block to construct Byzantine agreement protocols in many works, most notably in the work of Dwork, Shmoys and Stockmeyer [DSS90], and the work of Feldman Micali [FM85]. Our schemes to construct collective coin tossing utilize broadcast channels as a primitive (which are equivalent to Byzantine agreement), and thus obviously cannot be used to construct Byzantine agreement. It is an interesting question for future research how to achieve coin tossing in the presence of leakage without assuming broadcast channels.

Using Collective Coin Tossing to Force Honest Behavior: An important usage of coin tossing dates back to the work of Goldwasser and Micali [GM82] on mental poker protocols. Here they propose the idea of forcing the local coins used by parties in a protocol to be the result of a common coin toss, to ensure that parties do not rig their coins. In this case, the result of the collective coin toss will be known only to one party Alice, and yet all other parties will be able to verify (via, say, zero-knowledge

[3] A very recent theorem by Bitansky, Canetti, and Halevi [BCH11] shows that by sending messages using non-committing encryption (introduced by Canetti, Feige, Goldreich and Naor [CFGN96]), protocols in the secure channel model can be transformed into leakage-resilient secure protocols without assuming secure channels.

protocols) that Alice is using the result of the collective coins in her computations. This idea was later used in the compiler of [GMW87] from the n-party secure function evaluation (SFE) protocol with honest-but-curious majority to an SFE protocol with malicious majority. Our collective coin tossing protocol can similarly be turned into one where only one party Alice knows the result but all other parties can verify that Alice (via, say, a leakage-resilient zero knowledge protocol [BCH11]) is using the result of the collective coins in her computations.

1.1.1 Leakage-Resilient VSS

The main tool we use in our construction, which is of independent interest, is a new *weakly leakage-resilient verifiable secret sharing scheme (WLR-VSS)*. Verifiable secret sharing extends Shamir's [Sha79] secret sharing to ensure not only secrecy (i.e., faulty parties do not have any information about the dealer's secret) and robustness (i.e., honest parties can reconstruct the secret even if faulty parties do not provide their true shares), but also to ensure reconstruction even if the dealer is dishonest. WLR-VSS is a VSS scheme with the additional guarantee that given the view of any adversary who corrupts up to t parties *and leaks* λ-fraction of each honest party's secret state, the secret still retains a constant fraction of its original entropy. We refer to this property as weak leakage resilience.

We are now ready to state our second main theorem.

Theorem (WLR-VSS): Let $n = (3+\delta)t$ for some constant $\delta > 0$. Then for any constants $\epsilon < 1$ and $\lambda \leq \frac{\delta(1-\epsilon)}{10+6\delta}$, there exists a VSS protocol $(\mathsf{Share}, \mathsf{Reconstruct})$ that runs in $O(1)$ rounds and tolerates t malicious parties *and* up to λ-fraction leakage of the secret state of each (honest) party, with the following modified secrecy guarantee:

> If the dealer is honest during the sharing phase, then for any adversary $\mathcal{A}$ controlling up to t parties and leaking up to λ fraction of each (honest) party' secret state, with overwhelming probability over $y \leftarrow \mathsf{view}_{\mathcal{A}}(S)$ it holds that $H_\infty(S|\mathsf{view}_{\mathcal{A}}(S) = y) \geq \epsilon H_\infty(S)$, where $\mathsf{view}_{\mathcal{A}}(s)$ denotes the view of $\mathcal{A}$ at the conclusion of the sharing phase of the protocol using secret s.

In addition, we define and obtain a stronger version of leakage-resilient VSS, with the requirement that the secret looks *uniform* even if all the shares were partially leaked. We do not need this stronger version for our collective coin tossing application; however, we believe that it is of independent interest. We refer the reader to Section 4.3 for details.

1.2 Overview of Our Solution

Let us first see why simple and known common coin tossing protocols are not resilient to leakage. Consider the following well-known coin tossing protocol paradigm: First, each party P_i chooses a random value r_i and secret shares it to all other parties using a *verifiable secret sharing* (VSS) protocol. Then, all the parties reveal their shares and reconstruct $r_1, \ldots, r_n$. Finally, the parties output $\oplus r_i$. This protocol is not resilient to leakage for several reasons.

First, the reduction from coin tossing to VSS fails. For example, a malicious party P_j can simply leak from each party P_i the least significant bit of r_i, and then choose r_j such that the xor of these least significant bits is zero. This way, P_j can bias the output string so that the least significant bit will always be zero. So, the problem is that in the leaky setting, we cannot claim that the r_i's look random to the adversary. Instead, all we can claim is that they have high min-entropy in the view of the adversary. To address this first problem, the first idea is to use a multi-source extractor instead of the xor function. Namely, output $\mathsf{Ext}(r_1, \ldots, r_n)$, where Ext is an extractor that takes n independent sources and outputs a string that is statistically close to uniform. Note however, that we cannot use any such multi-source extractor, since some of the sources (i.e., some of the r_j's) may be chosen maliciously. Thus, what we need is a multi-source extractor that outputs a (statistically close to) uniform string,

even if some of the sources are arbitrary, but independent of the "honest" sources. Indeed, such an extractor was constructed by Kamp, Rao, Vadhan and Zuckerman [KRVZ06].

Secondly, VSS protocols by and large are not resilient to leakage. Consider a single VSS protocol execution in the above paradigm. If the adversary leaks λ-fraction from each share, the total number of bits leaked is too large (indeed, potentially larger than the size of the secret being shared), and we cannot even guarantee that the secret r_i still has any min-entropy left. Thus, we cannot use any VSS scheme, but rather we need to use a *leakage-resilient* one, with the guarantee that even if λ-fraction of each share is leaked, the secret still has high min-entropy. Indeed, we construct such a weakly leakage-resilient VSS in Section 4.2 (as informally defined earlier in this introduction). We note that many distributed protocols use VSS protocols, which immediately make them susceptible to leakage. Thus, we are hopeful that our leakage-resilient VSS scheme may be used in other protocols as well.

Finally, two technical difficulties remain. In the above coin-tossing paradigm utilizing VSS (or even a weakly leakage-resilient WLR-VSS), each party shares his random value with all other n parties, and thus each honest party holds information on *all* secret values r_i. Since the leakage is computed on a party's entire secret state, by leaking from honest parties, the adversary may learn information on the joint distribution of the r_i's. But, this creates a dependency issue: recall that the output of the multi-source extractor is only guaranteed to be random if the inputs r_i are independent. Further, in this paradigm the secret state of each party will be quite large, consisting of n secret shares (one for each secret value r_i). This can potentially yield poor leakage bounds, with leakage rate less than $\frac{1}{n}$ if we want to ensure no share of one particular secret can be entirely leaked.

We avoid these problems by ensuring that each of the n parties will never hold more than one share of any of the random values r_i. To this end, we follow a two-step approach. The first step is a universe reduction idea similar to the one going back to Bracha [Bra84]. Instead of having *all* parties generate and secret share random strings r_i, we elect a small committee $\mathcal{E}$ (of size approximately $\log^2 n$), and only the members of $\mathcal{E}$ choose a random string r_i which will be shared via WLR-VSS (and later used in the construction of the collective coin). We utilize Feige's protocol [Fei99] to elect this committee, which guarantees with high probability that the fraction of faulty parties in the elected committee is the same as in the global network. The second idea is that members of this committee do not WLR-VSS the r_i they chose to all n parties, but rather each WLR-VSS his r_i to a small secondary committee. Namely, for every party $i \in \mathcal{E}$ all parties elect a secondary subcommittee $\mathcal{E}_i$, and party P_i will WLR-VSS her random string r_i only to parties in $\mathcal{E}_i$. We need to ensure that all the secondary committees $\mathcal{E}_i$ are disjoint, to avoid the case where one party has many shares, and thus will be vulnerable to small leakage rate. One may be tempted to simply force these committees to be disjoint by eliminating members that appear in previous committees. Indeed, we follow this approach. However, care must be taken when eliminating parties, since we may eliminate too many honest parties, and remain with dishonest majority. In Proposition 5.1, we modify Feige's lightest bit leader election protocol [Fei99] to select such disjoint committees, where we carefully choose the parameters so that when eliminating recurring honest parties, we have the guarantee that (with overwhelming probability) we are left with enough honest parties. In particular, we need to ensure that the sub-committees $\mathcal{E}_i$ are not too small. See details in Section 5.

1.3 Paper Organization

In Section 2 we introduce some preliminaries and notation. Section 3 contains a discussion on the leakage model considered in this paper. In Section 4 we define and construct a leakage-resilient verifiable secret sharing scheme, a tool used in our construction. In Section 5 we present a protocol for electing several disjoint committees. Section 6 contains the construction and proof of our leakage-resilient coin tossing protocol.

2 Preliminaries

2.1 Verifiable Secret Sharing

A *secret sharing* protocol, a notion introduced by Shamir [Sha79], is a protocol that allows a dealer who holds a secret input s, to share his secret among n parties. The guarantee is that even if t of the parties are malicious, they gain no information about the secret s. Moreover, the (honest) parties can recover the secret s from all the shares, even if the malicious parties give malicious shares. Namely, the shares form a codeword that can be efficiently decoded even in the presence of errors.

Note that a secret sharing scheme does not give any guarantee for a dishonest dealer. A *verifiable secret sharing* (VSS) scheme, introduced by Chor et al. [CGMA85], is a secret sharing scheme that has the additional guarantee that after the sharing phase, a dishonest dealer is either rejected, or is committed to a single secret s, that the honest parties can later reconstruct.

Definition 2.1 (Verifiable Secret Sharing)**.** *An n-party* VSS protocol tolerating t malicious parties *is a two-phase protocol for parties $\mathcal{P} = \{P_1, ..., P_n\}$, where a distinguished dealer $P^* \in \mathcal{P}$ holds an initial input s, such that the following conditions hold for any adversary controlling at most t parties:*

- **Reconstruction:** *Even if the dealer is dishonest, at the end of the sharing phase, the joint view of the honest parties defines a value s' (which can be computed in polynomial time from this view) such that at the end of the reconstruction phase, all honest parties will output s'.*
- **Validity:** *If the dealer is honest, then $s' = s$.*
- **Secrecy:** *If the dealer is honest during the sharing phase, then at the end of this phase the joint view of the malicious parties is independent of the dealer's input s.*

2.2 Multi-Source Randomness Extractors

A multi-source randomness extractor is a deterministic function which takes as input independent sources, each with sufficient amount of entropy, and outputs a string that is statistically close to uniform. The notion of entropy that is used is *min-entropy*.

Definition 2.2. *A random variable $X \subseteq \{0,1\}^n$ is said to have min entropy k, denoted by $H_\infty(X) = k$, if for every $x \in \{0,1\}^n$, $\Pr[X = x] \leq \frac{1}{2^k}$. It is said to have min-entropy rate α if $H_\infty(X) \geq \alpha n$.*

2.2.1 Two-Source Extractors

Constructions of two-source extractors are given, for example, by Raz [Raz05] and Bourgain [Bou05]. We use the following simplified version of the Bourgain result in the construction of strongly leakage-resilient VSS in Section 4.3.

Theorem 2.3 ([Bou05])**.** *There exists a polynomial-time computable two-source extractor $\mathsf{Ext}_2 : (\{0,1\}^d)^2 \to \{0,1\}^m$ that takes as input two independent sources X and Y and outputs an m-bit string that is ϵ-close to uniform, as long as $H_\infty(X), H_\infty(Y) \geq \frac{1}{2}d$, and where $m = \Omega(d)$ and $\epsilon = 2^{-\Omega(d)}$.*

2.2.2 Robust Multi-Source Extractors

A multi-source extractor is an extractor that takes as input several independent sources, each with sufficient amount of entropy, and outputs a string that is statistically close to uniform. In this work, we need such a multi-source extractor that extracts randomness even if some of the sources are "malicious," but independent of the "honest" ones.[4] Such an extractor, which we refer to as a robust multi-source extractor, was constructed by Kamp, Rao, Vadhan and Zuckerman [KRVZ06].

[4]Note that if the malicious sources may depend on the honest ones, then such (deterministic) extractors do not exist.

Theorem 2.4 ([KRVZ06])**.** *For any constant $\delta > 0$ and every $n \in \mathbb{N}$, there is a polynomial-time computable robust multi-source extractor $\mathsf{Ext} : \left(\{0,1\}^d\right)^n \to \{0,1\}^m$ that takes as input n independent sources, each in $\{0,1\}^d$, and produces an m-bit string that is ϵ-close to uniform, as long as the min-entropy rate of the combined sources is δ, and where $m = 0.99\delta nd$ and $\epsilon = 2^{-\Omega((nd)/\log^3(nd))}$.*

2.3 Leakage Lemma

In this section, we prove a lemma regarding entropy loss due to leakage.

Lemma 2.5. *Let X be a random variable and L a leakage function on the support of X. Then for sufficiently large n,*

$$\Pr_{y \leftarrow L(X)}\left[H_\infty(X|L(X)=y) \leq H_\infty(X) - |L(X)| - \log^2 n\right] = negl(n).$$

Proof. Define the set $\mathsf{Bad} = \{y \in L(X) : H_\infty(X|L(X)=y) < H_\infty(X) - |L(X)| - \log^2 n\}$. Suppose for contradiction $\Pr_{y\leftarrow L(X)}[y \in \mathsf{Bad}] > \frac{1}{n^c}$ for some constant $c > 0$. Comparing to the *average min-entropy* $\tilde{H}_\infty()$, as defined by Dodis et al. [DORS08], we have

$$\begin{aligned}
\tilde{H}_\infty(X|L(X)) &:= -\log \mathbb{E}_{y\leftarrow L(X)} 2^{-H_\infty(X|L(X)=y)} \\
&= -\log\Big(\Pr[y \in \mathsf{Bad}] \cdot \mathbb{E}_{y\leftarrow \mathsf{Bad}} 2^{-H_\infty(X|L(X)=y)} + \Pr[y \notin \mathsf{Bad}] \cdot \mathbb{E}_{y\leftarrow L(X)\setminus\mathsf{Bad}} 2^{-H_\infty(X|L(X)=y)}\Big) \\
&\leq -\log\Big(\Pr[y \in \mathsf{Bad}] \cdot \mathbb{E}_{y\leftarrow \mathsf{Bad}} 2^{-H_\infty(X|L(X)=y)}\Big) \\
&\leq -\log\Big(\frac{1}{n^c} \cdot 2^{-(H_\infty(X) - |L(X)| - \log^2 n)}\Big) \\
&= c\log n + H_\infty(X) - |L(X)| - \log^2 n \\
&< H_\infty(X) - |L(X)|.
\end{aligned}$$

But, it is shown in [DORS08] that $\tilde{H}_\infty(X|L(X)) \geq H_\infty(X) - |L(X)|$, yielding a contradiction. Thus, the lemma holds.

□

2.4 Feige Committee Election Protocol

Our leakage-resilient coin flipping protocol uses Feige's lightest bin committee election protocol as a subroutine [Fei99]. Feige's protocol gives a method for selecting a committee of approximately k parties for a given parameter k.[5] It consists of one round, in which each party chooses and broadcasts a random bin in $\left[\frac{n}{k}\right]$. The committee consists of the parties in the lightest bin.

Lemma 2.6 (Feige)**.** *For any constant $\beta > 0$ and any $k < n$, Feige's lightest bin protocol is a 1-round public-coin protocol for electing a committee $\mathcal{E}$ such that for any set of corrupted parties $\mathcal{C} \subset [n]$ of size $t = \beta n$,*

1. $|\mathcal{E}| \leq k$,
2. $\Pr[|\mathcal{E} \setminus \mathcal{C}| \leq (1-\beta-\epsilon)k] < \frac{n}{k} e^{-\frac{\epsilon^2 k}{2(1-\beta)}}$ $\forall$ *constant* $\epsilon > 0$,
3. $\Pr\left[\frac{|\mathcal{E}\cap\mathcal{C}|}{|\mathcal{E}|} \geq \beta + \epsilon\right] < \frac{n}{k} e^{-\frac{\epsilon^2 k}{2(1-\beta)}}$ $\forall$ *constant* $\epsilon > 0$.

[5] In Feige's original work [Fei99], he considered the specific case of $k = \log n$. For our purpose, we need to elect larger committees (to achieve negligible error), and thus we consider general k.

Proof. We first note that by the pigeonhole principle, the lightest bin necessarily contains no more than $n/(\frac{n}{k}) = k$ parties, implying property (1).

For each bin b and honest party i, we define the indicator variable $X_{i,b}$ to be 1 if and only if party i selects bin b. Since we consider only honest parties, this is a Bernoulli random variable with $p = \frac{k}{n}$. For a particular bin b, we can now bound the probability that few honest parties selected this bin as compared to the expected value $(1-\beta)k$.

$$\begin{aligned}\Pr\left[\sum_{i\notin\mathcal{C}} X_{i,b} < (1-\beta-\epsilon)k\right] &= \Pr\left[\sum_{i\notin\mathcal{C}} X_{i,b} < \left(1-\frac{\epsilon}{1-\beta}\right)(1-\beta)\,k\right] \\ &< e^{-(1-\beta)k(\frac{\epsilon}{1-\beta})^2/2} \\ &= e^{-\frac{\epsilon^2 k}{2(1-\beta)}},\end{aligned}$$

where the second inequality holds by a Chernoff bound.[6] This proves property (2). Now, taking a union bound, the probability that *any* bin b has fewer than $(1-\beta-\epsilon)k$ honest parties will be

$$\Pr[\exists \text{ Bin } b : \sum_{i\notin\mathcal{C}} X_{i,b} < (1-\beta-\epsilon)k] < \frac{n}{k} e^{-\frac{\epsilon^2 k}{2(1-\beta)}},$$

proving property (2). Finally, combining properties (1) and (2), we have that with probability $1 - \frac{n}{k}e^{-\frac{\epsilon^2 k}{2(1-\beta)}}$,

$$\frac{|\mathcal{E}\cap\mathcal{C}|}{|\mathcal{E}|} = 1 - \frac{|\mathcal{E}\setminus\mathcal{C}|}{|\mathcal{E}|} \le 1 - \frac{(1-\beta-\epsilon)k}{k} = \beta+\epsilon,$$

implying property (3). □

Corollary 2.7. *Feige's lightest bin protocol for $k = \log^2 n$ is a 1-round public-coin protocol such that for any set of corrupted parties $\mathcal{C}$ of size βn, for any constant $\epsilon > 0$, with* overwhelming probability *in n, a committee $\mathcal{E}$ will be elected such that $(1-\beta-\epsilon)\log^2 n \le |\mathcal{E}| \le \log^2 n$ and $|\mathcal{E}\setminus\mathcal{C}| \ge (1-\beta-\epsilon)\log^2 n$.*

3 Modeling Leakage in Distributed Protocols

We consider synchronous point-to-point networks with a broadcast channel. Point-to-point channels are assumed to be authenticated and to provide partial privacy guarantees (see discussion below). We consider n-party protocols where up to t statically corrupted parties perform arbitrary malicious faults. More precisely, we consider a computationally unbounded adversary who sees the internal state of all corrupted parties and controls their actions. We also assume the adversary is *rushing*, i.e. in any round he can wait until all honest parties send their messages before selecting the messages of corrupted parties. Our results hold information theoretically, with no computational assumptions.

In this work we propose a strengthening of the standard model, where in addition the adversary is able to *leak* a constant fraction of information on the secret state of each (honest) party. We model this by allowing the adversary to adaptively make leakage queries (i, f) throughout the protocol, where $i \in [n]$ and $f : \{0,1\}^* \to \{0,1\}$, and giving him the evaluation of f on the secret state of party i. Note that this also captures leakage on communication channels, as parties' messages are computed as a function of public information and their personal secret state; thus, we do not need to assume fully private channels, but rather channels that achieve privacy with bounded information leakage.

For simplicity, we consider length-bounded leakage. Namely, we require that no more than $\lambda|\mathsf{state}_i|$ leakage queries can be made on any single party i's secret state for some leakage rate λ, where $|\mathsf{state}_i|$ denotes the maximal size of the secret state of party i at any given time during the protocol. But, our

[6] Exact Chernoff bound used: For $X_1, ..., X_n$ independent Bernoulli random variables and $\mu = \mathbb{E}[\sum_i X_i]$, then for $0 < \delta < 1$, it holds that $\Pr[\sum_i X_i < (1-\delta)\mu] < e^{-\mu\delta^2/2}$.

constructions work equally well in the more general model of [NS09] where the output length of the leakage on state_i is not restricted, as long as the entropy of state_i is decreased by no more than the fraction λ.

Note that in this model, each leakage query is applied to the secret state of a single party. Since participants of a distributed protocol typically run on different physical hardware (and in fact in many cases in different locations across the world), it is reasonable to assume each physical attack reveals information about one party's execution. To maximize generality within this setting, we allow leakage queries on different parties' secret states to be intermingled (i.e., leak some from party i then some from party j, and then some more from party i), and to be adaptively selected as a function of prior leakage.

We say that a distributed protocol is λ *leakage resilient* if its original properties are satisfied even against this type of strengthened leakage adversary. In this paper, we will focus on constructing a leakage-resilient unbiased coin tossing protocol.

Definition 3.1 (Leakage-Resilient Distributed Coin Tossing)**.** *A protocol for parties $\mathcal{P} = \{P_1, ..., P_n\}$ is a λ* leakage-resilient *m*-bit distributed coin tossing protocol tolerating *t* malicious parties *if the following conditions hold for any adversary controlling at most t parties and leaking up to λ-fraction of the secret state of each (honest) party:*

- **Agreement:** *At the conclusion of the protocol, each party outputs a value $v_i \in \{0,1\}^m$. For all honest parties P_i, P_j, it holds that $v_i = v_j$.*
- **Randomness:** *With overwhelming probability in n (even if malicious parties abort prematurely), the distribution of the honest output value v given the view of the adversary is statistically close to uniform in $\{0,1\}^m$.*

4 Verifiable Secret Sharing with Leakage

One of the subroutines in our leakage-resilient coin tossing protocol is a protocol achieving verifiable secret sharing (VSS) in the presence of leakage. Recall the standard VSS guarantee is that for any adversary $\mathcal{A}$ who corrupts up to t parties, a dishonest dealer is committed to a single secret which will be reconstructed by honest parties, and the secret input s of an honest dealer retains full entropy given the view of $\mathcal{A}$ at the conclusion of the sharing phase. For our purposes, we will need stronger guarantees, where for any adversary $\mathcal{A}$ who corrupts up to t parties *and leaks* λ-fraction of each honest party's secret state, the VSS reconstruction property still holds, and the secret input s of an honest dealer retains a constant fraction of its original entropy given the entire view of $\mathcal{A}$ (including leakage). We refer to this property as weak leakage resilience.

In Section 4.1, we show that a modified version of the Shamir secret sharing scheme [Sha79] satisfies a notion of weak leakage resilience. In Section 4.2, we use this underlying secret sharing scheme to construct a weakly leakage-resilient VSS protocol by incorporating a method of verifying that the dealer has distributed good shares.

We note that one can define a stronger version of leakage-resilient VSS, with the requirement that the secret looks *uniform* even if all the shares are partially leaked. Although this stronger version is not required for our coin-tossing construction, we believe that it is of independent interest. We formally define and construct such a protocol in Section 4.3.

4.1 Weakly Leakage-Resilient Secret Sharing

Recall in the standard Shamir secret sharing scheme, to secret share an input s the dealer samples a random degree d polynomial $a_0 + a_1 x + \cdots + a_d x^d \in \mathbb{F}[x]$ such that $a_0 = s$, and generates the shares by evaluating the polynomial at different points. The degree d of the polynomial is typically chosen to be equal to the number of assumed corrupted parties, t.

We modify the standard Shamir secret sharing scheme in two ways. First, we take d to be *strictly greater* than the number of corrupted parties. Second, we take the relative size of the secret to be

larger: instead of s being a single element $s \in \mathbb{F}$ embedded as the single coefficient a_0, we consider secrets $s \in \mathbb{F}^{d-t+1}$ consisting of $d-t+1$ elements, embedded as the first several coefficients $s = a_0||a_1|| \cdots ||a_{d-t}$ (where $||$ denotes concatenation). The reason we embed s only as $a_0, ..., a_{d-t}$ as opposed to all $a_0, ..., a_d$ is to avoid the situation where shares of corrupted parties give information about the secret s. By increasing the degree and making our secrets larger while maintaining the size of each secret share, we can allow a higher fraction of leakage from honest shares without reducing the entropy of s by too much.

This leaves us with the question of how to set d, given values for n and t. The larger the d we choose, the more leakage we will be able to tolerate while maintaining entropy in the secret s. However, we also require the original secret to be recoverable even if t parties reveal incorrect shares. To achieve this we rely on the decoding properties of the Shamir secret sharing scheme when viewed as a Reed-Solomon error-correcting code [MS81]. Namely, we can uniquely (and efficiently) decode any vector of secret shares with up to $\left\lfloor \frac{n-(d+1)}{2} \right\rfloor$ errors. To guarantee decoding of up to t errors, we must have $d \leq n - 2t - 1$. To maximize leakage resilience while maintaining unique decoding, we will thus take $d = n - 2t - 1$.

We now formalize the scheme described above. In future discussions, we will refer to it as the "modified Shamir" secret sharing scheme and denote it as $(\mathsf{SS}, \mathsf{RecSS})$.

- $\mathsf{SS}(n, t, s)$. Let $d = n - 2t - 1$. Split the secret s into $(d - t + 1)$ pieces: $s = a_0|| \cdots ||a_{d-t}$. Choose t random values $a_{d-t+1}, ..., a_d \leftarrow \mathbb{F}$. The secret share for party P_i is the evaluation $f(i)$, where $f(x) = a_0 + a_1 x + \cdots + a_d x^d$.
- $\mathsf{RecSS}(m_1, ..., m_n)$. Use Reed-Solomon decoding on the vector of shares $(m_1, ..., m_n)$ to yield $(a_0, ..., a_d)$. Output $s = a_0|| \cdots ||a_{d-t}$.

Proposition 4.1. *For $n = (3 + \delta)t$, the following properties hold.*

1. *Unique Decoding: Given any vector $(s_1, ..., s_n)$ of distance at most t from a valid codeword corresponding to a secret s, the output of $\mathsf{RecSS}(s_1, ..., s_n)$ will be s.*
2. *Leakage Resilience: For any adversary corrupting up to t parties and leaking a total of ℓ bits on the secret state of honest parties, with overwhelming probability in n, the secret s retains at least $H_\infty(S) - \ell - \log^2 n = \delta t \log |\mathbb{F}| - \ell - \log^2 n$ bits of min-entropy.*

Proof. Property 1 holds immediately from the decoding property of Reed-Solomon codes [MS81]. Property 2 holds since $H_\infty(S) = (d - t + 1) \log |\mathbb{F}| = ((n - 2t - 1) + 1) \log |\mathbb{F}| = \delta t \log |\mathbb{F}|$, together with a standard entropy argument (see Lemma 2.5). □

4.2 Weakly Leakage-Resilient VSS

In our coin tossing protocol, we make use of a VSS protocol satisfying the following notion of weak leakage resilience.

Definition 4.2 (Weakly Leakage-Resilient VSS)**.** *A (λ, ϵ)-weakly leakage-resilient VSS protocol tolerating t malicious parties for parties $\mathcal{P} = \{P_1, ..., P_n\}$ is a protocol with two phases (sharing and reconstruction), where a distinguished dealer $P^* \in \mathcal{P}$ holds an initial input s, such that with overwhelming probability in n, the following conditions hold for any adversary $\mathcal{A}$ controlling at most t parties and leaking up to λ fraction of the secret state of each (honest) party:*

- **Reconstruction:** *Even if the dealer is dishonest, at the end of the sharing phase, the joint view of the honest parties defines a value s' (which can be computed in polynomial time from this view) such that at the end of the reconstruction phase, all honest parties will output s'.*
- **Validity:** *If the dealer is honest, then $s' = s$.*

- **Secrecy:** *If the dealer is honest during the sharing phase, then it holds with overwhelming probability over* $y \leftarrow \mathsf{view}_{\mathcal{A}}(S)$ *that* $H_\infty(S|\mathsf{view}_{\mathcal{A}}(S) = y) \geq \epsilon H_\infty(S)$*, where* $\mathsf{view}_{\mathcal{A}}(s)$ *denotes the view of* $\mathcal{A}$ *at the conclusion of the sharing phase of the protocol using secret* s*.*

We now construct a (λ, ϵ)-weakly leakage-resilient VSS protocol $(\mathsf{Share}_{\mathsf{LR}}, \mathsf{Rec}_{\mathsf{LR}})$, taking inspiration from the VSS construction of [BGW88]. We use as a black box the modified Shamir secret sharing scheme with polynomial degree $d = n - 2t - 1$ (see Section 4.1 above).

Let $\mathbb{F}$ be a field such that $\log |\mathbb{F}| = 2n$. We define the protocol $(\mathsf{Share}_{\mathsf{LR}}, \mathsf{Rec}_{\mathsf{LR}})$ as follows.

- $\mathsf{Share}_{\mathsf{LR}}(s)$:

  1. **Round 1:** The dealer P^* selects two values $r, r' \leftarrow \mathbb{F}^{\delta t}$ uniformly at random, and runs three independent executions of the modified Shamir secret sharing algorithm (as defined in Section 4.1): $(s_1, ..., s_n) \leftarrow \mathsf{SS}(n, t, s)$, $(r_1, ..., r_n) \leftarrow \mathsf{SS}(n, t, r)$, $(r'_1, ..., r'_n) \leftarrow \mathsf{SS}(n, t, r')$. Recall the modified Shamir scheme uses polynomial degree $d = n - 2t - 1$. To each party i, the dealer sends the corresponding three shares s_i, r_i, and r'_i.
  2. **Round 2:** Each party P_i broadcasts three random pairs of bits $\alpha_i, \beta_i, \gamma_i \in \{0, 1\}^2$. Take α, β, γ to be the corresponding elements in $\mathbb{F}$ with bit descriptions $(\alpha_1, ..., \alpha_n)$, $(\beta_1, ..., \beta_n)$, $(\gamma_1, ..., \gamma_n)$. (Recall $\log |\mathbb{F}| = 2n$).
  3. **Round 3:** Each party P_i broadcasts the linear combination of his shares
  $$\alpha s_i + \beta r_i + \gamma r'_i \in \mathbb{F}.$$
  4. **Round 4:** Consider the received vector $v = (v_1, ..., v_n)$, where supposedly $v_i = \alpha s_i + \beta r_i + \gamma r'_i \ \forall i$.
     - If v is a valid codeword (i.e., all points lie on a degree-d polynomial), the dealer is accepted, and the sharing phase concludes.
     - If v is distance $> t$ away from a valid codeword, the dealer is rejected, and the sharing phase concludes.
     - Otherwise, let $D \subset [n]$ be the components i in disagreement with those of the nearest codeword. For each $i \in D$, the dealer P^* broadcasts all three shares s_i, r_i, r'_i. If any linear combination $\alpha s_i + \beta r_i + \gamma r'_i$ with $i \in D$ is inconsistent with the nearest codeword, the dealer is rejected. Otherwise, all parties continue to the next step.
  5. **Rounds 5-6:** Repeat Rounds 2-3. That is, each party broadcasts a new triple of random $\tilde{\alpha}_i, \tilde{\beta}_i, \tilde{\gamma}_i$ and then broadcasts the linear combination $\tilde{v}_i = \tilde{\alpha} s_i + \tilde{\beta} r_i + \tilde{\gamma} r'_i$ of his shares.
  6. **Local Computation:** Consider the new vector $\tilde{v}$ of values received in Round 6, where $\forall i \in D$ we use the values (s, r_i, r'_i) broadcast by the dealer in Round 4.
     - If $\tilde{v}$ is distance $> t$ away from a valid codeword, the dealer is rejected.
     - Otherwise, let $\tilde{D}$ be the set of parties whose components differ from the codeword closest to $\tilde{v}$.
       * If $D \cap \tilde{D} \neq \emptyset$ or $|D \cup \tilde{D}| > t$, then the dealer is rejected.
       * Otherwise, the dealer is accepted.

- $\mathsf{Rec}_{\mathsf{LR}}()$:

  1. **Round 1:** Each party P_i broadcasts his share s_i.
  2. **Local Computation:** Locally, P_i runs the modified Shamir secret sharing reconstruction algorithm $s' \leftarrow \mathsf{RecSS}(s'_1, ..., s'_n)$, where s'_i is the value broadcast by party P_i, and outputs this value s'.

Theorem 4.3. *Let $n = (3+\delta)t$ for some constant $\delta > 0$. Then for any constants $\epsilon < 1$ and $\lambda \leq \frac{\delta(1-\epsilon)}{10+6\delta}$, the protocol $(\mathsf{Share}_{\mathsf{LR}}, \mathsf{Rec}_{\mathsf{LR}})$ described above is a (λ, ϵ)-weakly leakage-resilient VSS protocol tolerating t malicious parties that runs in $O(1)$ rounds.*

Proof. We show that $(\mathsf{Share}_{\mathsf{LR}}, \mathsf{Rec}_{\mathsf{LR}})$ satisfies the validity, reconstruction, and secrecy properties described in Definition 4.2.

Validity. If the dealer is honest, then only malicious parties can complain of bad shares; the dealer will broadcast honest shares to the complaining parties and thus will not be rejected. Further, by the unique decoding property of the underlying modified Shamir secret sharing scheme (Property 4.1.1), any secret sharing of s with up to t corrupted shares will be uniquely decoded to yield the original secret s.

Reconstruction. Consider the case of a dishonest dealer: we show that if the dealer is not rejected, then at the end of the sharing phase all honest parties hold consistent shares of some value s. This will be sufficient to argue that all honest parties will output s at the conclusion of the reconstruction phase, again by Property 4.1.1.

Lemma 4.4. *If the dealer is accepted in the sharing phase of the protocol, then at the conclusion of the sharing phase all honest parties hold shares consistent with a single degree-d polynomial.*

Before we prove the lemma, we introduce some notation and prove a claim that we will invoke later. Let $H \subseteq [n]$ be the set of honest parties. For any vector $v \in \mathbb{F}^n$ and subset $W \subseteq [n]$, we denote by v_W the vector in $\mathbb{F}^{|W|}$ formed by taking the components v_i with $i \in W$. We say that a set of shares v_W is "d-consistent" if the interpolation of the corresponding points yields a polynomial of degree no greater than d. Note that any set of at most $d+1$ shares is trivially d-consistent.

Claim 4.5. *Let $p_1, p_2, p_3 \in \mathbb{F}[x]$ be polynomials with $\deg p_1 > d$. Then for any distributions A, B, C over $\mathbb{F}$ such that $H_\infty(A), H_\infty(B), H_\infty(C) \geq k$, it holds that*

$$\Pr_{\substack{\alpha \leftarrow A, \beta \leftarrow B, \\ \gamma \leftarrow C}} [\deg(\alpha p_1 + \beta p_2 + \gamma p_3) \leq d] \leq \frac{1}{2^{k-1}}.$$

Proof. Consider the term of highest degree $ax^{d'}$ in p_1, and let $bx^{d'}$ and $cx^{d'}$ be the terms of corresponding degree in p_2 and p_3.

$$\begin{aligned}
\Pr[\deg(\alpha p_1 + \beta p_2 + \gamma p_3) \leq d] &\leq \Pr[\alpha a + \beta b + \gamma c = 0] \\
&\leq \Pr[\alpha = 0] + \Pr[\alpha a + \beta b + \gamma c = 0 | \alpha \neq 0] \\
&\leq \frac{1}{2^k} + \Pr[a = -\alpha^{-1}(\beta b + \gamma c) | \alpha \neq 0] \\
&\leq \frac{1}{2^k} + \frac{1}{2^k} = \frac{1}{2^{k-1}}.
\end{aligned}$$

□

We will use this claim to argue that if any collection of shares s_W (or r_W or r'_W) of size $|W| > d+1$ is not d-consistent (ie, they interpolate to a polynomial of degree strictly greater than d), then with high probability over the choice of α, β, γ, the linear combination of these shares $\alpha s_W + \beta r_W + \gamma r'_W$ will not be d-consistent.

Proof of Lemma 4.4. Consider the vector $v = (v_1, ..., v_n)$ received in Round 4 of the protocol, where allegedly $v_i = \alpha s_i + \beta r_i + \gamma r'_i \ \forall i$. Recall if v is distance greater than t from any codeword, then the dealer is rejected at this stage, and the lemma holds. Otherwise, there exists a unique codeword closest to v, corresponding to some polynomial p of degree d, and we define $D \subseteq H$ be the set of honest parties

i for which $v_i \neq p(i)$.[7] For each such party $i \in D$, update their shares s_i, r_i, r'_i with the corresponding values broadcast by the dealer.

Consider the collection of shares $v_{H\setminus D}$. By definition of D, this collection of remaining shares is d-consistent. Since these are honest parties, we know that $v_i = \alpha s_i + \beta r_i + \gamma r'_i$ for each $i \in H \setminus D$. Claim 4.5 tells us that for any set $W \subset [n]$ chosen before α, β, γ,

$$\Pr_{\alpha,\beta,\gamma}[(v_W \text{ } d\text{-consistent}) \wedge (s_W \text{ not } d\text{-consistent})] \leq 2^{-(k-1)},$$

where $k = H_\infty(\alpha) = H_\infty(\beta) = H_\infty(\gamma)$. Note since $|\alpha_i| = |\beta_i| = |\gamma_i| = 2$ and the number of honest parties is $n - t$, then $k \geq 2(n-t)$. In our case, the set $H \setminus D$ is not defined a priori, but rather is determined as a function of the random variables α, β, γ themselves, so we cannot apply this claim for $W = H \setminus D$ outright. But,

$$\begin{aligned}
&\Pr_{\alpha,\beta,\gamma}[(v_{H\setminus D} \text{ } d\text{-consistent}) \wedge (s_{H\setminus D} \text{ not } d\text{-consistent})] \\
&\qquad \leq \Pr_{\alpha,\beta,\gamma}[\exists W \subseteq [n] \text{ s.t. } (v_W \text{ } d\text{-consistent}) \wedge (s_W \text{ not } d\text{-consistent})] \\
&\qquad \leq (2^n)(2^{-(k-1)}) \\
&\qquad = (2^n)2^{-2(n-t)+1} \\
&\qquad = 2^{\Omega(n)},
\end{aligned}$$

where the second inequality follows from the union bound and Claim 4.1, and the last equality follows from the fact that $n = (3+\delta)t$. The same probability bound holds for $r_{H\setminus D}$ and $r'_{H\setminus D}$ in the place of $s_{H\setminus D}$. Thus, by a simple union bound, the probability that any one of $s_{H\setminus D}, r_{H\setminus D}$, or $r'_{H\setminus D}$ is not d-consistent given that v is d-consistent is negligible, and thus we will assume it is not the case.

In particular, this implies the shares $\tilde{v}_{H\setminus D}$, defined by $\tilde{v}_i = \tilde{\alpha} s_i + \tilde{\beta} r_i + \tilde{\gamma} r'_i \ \forall i \in H \setminus D$, are d-consistent. Consider the rest of this vector $\tilde{v} = (\tilde{v}_1, ..., \tilde{v}_n)$ received in Round 5 of the protocol. We wish to show that either the updated shares of s held by honest parties (s_H) are all d-consistent, or the vector $\tilde{v}$ will lead us to reject the dealer. We consider two cases:

Case 1: $\tilde{v}_H$ is d-consistent. In this case, by Claim 4.5, with probability at least $1 - 2^{-(k-1)} = 1 - negl(n)$ we have that s_H is d-consistent, and we are done. (Note that here Claim 4.5 *can* be applied directly, since the set H does not depend on $\tilde{\alpha}, \tilde{\beta}, \tilde{\gamma}$).

Case 2: $\tilde{v}_H$ is not d-consistent. We show that in this case the dealer is rejected. If $\tilde{v}$ is distance greater than t from any codeword, then the dealer is rejected immediately, and we are done. Otherwise, there is a unique closest codeword to $\tilde{v}$, corresponding to some polynomial $\tilde{p}$ of degree d. Let $\tilde{D} \subseteq H$ be the set of honest parties i for which $\tilde{v}_i \neq \tilde{p}(i)$. If $D \cap \tilde{D} \neq \emptyset$, then the dealer will be rejected, and again we are done. So assume $\tilde{v}_i = \tilde{p}(i)$ for all $i \in D$. We know that $\tilde{V}_{H\setminus D}$ is d-consistent, corresponding to the evaluations of some degree d polynomial p'. But, since we are in the case that $\tilde{v}_H$ is not d-consistent, it must be that $p' \neq \tilde{p}$. Since p' and $\tilde{p}$ are polynomials of degree d, this means $p'(i)$ can equal $\tilde{p}(i)$ for at most d values of i. Thus, $|\tilde{D}| \geq |H \setminus D| - d$. But, this means $|D \cup \tilde{D}| \geq |D| + (|H \setminus D| - d) = |H| - d = (2+\delta)t - ((1+\delta)t - 1) = t + 1$, and therefore the dealer will be rejected.

□

Secrecy. We now consider the case of an honest dealer, and show that even an adversary who corrupts t parties and leaks cannot learn too much about the secret value s.

Lemma 4.6. *Let $s \leftarrow \mathbb{F}^{\delta t}$. Let $\mathcal{A}$ be a computationally unbounded adversary for the VSS protocol who adaptively leaks a total of ℓ bits from shares of honest parties during the execution of the protocol.*

[7] Note the minor inconsistency in notation, where before we defined D to be the set of *all* parties (not just honest parties) whose component is in disagreement.

Let $\mathsf{view}_{\mathcal{A}}(s)$ *denote the view of* $\mathcal{A}$ *at the conclusion of the sharing phase. Then with overwhelming probability in* n*,*

$$H_\infty(S|\mathsf{view}_{\mathcal{A}}(s)) \geq H_\infty(S) - \ell - \log^2 n,$$

where the probability is taken over $s \leftarrow S$ *and the random coins of all parties.*

Proof. At the conclusion of the sharing phase, the adversary has received three pieces of information: (1) the secret shares of corrupted parties $\{s_i, r_i, r'_i\}_{i\in\mathcal{C}}$, (2) the answers to his leakage queries L, and (3) the linear combinations $(\alpha s + \beta r + \gamma r'), (\tilde{\alpha} s + \tilde{\beta} r + \tilde{\gamma} r')$.

$$\begin{aligned}
H_\infty(S|\mathsf{view}_{\mathcal{A}}(s)) &= H_\infty\big(S|(\{s_i, r_i, r'_i\}_{i\in\mathcal{C}}, L, (\alpha S + \beta R + \gamma R'), (\tilde{\alpha} S + \tilde{\beta} R + \tilde{\gamma} R'))\big) \\
&\geq H_\infty\big(S|(\{s_i, r_i, r'_i\}_{i\in\mathcal{C}}, (\alpha S + \beta R + \gamma R'), (\tilde{\alpha} S + \tilde{\beta} R + \tilde{\gamma} R'))\big) - \ell - \log^2 n \\
&\text{with overwhelming probability, since } |L| \leq \ell \text{ (see Lemma 2.5).} \\
&= H_\infty\big(S|((\alpha S + \beta R + \gamma R'), (\tilde{\alpha} S + \tilde{\beta} R + \tilde{\gamma} R'))\big) - \ell - \log^2 n \\
&\text{since any } t \text{ shares are independent of } S, R, R'. \\
&= H_\infty(S) - \ell - \log^2 n \\
&\text{with overwhelming probability, since } \alpha S + \beta R + \gamma R' \text{ and } \tilde{\alpha} S + \tilde{\beta} R + \tilde{\gamma} R' \text{ are} \\
&\text{independent of } S \text{ for any nonzero fixed choice of coefficients } \alpha, \beta, \gamma, \tilde{\alpha}, \tilde{\beta}, \tilde{\gamma}.
\end{aligned}$$

□

What remains is to prove that $H_\infty(S) - \ell - \log^2 n \geq \epsilon H_\infty(S)$, where ℓ is the total amount of leakage. Recall that the adversary can leak $\lambda = \frac{\delta(1-\epsilon)}{10+6\delta}$ fraction of each honest party's secret state. The secret state of each non-dealer party consists of precisely three elements of $\mathbb{F}$, corresponding to his shares s_i, r_i, and r'_i. Note that $\alpha_i, \beta_i, \gamma_i, \tilde{\alpha}_i, \tilde{\beta}_i, \tilde{\gamma}_i$ are broadcast immediately after being generated, and thus are not part of the secret state. The dealer must hold additional secret information, since he must be able to produce a valid secret share for any complaining party in Round 4. Thus, he must store $3(d+1) = 3(1+\delta)t$ secret elements of $\mathbb{F}$, corresponding to the coefficients of the secret sharing polynomials for s, r, and r'. Thus,

$$\begin{aligned}
\ell = \lambda \sum_{i\notin\mathcal{C}} |\mathsf{state}_i| &= \lambda\Big((n-t)(3\log|\mathbb{F}|) + 3(d+1)\log|\mathbb{F}|\Big) \\
&= \lambda\Big((2+\delta)t(3\log|\mathbb{F}|) + 3(1+\delta)t\log|\mathbb{F}|\Big) \\
&= 3t\log|\mathbb{F}|\lambda\big(2+\delta+1+\delta\big) \\
&= 3t\log|\mathbb{F}|\lambda(3+2\delta)
\end{aligned}$$

Combining this with Lemma 4.6, we have that with overwhelming probability:

$$\begin{aligned}
H_\infty(S|\mathsf{view}_{\mathcal{A}}(s)) &\geq H_\infty(S) - \ell - \log^2 n \\
&= \delta t\log|\mathbb{F}| - 3t\log|\mathbb{F}|\lambda(3+2\delta) - \log^2 n \\
&= t\log|\mathbb{F}|\left(\delta - \lambda 3(3+2\delta) - \frac{\log^2 n}{t\log|\mathbb{F}|}\right) \\
&\geq t\log|\mathbb{F}|\left(\delta - \lambda\Big((9+6\delta) + \frac{1}{n}\Big)\right) \\
&\geq t\log|\mathbb{F}|\Big(\delta - \lambda(10+6\delta)\Big) \\
&= t\log|\mathbb{F}|\Big(\delta - \frac{\delta(1-\epsilon)}{10+6\delta}(10+6\delta)\Big) \\
&= \epsilon\delta t\log|\mathbb{F}| = \epsilon H_\infty(S)
\end{aligned}$$

Thus, the protocol $(\mathsf{Share}_{\mathsf{LR}}, \mathsf{Rec}_{\mathsf{LR}})$ satisfies the properties of a (λ, ϵ)-weakly leakage-resilient VSS protocol.

□

4.3 *Strongly* Leakage-Resilient (Oblivious) VSS

For our coin tossing protocol, we only need VSS achieving a weak notion of leakage resilience, where for any adversary who corrupts t parties and leaks a constant fraction of the secret state of the remaining parties, the secret still retains a constant fraction of its original entropy. However, one can also consider a stronger version of leakage resilience, where the secret retains its *full* entropy.

Naturally, this notion cannot be achieved if any party knows the secret in its entirety, since the adversary can simply leak on this value outright. In particular, this immediately rules out the possibility of standard VSS, since the dealer himself cannot know the secret! We thus put forth the notion of *oblivious* secret sharing, where the dealer secret shares a random value *without knowing* its value. We also show that this is, in fact, achievable (see below). We believe that leakage-resilient oblivious VSS primitives can serve as a useful building block for constructing future leakage-resilient protocols, which anyway make use of VSS in this fashion (e.g., in [FM85] to achieve Byzantine Agreement).

Definition 4.7 (Strongly Leakage-Resilient (Oblivious) VSS)**.** *A λ-strongly leakage-resilient (oblivious) VSS protocol tolerating t malicious parties for parties $\mathcal{P} = \{P_1, ..., P_n\}$ is a protocol with two phases (sharing and reconstruction), such that with overwhelming probability in n, the following conditions hold for any adversary controlling at most t parties and leaking up to λ fraction of the secret state of each (honest) party:*

- **Reconstruction:** *Even if the dealer is dishonest, at the end of the sharing phase, the joint view of the honest parties defines a value s' (which can be computed efficiently from this view) such that at the end of the reconstruction phase, all honest parties will output s'.*
- **Secrecy:** *If the dealer is honest during the sharing phase, then it holds with overwhelming probability over $y \leftarrow \mathsf{view}_{\mathcal{A}}$ that at the end of this phase the view of the adversary (consisting of shares of corrupted parties and leakage on remaining shares) is independent of this value s'.*

Even without leakage considerations, it is not immediately clear whether one can hope to achieve oblivious secret sharing robust to malicious parties. Consider, as an example, the Shamir secret sharing scheme. The dealer can sample random values for individual shares; but in order to ultimately make the shares consistent, he must somehow sample from a polynomial—without knowing the polynomial!

We show that this *can* be done. In Appendix A, we present a λ-strongly leakage-resilient (oblivious) VSS protocol for $\lambda = \Omega(1)$, tolerating $t \leq \frac{n}{3+\delta}$ malicious parties (for any constant $\delta > 0$). Our construction uses the tool of a *weakly* leakage-resilient VSS protocol as a black box (see Definition 4.2). At a high level, the protocol works by having the dealer sample and share two random values x and y using the weakly leakage-resilient protocol; the final output will be $\mathsf{Ext}_2(x, y)$, where Ext_2 is a two-source extractor. To ensure that information is never leaked on x and y together, the dealer first samples and verifiably secret shares x, erases it, then samples and verifiably secret shares y. (Note that we do not need to assume complete erasures, but rather can allow some fraction of information to remain, which is simply treated as leakage). As before, to ensure independence, instead of sharing x and y to all parties, he will share x and y among two disjoint committees, which are selected by all parties using a version of the Feige committee election protocol.

Theorem 4.8. *Let $n = (3 + \delta)t$ for any constant $\delta > 0$. Then for any constant $\lambda \leq \frac{\delta}{4(5+3\delta)}$, there exists a strongly leakage-resilient (oblivious) VSS protocol tolerating t malicious parties that runs in $O(1)$ rounds.*

Proof. See Appendix A. □

5 Disjoint Committee Election

We now exhibit a 1-round public-coin protocol for electing $m = \log^2 n$ *disjoint* "good" committees $\mathcal{E}_1, ..., \mathcal{E}_m$ of size approximately $n^{1/2}$.

Let $m = \log^2 n$ and $k = n^{1/2}$. We consider m parallel repetitions of the Feige lightest bin protocol with $\frac{n}{k}$ bins (See Section 2.4). Then, to ensure disjointness of the committees, we remove parties who are elected to multiple committees from all but the first in which they appear.

More explicitly, we define the protocol $\mathsf{ElectDisj}$ as follows. In a single round, each party $i \in [n]$ broadcasts m random values $r_1^i, ..., r_m^i \leftarrow \left[\frac{n}{k}\right]$. Locally, everyone iterates through $j = 1, ..., m$, setting $\mathcal{E}_j$ to be the set of parties in the lightest bin in the jth election, defined by $r_j^1, ..., r_j^n$; then, to ensure disjointness, all parties who have been previously elected to any committee are removed from $\mathcal{E}_j$, and this becomes the final jth elected committee.

Proposition 5.1. *The protocol* $\mathsf{ElectDisj}$ *1-round public-coin protocol for electing* $m = \log^2 n$ *committees* $\mathcal{E}_i$ *such that for any constants* $\beta, \epsilon > 0$*, and any set of corrupted parties* $\mathcal{C} \subset [n]$ *of size* βn*, the following events simultaneously occur with overwhelming probability in* n*:*

1. $\forall i \neq j,\ \mathcal{E}_i \cap \mathcal{E}_j = \emptyset$,
2. $\forall i,\ (1 - \beta - \epsilon)n^{1/2} \leq |\mathcal{E}_i| \leq n^{1/2}$,
3. $\forall i,\ \frac{|\mathcal{E}_i \cap \mathcal{C}|}{|\mathcal{E}_i|} < \beta + \epsilon$.

Proof. By construction, property (1) holds immediately. Further, by Corollary 2.7, each $\mathcal{E}_j$ is of size at most k and has at least $(1 - \beta - \frac{\epsilon}{2})k$ honest parties before erasures. It thus remains to show that by removing parties who appear in multiple committees, we do not decrease the number of honest parties in any committee by too much.

Consider an execution of the protocol. In particular, consider the placement of all honest parties in bins for all of the parallel elections, ignoring any actions of malicious parties. We will prove that with overwhelming probability, for any choice of $j \in [m]$ and any choice of bins $B_1, ..., B_m$ (one from each election), the number of honest parties that would be erased from B_j, given that $B_1, ..., B_j$ are elected, is at most $\frac{\epsilon}{2}k$. That is,

$$\Pr\left[\exists\ j \in [m],\ \{B_{j'}\}_{j'<j} \text{ s.t. } \left|B_j \cap \Big(\bigcup_{j'<j} B_{j'}\Big)\right| > \frac{\epsilon}{2}k\right] = negl(n). \tag{1}$$

If equation (1) holds, then with overwhelming probability, even after erasing recurring parties, each elected committee must contain at least $(1 - \beta - \frac{\epsilon}{2})k - \frac{\epsilon}{2}k = (1 - \beta - \epsilon)k$ honest parties, implying property (2). Further, since each $|\mathcal{E}_i| \leq k$, then with overwhelming probability we will have

$$\frac{|\mathcal{E}_i \cap \mathcal{C}|}{|\mathcal{E}_i|} = 1 - \frac{|\mathcal{E}_i \setminus \mathcal{C}|}{|\mathcal{E}_i|} \leq 1 - \frac{(1 - \beta - \epsilon)k}{k} = \beta + \epsilon,$$

implying property (3).

We now prove that equation (1) holds. Consider any fixed choice of $j \in [m]$ and bins $B_1, ..., B_j$. Denote by S_j the set $\bigcup_{j'<j} B_{j'}$. We can assume that

$$\left(1 - \beta - \frac{\epsilon}{2}\right)(j-1)k \ \leq\ |S_j| \ \leq\ (j-1)k; \tag{2}$$

the lower bound is guaranteed with overwhelming probability by Corollary 2.7, and the upper bound can be assumed since any bin with more than k parties will never be elected as a lightest bin, and thus can be ignored.

For each party $i \in S_j$, let X_i be the indicator variable that is equal to 1 iff party P_i selected bin B_j in the jth election. Namely, $\sum_{i \in S_j} X_i = |B_j \cap (\bigcup_{j'<j} B_{j'})|$. Since we are considering only honest

parties, P_i selects a bin uniformly at random, and so the X_i are Bernoulli variables with probability $\frac{k}{n}$. By equation (2) above, the expected value of the sum $\mu = \mathbb{E}[\sum_{i \in S_j} X_i]$ satisfies

$$\mu \;\leq\; (j-1)k\frac{k}{n} \;\leq\; (\log^2 n)\frac{(n^{1/2})^2}{n} = \log^2 n, \text{ and}$$

$$\mu \;\geq\; \left(1-\beta-\frac{\epsilon}{2}\right)(j-1)k\frac{k}{n} \;\geq\; \left(1-\beta-\frac{\epsilon}{2}\right).$$

For any constant $\zeta > 0$, the Chernoff bound[8] gives us

$$\Pr\Big[\sum_{i \in S_j} X_i > n^{\zeta}\mu\Big] < 2^{-n^{\zeta}\mu} < 2^{-cn^{\zeta}},$$

where $c = (1-\beta-\frac{\epsilon}{2})$. Taking a union bound over all possible values of $j \in \{1, ..., m\}$ and all $(\frac{n}{k})^j \leq (\frac{n}{k})^m$ choices of bins $B_1, ..., B_j$, we have

$$\Pr\Big[\exists\ j \in [m],\ \{B_{j'}\}_{j' \leq j} \text{ s.t. } \sum_{i \in S_j} X_i > n^{\zeta}\log^2 n\Big] \leq m\left(\frac{n}{k}\right)^m 2^{-cn^{\zeta}} \leq 2^{-cn^{\zeta/2}},$$

which is negligible in n. Since $\sum_{i \in S_j} X_i = |B_j \cap (\bigcup_{j'<j} B_{j'})|$, and $n^{\zeta}\log^2 n < \frac{\epsilon}{2}k = \frac{\epsilon}{2}n^{1/2}$ for any $\zeta < \frac{1}{2}$, this implies equation (1). Hence, by the discussion above, properties (2) and (3) hold.

□

6 Unbiased Coin Tossing with Leakage

In this section, we construct our final leakage-resilient coin tossing protocol, as characterized by Definition 3.1. Our construction makes black-box use of the tools developed in the previous sections: in particular, a weakly leakage-resilient verifiable secret sharing (VSS) protocol (from Section 4.2), and a disjoint committee election protocol (from Section 5).

Recall we are within the model of a synchronous point-to-point network with broadcast, and that channels are assumed to be authenticated and private (with leakage). Our results are information theoretic, without cryptographic assumptions.

Theorem 6.1. *For any constants $\delta, \epsilon > 0$, any $\lambda \leq \frac{\delta(1-\epsilon)}{10+6\delta}$, any $n \geq (3+\delta)t$, and any m, there exists a λ-leakage-resilient n-party distributed coin tossing protocol tolerating t malicious parties that generates m unbiased random bits, and terminates in $O(1)$ rounds.*

Proof. Let δ' be any constant such that $\delta' < \delta$. We construct the desired coin tossing protocol using the following tools, with the corresponding listed parameters:

1. Elect: Feige's 1-round public-coin protocol to elect a primary committee of size approximately $\log^2 n$, as in Corollary 2.7.

2. $\mathsf{ElectDisj}$: a 1-round public-coin protocol for electing $\log^2 n$ *disjoint* secondary committees of size $n' \approx n^{1/2}$,[9] as in Proposition 5.1.

3. $(\mathsf{Share}_{\mathsf{LR}}, \mathsf{Rec}_{\mathsf{LR}})$: a (λ, ϵ)-weakly leakage-resilient VSS protocol for n' parties, tolerating $t' \leq \frac{n'}{3+\delta'}$ malicious parties, terminating in $O(1)$ rounds, as in Theorem 4.3.

4. $\mathsf{Ext} : (\{0,1\}^d)^{\log^2 n} \to \{0,1\}^m$: a robust multi-source extractor, where $m = .99(\frac{2}{3}\log^2 n)(\epsilon d)$, as in Theorem 2.4. We interpret each element $\{0,1\}^d$ as an element of $\mathbb{F}^{\delta' t'}$ $(d = \delta' t' \log|\mathbb{F}|)$ where the size of $\mathbb{F}$ depends on the desired output length m.

[8] Exact Chernoff bound: For $X_1, ..., X_n$ independent Bernoulli random variables and $\mu = \mathbb{E}[\sum_i X_i]$, then for $a > 6$, it holds that $\Pr[\sum_i X_i > a\mu] < 2^{-a\mu}$.

[9] Note that we will use prime notation (e.g., n', t', δ') to denote parameters pertaining to the secondary committees.

We now construct the desired coin tossing protocol.

CoinToss:

1. **Step 1:** Run Elect to elect a primary committee of approximate size $\log^2 n$ (see Corollary 2.7). Denote the set of indices of elected parties by $\mathcal{E} \subset [n]$.

2. **Step 2:** Run the ElectDisj protocol on the remaining parties $[n] \setminus \mathcal{E}$ to elect $|\mathcal{E}|$ disjoint secondary committees $\mathcal{E}'_1, ..., \mathcal{E}'_{|\mathcal{E}|}$, each of size approximately $n^{1/2}$ (see Proposition 5.1).

3. **Step 3:** $\forall i \in \mathcal{E}$, P_i samples a random value $r_i \leftarrow \mathbb{F}^{\delta' t'}$ and verifiably secret shares it among the parties in his corresponding secondary committee, $\mathcal{E}'_i$. That is, he acts as a dealer in an execution of $\mathsf{Share}_{\mathsf{LR}}(r_i)$.

4. **Step 4:** For each $i \in \mathcal{E}$, all parties in the secondary committee $\mathcal{E}'_i$ execute the reconstruction phase $r_i \leftarrow \mathsf{Rec}_{\mathsf{LR}}()$ on the shares dealt by P_i. For any party $i \in \mathcal{E}$ who was rejected as a dealer in the previous step, set $r_i = 0$. Each secondary committee member broadcasts his reconstructed value for r_i.

5. **Local Computation:** Let r_i^* be the most common value received from the parties in secondary committee $\mathcal{E}'_i$ in the previous step. Output $r \leftarrow \mathsf{Ext}(\{r_i^*\}_{i \in \mathcal{E}})$.

By Proposition 5.1, with overwhelming probability in n, the disjoint secondary committees $\mathcal{E}'_i$ will be "good," in that they each have size $n^{1/2-\zeta} \leq |\mathcal{E}'_i| \leq n^{1/2}$ for any constant $\zeta > 0$ and it holds that $n'_i \geq (3+\delta')t'_i$, where $n'_i = |\mathcal{E}'_i|$ and $t'_i = |\mathcal{E}'_i \cap \mathcal{C}|$. We will thus assume this is the case. Since $n'_i \geq (3+\delta')t'_i$, the validity, reconstruction, and secrecy properties of the (λ, ϵ)-weakly leakage-resilient VSS protocol (see Definition 4.2) will hold for the ith execution of $(\mathsf{Share}_{\mathsf{LR}}, \mathsf{Rec}_{\mathsf{LR}})$ with overwhelming probability in n'_i (and thus in n).

We now show that the protocol CoinToss satisfies the desired agreement and randomness properties (see Definition 3.1).

Agreement By the reconstruction property of the leakage-resilient VSS protocol, for each $P_i \in \mathcal{E}$, the honest parties in $\mathcal{E}'_i$ will agree on the reconstructed value $r_i \leftarrow \mathsf{Rec}_{\mathsf{LR}}()$ and will broadcast this value to all parties in Step 4 (where $r_i = 0$ if P_i was rejected as a dealer in the sharing phase of the VSS). Since a majority of the parties in $\mathcal{E}'_i$ are honest, all honest parties in $[n]$ will agree on $r_i^* = r_i$ for each i, and so will agree on the final output r.

Randomness Consider the values r_i reconstructed by each secondary committee $\mathcal{E}'_i$. By the reconstruction property of the leakage-resilient VSS, each r_i is fully determined by the conclusion of the sharing phase (Step 3 of the CoinToss protocol). The secrecy property of the leakage-resilient VSS implies that at the end of the sharing phase, even given the view of the adversary ($\mathsf{view}_{\mathcal{A}}$), each *honest* party's r_i retains at least $\epsilon \cdot (\delta' t' \log |\mathbb{F}|)$ bits of entropy. Therefore, conditioned on $\mathsf{view}_{\mathcal{A}}$ (which includes leakage), the random variables $r_1^*, ..., r_{|\mathcal{E}|}^* \in \mathbb{F}^{\delta' t'}$ are independent, where for all $j \in \mathcal{E} \cap \mathcal{C}$ we think of r_j^* as fixed. Further, together they have total min-entropy at least $(|\mathcal{E} \setminus \mathcal{C}|)(\epsilon \delta' t' \log |\mathbb{F}|)$. By Corollary 2.7, $|\mathcal{E} \setminus \mathcal{C}| \geq (1 - \frac{1}{3+\delta} - \zeta) \log^2 n$ for any constant $\zeta > 0$ with overwhelming probability in n. Since the extractor we use can extract even when many of the sources r_j^* are fixed, we can simply take the loose bound $|\mathcal{E} \setminus \mathcal{C}| \geq \frac{2}{3} \log^2 n$. By Theorem 2.4, the final output $r = \mathsf{Ext}(\{r_i^*\}_{i \in \mathcal{C}})$ will be statistically close to uniform over $\{0,1\}^m$ with $m = .99(\frac{2}{3} \log^2 n)(\epsilon \delta' t' \log |\mathbb{F}|)$.

□

References

[AGH10] Adi Akavia, Shafi Goldwasser, and Carmit Harzay. Distributed public key schemes secure against continual leakage. Manuscript, 2010.

[AGV09] Adi Akavia, Shafi Goldwasser, and Vinod Vaikuntanathan. Simultaneous hardcore bits and cryptography against memory attacks. In *Proceedings of the 6th Theory of Cryptography Conference*, pages 474–495, 2009.

[BCH11] Nir Bitansky, Ran Canetti, and Shai Halevi. Leakage tolerant interactive protocols. Cryptology ePrint Archive, Report 2011/204, 2011.

[Ben83] Michael Ben-Or. Another advantage of free choice: Completely asynchronous agreement protocols (extended abstract). In *PODC*, pages 27–30, 1983.

[BGW88] 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, 1988.

[BKKV10] Zvika Brakerski, Yael Tauman Kalai, Jonathan Katz, and Vinod Vaikuntanathan. Overcoming the hole in the bucket: Public-key cryptography resilient to continual memory leakage. In *FOCS*, pages 501–510, 2010.

[Bou05] Jean Bourgain. More on the sum-product phenomenon in prime fields and its applications. *International Journal of Number Theory*, 1(1):1–32, 2005.

[Bra84] Gabriel Bracha. An asynchronous $[(n-1)/3]$-resilient consensus protocol. In *PODC*, pages 154–162, 1984.

[CCD88] David Chaum, Claude Crépeau, and Ivan Damgård. Multiparty unconditionally secure protocols (extended abstract). In *STOC*, pages 11–19, 1988.

[CDH+00] Ran Canetti, Yevgeniy Dodis, Shai Halevi, Eyal Kushilevitz, and Amit Sahai. Exposure-resilient functions and all-or-nothing transforms. In *Advances in Cryptology – EUROCRYPT '00*, pages 453–469, 2000.

[CFGN96] Ran Canetti, Uriel Feige, Oded Goldreich, and Moni Naor. Adaptively secure multi-party computation. In *STOC*, pages 639–648, 1996.

[CGMA85] Benny Chor, Shafi Goldwasser, Silvio Micali, and Baruch Awerbuch. Verifiable secret sharing and achieving simultaneity in the presence of faults. In *Proceedings of the 26th Annual Symposium on Foundations of Computer Science*, pages 383–395, 1985.

[Cle86] Richard Cleve. Limits on the security of coin flips when half the processors are faulty (extended abstract). In *STOC*, pages 364–369, 1986.

[DHLW10] Yevgeniy Dodis, Kristiyan Haralambiev, Adriana López-Alt, and Daniel Wichs. Cryptography against continuous memory attacks. In *FOCS*, pages 511–520, 2010.

[DORS08] Yevgeniy Dodis, Rafail Ostrovsky, Leonid Reyzin, and Adam Smith. Fuzzy extractors: How to generate strong keys from biometrics and other noisy data. *SIAM Journal on Computing*, 38(1):97–139, 2008.

[DP08] Stefan Dziembowski and Krzysztof Pietrzak. Leakage-resilient cryptography. In *Proceedings of the 49th Annual IEEE Symposium on Foundations of Computer Science*, pages 293–302, 2008.

[DSS90] Cynthia Dwork, David B. Shmoys, and Larry J. Stockmeyer. Flipping persuasively in constant time. *SIAM J. Comput.*, 19(3):472–499, 1990.

[Fei99] Uriel Feige. Noncryptographic selection protocols. In *Proceedings of the 40th Annual Symposium on Foundations of Computer Science*, 1999.

[FM85] Paul Feldman and Silvio Micali. Byzantine agreement in constant expected time (and trusting no one). In *FOCS*, pages 267–276, 1985.

[GM82] Shafi Goldwasser and Silvio Micali. Probabilistic encryption and how to play mental poker keeping secret all partial information. In *STOC*, pages 365–377, 1982.

[GMW87] Oded Goldreich, Silvio Micali, and Avi Wigderson. How to play any mental game or a completeness theorem for protocols with honest majority. In *STOC*, pages 218–229, 1987.

[GSV05] Shafi Goldwasser, Madhu Sudan, and Vinod Vaikuntanathan. Distributed computing with imperfect randomness. In *DISC*, pages 288–302, 2005.

[ISW03] Yuval Ishai, Amit Sahai, and David Wagner. Private circuits: Securing hardware against probing attacks. In *Advances in Cryptology – CRYPTO '03*, pages 463–481, 2003.

[KLR09] Yael Tauman Kalai, Xin Li, and Anup Rao. 2-source extractors under computational assumptions and cryptography with defective randomness. In *FOCS*, pages 617–626, 2009.

[KLRZ08] Yael Tauman Kalai, Xin Li, Anup Rao, and David Zuckerman. Network extractor protocols. In *FOCS*, pages 654–663, 2008.

[KRVZ06] Jesse Kamp, Anup Rao, Salil Vadhan, and David Zuckerman. Deterministic extractors for small-space sources. In *Proceedings of the thirty-eighth annual ACM symposium on Theory of computing*, pages 691–700, 2006.

[KV09] Jonathan Katz and Vinod Vaikuntanathan. Signature schemes with bounded leakage resilience. In *Advances in Cryptology – ASIACRYPT '09*, pages 703–720, 2009.

[MNS09] Tal Moran, Moni Naor, and Gil Segev. An optimally fair coin toss. In *TCC*, pages 1–18, 2009.

[MR04] Silvio Micali and Leonid Reyzin. Physically observable cryptography. In *Proceedings of the 1st Theory of Cryptography Conference*, pages 278–296, 2004.

[MS81] R. J. McEliece and D. V. Sarwate. On sharing secrets and reed-solomon codes. *Commun. ACM*, 24:583–584, September 1981.

[NS09] Moni Naor and Gil Segev. Public-key cryptosystems resilient to key leakage. In *Advances in Cryptology – CRYPTO '09*, pages 18–35, 2009.

[Rab83] Michael O. Rabin. Randomized byzantine generals. In *FOCS*, pages 403–409, 1983.

[Raz05] Ran Raz. Extractors with weak random seeds. In *STOC*, pages 11–20, 2005.

[Riv97] Ronald L. Rivest. All-or-nothing encryption and the package transform. In *FSE*, pages 210–218, 1997.

[Sha79] Adi Shamir. How to share a secret. *Commun. ACM*, 22:612–613, November 1979.

A Proof of Theorem 4.8

In this section, we provide a construction and proof of a strongly leakage-resilient (oblivious) VSS protocol.

Proof of Theorem 4.8. Let δ' be any constant such that $\delta' < \delta$. Fix any constant $0 < \epsilon < 1$. We construct the desired protocol $(\mathsf{Share}_{\mathsf{SLR}}, \mathsf{Rec}_{\mathsf{SLR}})$, making use of the following tools:

1. Elect: Feige's 1-round public-coin protocol to elect a primary committee of size approximately $n' = n^{\epsilon}$, as in Lemma 2.6.
2. $(\mathsf{Share}_{\mathsf{WLR}}, \mathsf{Rec}_{\mathsf{WLR}})$: a $(\lambda, \frac{1}{2})$-*weakly* leakage-resilient VSS protocol for n' parties tolerating $t' = \frac{n'}{3+\delta'}$ corrupted parties, terminating in $O(1)$ rounds, as in Theorem 4.3. (Recall $\frac{1}{2}$ refers to the fraction of entropy guaranteed to remain in the secret.)
3. $\mathsf{Ext}_2 : \{0,1\}^k \times \{0,1\}^k \to \{0,1\}^m$: a two-source extractor, where $k = \delta' t' \log |\mathbb{F}|$ and $m = \Omega(k)$, as in Theorem 2.3.

- $\mathsf{Share}_{\mathsf{SLR}}()$:
  1. **Step 1:** Run Elect to elect a committee $\mathcal{E}_1$ of approximate size $n' = n^{\epsilon}$ from the set of non-dealer parties $[n] \setminus \{P^*\}$ (see Lemma 2.6).
  2. **Step 2:** The dealer P^* samples a random value $x \leftarrow \mathbb{F}^{\delta' t'}$ and verifiably secret shares it among the parties in the committee $\mathcal{E}_1$. That is, he acts as a dealer in an execution of $\mathsf{Share}_{\mathsf{WLR}}(x)$. He then erases x (and all values related to x).
  3. **Step 3:** Run Elect to elect a second committee $\mathcal{E}_2$ of approximate size n' from $[n] \setminus (\{P^*\} \cup \mathcal{E}_1)$ (see Lemma 2.6).
  4. **Step 4:** The dealer P^* samples a random value $y \leftarrow \mathbb{F}^{\delta' t'}$ and verifiably secret shares it among the parties in the committee $\mathcal{E}_2$. That is, he acts as a dealer in an execution of $\mathsf{Share}_{\mathsf{WLR}}(y)$. He then erases y (and all values related to y).
  5. **Step 5:** Each party in $\mathcal{E}_1$ and $\mathcal{E}_2$ broadcasts Accept or Reject, corresponding to whether the dealer was accepted or rejected in $\mathsf{Share}_{\mathsf{WLR}}(x)$ during Step 2 or 4, respectively.
  6. **Local Computation:** The dealer is accepted if a majority of parties in both $\mathcal{E}_1$ and $\mathcal{E}_2$ broadcast Accept. Otherwise, the dealer is rejected.
- $\mathsf{Rec}_{\mathsf{SLR}}()$:
  1. **Step 1:** All parties in $\mathcal{E}_1$ (respectively, $\mathcal{E}_2$) execute the reconstruction phase $x \leftarrow \mathsf{Rec}_{\mathsf{WLR}}()$ (resp, $y \leftarrow \mathsf{Rec}_{\mathsf{WLR}}()$), on the shares dealt in Step 2 (resp, Step 4). Each committee member broadcasts his reconstructed value of x (resp, y).
  2. **Local computation:** Let x^*, y^* be the most common value received from the parties in $\mathcal{E}_1$ and $\mathcal{E}_2$, respectively, in the previous step. Output $s \leftarrow \mathsf{Ext}_2(x^*, y^*)$.

By Lemma 2.6, with overwhelming probability in n, both committees $\mathcal{E}_1, \mathcal{E}_2$ will be "good," in that they each have size $n^{\epsilon/2} \le |\mathcal{E}_i| \le n^{\epsilon}$ and it holds that $n'_i \ge (3+\delta')t'_i$, where $n'_i = |\mathcal{E}_i|$ and $t'_i = |\mathcal{E}_i \cap \mathcal{C}|$ for $i \in \{1,2\}$. We will thus assume this is the case. Since $n'_i \ge (3+\delta')t'_i$, the validity, reconstruction, and secrecy properties of the $(\lambda, \frac{1}{2})$-weakly leakage-resilient VSS protocol (see Definition 4.2) will hold for the ith execution of $(\mathsf{Share}_{\mathsf{WLR}}, \mathsf{Rec}_{\mathsf{WLR}})$ with overwhelming probability in n'_i (and thus in n).

We now show that $(\mathsf{Share}_{\mathsf{SLR}}, \mathsf{Rec}_{\mathsf{SLR}})$ satisfies the reconstruction and secrecy properties given in Definition 4.7.

Reconstruction By the reconstruction property of the underlying λ-weakly leakage-resilient VSS protocol, the honest parties in $\mathcal{E}_1$ (respectively, in $\mathcal{E}_2$) will agree on the reconstructed value $x' \leftarrow \mathsf{Rec}_{\mathsf{WLR}}()$ (resp, $y' \leftarrow \mathsf{Rec}_{\mathsf{WLR}}$), and will broadcast this value to all parties in Step 1 of the reconstruction phase. Since a majority of the parties in $\mathcal{E}_i$ are honest, all honest parties in $[n]$ will agree on the values of $x^* = x, y^* = y$, and thus will output the same value $\mathsf{Ext}_2(x^*, y^*)$.

Secrecy Assume the dealer is honest. Note that since the dealer erases x (and all values related to x) before generating y, any leakage function will be a function of purely x or y, when conditioned on prior leakage. Thus, conditioned on the leakage, the reconstructed values of x and y will be independent. By the secrecy property of the underlying λ-weakly leakage-resilient VSS protocol, given the view of the adversary, both x and y retain at least $\frac{1}{2}$ of their original entropy. Therefore, by Theorem 2.3, the final output $\mathsf{Ext}_2(x, y)$ will be statistically close to uniform over $\{0,1\}^m$ for $m = \Omega(k)$.

□