Essentially Optimal Robust Secret Sharing with Maximal Corruptions

Abstract. In a t-out-of-n *robust secret sharing scheme*, a secret message is shared among n parties who can reconstruct the message by combining their shares. An adversary can adaptively corrupt up to t of the parties, get their shares, and modify them arbitrarily. The scheme should satisfy *privacy*, meaning that the adversary cannot learn anything about the shared message, and *robustness*, meaning that the adversary cannot cause the reconstruction procedure to output an incorrect message. Such schemes are only possible in the case of an honest majority, and here we focus on unconditional security in the maximal corruption setting where $n = 2t + 1$.

In this scenario, to share an m-bit message with a reconstruction failure probability of at most 2^{-k}, a known lower-bound shows that the share size must be at least $m+k$ bits. On the other hand, all prior constructions have share size that scales linearly with the number of parties n, and the prior state-of-the-art scheme due to Cevallos et al. (EUROCRYPT '12) achieves $m + \widetilde{O}(k + n)$.

In this work, we construct the first robust secret sharing scheme in the maximal corruption setting with $n = 2t + 1$, that avoids the linear dependence between share size and the number of parties n. In particular, we get a share size of only $m + \widetilde{O}(k)$ bits. Our scheme is computationally efficient and relies on approximation algorithms for the minimum graph bisection problem.

1 Introduction

Secret sharing, originally introduced by Shamir [Sha79] and Blakely [Bla79], is a central cryptographic primitive at the heart of a wide variety of applications, including secure multiparty computation, secure storage, secure message transmission, and threshold cryptography. The functionality of secret sharing allows a dealer to split a secret message into shares that are then distributed to n parties. Any authorized subset of parties can reconstruct the secret reliably from their shares, while unauthorized subsets of parties cannot learn anything about the secret from their joint shares. In particular, a t-out-of-n secret sharing scheme requires that any t shares reveal no information about the secret, while any subset of $t + 1$ shares can be used to reconstruct the secret.

Many works (e.g. [RB89,CSV93,BS97,CDF01,CFOR12,LP14,CDD+15,Che15]) consider a stronger notion of secret sharing called *robust secret sharing* (RSS). Robustness requires that, even if an adversary can replace shares of corrupted parties by maliciously chosen values, the parties can still reconstruct the true secret. In particular, we consider a computationally unbounded adversary who maliciously (and adaptively) corrupts t out of n of the parties and learns their shares. After corrupting the t parties, the adversary can adaptively modify their shares and replace them with arbitrary values. The reconstruction algorithm is given the shares of all n parties and we require that it recovers the original secret.

It is known that robust secret sharing can only be achieved with an honest majority, meaning $t < n/2$. Moreover, for t in the range $n/3 \leq t < n/2$, robust secret sharing cannot be achieved with perfect correctness, meaning that we must allow at least a small (negligible) failure probability for reconstruction [Cev11]. Furthermore, in the maximal corruption setting with $n = 2t + 1$ parties, any robust secret sharing scheme for m-bit messages with failure probability 2^{-k} must have a share size that exceeds $m + k$ bits [CSV93,LP14].

On the positive side, several prior works show how to construct robust sharing schemes with $n = 2t + 1$ parties. The first such scheme was described in the work of Rabin and Ben-Or [RB89] with a share size of $m + \widetilde{O}(nk)$ bits. Cramer, Damgård and Fehr [CDF01] showed how to improve this to $m+\widetilde{O}(k+n)$ bits, using what later became known as algebraic-manipulation detection (AMD) codes [CDF$^+$08], but at the cost of having an inefficient reconstruction procedure. Cevallos et al. [CFOR12] then presented an efficient scheme with share size $m + \widetilde{O}(k + n)$.

Two recent works [CDD$^+$15,Che15] study the setting where the number of corruptions is below the maximal threshold by some constant fraction; i.e., $t = (1/2 - \delta)n$ for some constant $\delta > 0$. In this setting, the robustness requirement means that the secret can be reconstructed from n shares of which t are adversarially modified, but it does not necessarily imply threshold reconstructibility from $t + 1$ correct shares (it only implies reconstructibility from $(1/2 + \delta)n$ correct shares). This is often called a *ramp* setting, where there is a gap between the privacy threshold t and the reconstructibility threshold. The above works show that it is possible to achieve robustness in this setting with a share size of roughly $O((m + k)/n)$ bits. In particular, the share size can be smaller than the message size and, when n is large, the share size can even be constant.[1] Unfortunately, the techniques in these works do not translate to the setting of $n = 2t + 1$ parties.

In summary, despite intense study since the late 80s and early 90s, there is a large gap between the lower bound of $m + k$ bits and the best previously known upper bound of $m + O(n + k)$ bits on the share size in the maximal corruption setting where $n = 2t + 1$. In particular, prior to this work, it was not known if the linear dependence between the share size and n is necessary in this setting, or whether there exist (even inefficient) schemes that beat this upper bound.

Our Result. We present an efficient robust secret sharing scheme in the maximal corruption setting $n = 2t + 1$, with a share size of only $m + \widetilde{O}(k)$ bits, for

[1] One could also insist on separately requiring robustness *and* reconstructibility from $t + 1$ correct share. This could always be achieved by adding a standard non-robust threshold secret sharing scheme to a robust scheme, at the cost of adding m additional bits to the share size.

security parameter k (reconstruction failure $\leq 2^{-k}$) and message size m.[2] This is the first such scheme which removes the linear dependence between the share size and the number of parties n.

1.1 Our Techniques

Using MACs. Similarly to several prior schemes [RB89,CFOR12,LP14], we start with the idea of using information-theoretic message authentication codes (MACs) to help the reconstruction procedure identify illegitimate shares. The basic premise is to start with a standard (non-robust) t-out-of-n scheme, such as Shamir's scheme, and have parties authenticate each others' Shamir shares using MACs. Intuitively, this should make it more difficult for an adversary to present compelling false shares for corrupted parties as it would have to forge the MACs under unknown keys held by the honest parties.

The original implementation of this idea by Rabin and Ben-Or [RB89] required each party to authenticate the share of every other party with a MAC having unforgeability security 2^{-k}. Therefore, the keys and tags added an extra $\widetilde{O}(nk)$ overhead to the share of each party. The work of Cevallos et al. [CFOR12] showed that one can use a MAC with a weaker unforgeability security of only $\frac{1}{2}$ since the adversary would have to forge many times to succeed. This reduced the overhead to $\widetilde{O}(k + n)$ bits.

Random Authentication Graph. Our core insight is to have each party only authenticate a relatively small but randomly chosen subset of other parties' shares. This will result in a much smaller overhead in the share size.

More precisely, for each party we choose a random subset of $d = \widetilde{O}(k)$ other parties whose shares to authenticate. We can think of this as a random "authentication graph" $G = ([n], E)$ with out-degree d, having directed edges $(i, j) \in E$ if party i authenticates party j. This graph is stored in a distributed manner where each party i is responsible for storing the information about its d outgoing edges. It is important that this graph is not known to the attacker when choosing which parties to corrupt. In fact, the attacker does not know anything about the outgoing edges of uncorrutped parties.[3]

Requirements and Inefficient Reconstruction. As a first step, let's start by considering an inefficient reconstruction procedure, as this will already high-

[2] The $\widetilde{O}(\cdot)$ hides factors that are poly-logarithmic in k, n and m. This is justified if we think of n, m as some large polynomials in the security parameter k. See Section 5.2 for more details on parameters.

[3] If the graph were chosen at random but known to the attacker in advance, then the attacker could always corrupt a set of t parties none of which are being authenticated by some honest party i and modify their shares. Then the $t+1$ shares corresponding to the t corrupted parties along with honest party i would be consistent and the reconstruction would not be able to distinguish it from the set of $t+1$ honest parties. Therefore it is crucial that the graph is random and unknown.

light several challenges. The reconstruction procedure does not get to see the original graph G but a possibly modified graph $G' = ([n], E')$ where the corrupted parties can modify their set of outgoing edges. However, the edges that originate from uncorrupted parties are the same in G and G'. The reconstruction procedure labels each edge $e \in E'$ as either "good" or "bad" depending on whether the MAC corresponding to that edge verifies.

Let's denote the subset of uncorrupted *honest* parties by $H \subseteq [n]$. Let's also distinguish between corrupted parties where the adversary does not modify the share, which we call *passive corruptions* and denote by $P \subseteq [n]$, and the rest which we call *active corruptions* and denote by $A \subseteq [n]$. Assume that we can ensure that the following requirements are met:

(I) All edges $(i, j) \in E'$ where $i, j \in H \cup P$ are labeled "good".
(II) All edges $(i, j) \in E'$ where $i \in H$ and $j \in A$ are labeled "bad".

In this case, the reconstruction procedure can (inefficiently) identify the set $H \cup P$ by simply finding the *maximum self-consistent set* of vertices $C \subseteq [n]$ such that all of the tags corresponding to edges $(i, j) \in E'$ with $i, j \in C$ are labeled "good". We show that $C = H \cup P$ is the unique maximum consistent set with overwhelming probability (see Section 6). Once we identify the set $H \cup P$ we can simply reconstruct the secret message from the Shamir shares of the parties in $H \cup P$ since these have not been modified.

Implementation: Private MAC and Robust Storage of Tags. Let's now see how to implement the authentication process to satisfy requirements (I) and (II) defined above. A naive implementation, along the lines used in [RB89,CFOR12], would be for each party i to have a MAC key key_i for a d-time MAC (i.e., given the authentication tags of d messages one cannot forge the tag of a new message) and, for each edge $(i, j) \in E$, to create a tag $\sigma_{i \to j} = \mathsf{MAC}_{\mathsf{key}_i}(\widetilde{s}_j)$ where $\widetilde{s}_j$ is the Shamir share of party j. The tags $\sigma_{i \to j}$ would then be stored with party j. In particular, the full share of party i would be

$$s_i = (\widetilde{s}_i, E_i, \mathsf{key}_i, \{\sigma_{j \to i}\}_{(j,i) \in E})$$

where $E_i = \{j \in [n] \ : \ (i, j) \in E\}$ are the outgoing edges for party i.

Unfortunately, there are several problems with this. Firstly, if the adversary corrupts party i, it might modify the values key_i, E_i in the share of party i but keep the Shamir share $\widetilde{s}_i$ intact. This will keep the edges going from honest parties to party i labeled "good" but some of the edges going from party i to honest parties might now be labeled "bad". Therefore we cannot define such party as either passive (this would violate requirement I) or active (this would violate requirement II). Indeed, this would break our reconstruction procedure.

To fix this, when party i authenticates party j, we compute $\sigma_{i \to j} = \mathsf{MAC}_{\mathsf{key}_i}((\widetilde{s}_j, E_j, \mathsf{key}_j))$ where we authenticate the values E_j, key_j along with the Shamir share $\widetilde{s}_j$. This prevents party j from being able to modify these components without being detected. Therefore we can define a party as active if any of the components $\widetilde{s}_j, E_j, \mathsf{key}_j$ are modified and passive otherwise.

Unfortunately, there is still a problem. An adversary corrupting party j might keep the components $\widetilde{s}_j, E_j, \mathsf{key}_j$ intact but modify some subset of the tags $\sigma_{i\to j}$. This will make some of edges going from honest parties to j be labeled "good"and some "bad", which violates property I.

To fix this, we don't store tags $\sigma_{i\to j}$ with party j but rather we store all the tags in a distributed manner among the n parties in a way that guarantees recoverability even if t parties are corrupted. However, we do not provide any privacy guarantees for these tags and the adversary may be able to learn all of them in full after corrupting t parties. We call this *robust distributed storage* (without privacy), and show that we can use it to store the tags without additional asymptotic overhead. The fact that the tags are not stored privately requires us to use a special type of *private (randomized) MAC* where the tags do not reveal anything about the authenticated messages even given the secret key. With this implementation, we can guarantee that requirements I, II are satisfied.

Efficient Reconstruction using Graph Bisection. To get an efficient reconstruction procedure, we need to solve the following *graph identification problem*. An adversary partitions vertices $V = [n]$ into three sets H, P, A. We know that the out-going edges from H are chosen randomly and that the edges are labeled as either "good" or "bad" subject to requirements (I), (II) above. The goal is to identify $H \cup P$. We know that, with overwhelming probability, $H \cup P$ is the unique maximal consistent set having no bad edges between its vertices, but its not clear how to identify it efficiently.

Let's consider two cases of the above problem depending on the size of the passive set P. If P is of size $\Omega(n/\log n)$, then we can distinguish between vertices in A and $H \cup P$ by essentially counting the number of incoming bad edges of each vertex. In particular, the vertices in $H \cup P$ are likely to have noticably fewer in-coming good edges (only from A of size $(1/2 - \Omega(1/\log n))n$) than the vertices in A (which have in-coming bad edged from H of size $n/2$). [4]

On the other hand, if P is sufficiently small, of size $\le cn/\log n$ for some chosen constent c, then the graph can be bisected into components H and $A \cup P$ (whose sizes only differ by 1) with only $cnd/\log n$ good edges crossing from H to $A \cup P$ (only the edges from H to P). We use the fact that there exists an efficient $O(\log n)$ approximation algorithm to the *minimum bisection problem*. This is a classic NP-hard optimization problem [GJS76,FK02], and the best known polynomial-time algorithm is an $O(\log n)$-approximation algorithm due to [Räc08]. In particular, if c is chosen small enough, this allows us to bisect the graph it into two components X_0, X_1 with only (say) a $.1nd$ good edges crossing from X_0 to X_1. This must mean that one of X_0 or X_1 contains .9 fraction of the vertices in H (otherwise, if the H vertices were split more evenly, there would

[4] This is an oversimplification; for example it may be the case that many actively corrupted parties point a bad edge at an honest party to make it look bad. Our full solution is more complex, but essentially relies on the fact that this type of mis-classification cannot happen too often.

be many more edges crossing) and this is sufficient to then completely identify all of $H \cup P$.

There are many details to fill in for the above high-level description, but one major issue is that we only have efficient approximations for the graph bisection problem in *undirected* graphs. However, in the above scenario, we are only guaranteed that there are few good edges from H to $A \cup P$ but there may be many good edges in the reverse direction. To solve this problem, we need to modify the scheme so that, for any edge $(i, j) \in E$ corresponding to party i using its key to authenticate the share of party j with a tag $\sigma_{i \to j}$, we also add a "reverse-authentication" tag $\sigma_{i \leftarrow j}$ where we authenticate the share of party i under the key of party j. This ensures that edges from active parties to honest parties are bad. Therefore, when P is small, there are very few good edges between H and $A \cup P$ in either direction and we can use an algorithm for the undirected version of the graph bisection problem.

Parallel Repetition and Parameters. A naive instantiation of the above scheme would require a share size of $m + \widetilde{O}(k^2)$ since we need $O(k)$ tags per party and each tag needs to have length $O(k)$. To reduce the share size further, we rely on a variant of the idea of Cevallos et al. [CFOR12] where it sufficed to pick the MAC parameters so as to only provide weak security. However, we do this more abstractly. We pick the parameters for our entire scheme so as to only provide weak security so that the correct message is recovered with probability (say) $\geq 3/4$. We then use $O(k)$ parallel copies of this scheme to amplify security. The reconstruction outputs the majority value. One subtlety is that all of the copies needs to use the same underlying Shamir shares since we don't want a multiplicative blowup in the message size m. We show that this does not hurt security.

2 Notation and Preliminaries

Foe $n \in \mathbb{N}$, we let $[n] := \{1, \dots, n\}$. If X is a distribution or a random variable, we let $x \leftarrow X$ denote the process of sampling a value x according to the distribution X. If A is a set, we let $a \leftarrow A$ denote the process of sampling a uniformly at random from A. If f is a randomized algorithm, we let $f(x; r)$ denote the output of f on input x with randomness r. We let $f(x)$ be a random variable for $f(x; r)$ with random r.

Sub-Vector Notation. For a vector $\boldsymbol{s} = (s_1, \dots, s_n)$ and a set $I \subseteq [n]$, we let $\boldsymbol{s}_I$ denote the vector consisting only of values in indices $i \in I$; we will represent this as $\boldsymbol{s}_I = (s'_1, \dots, s'_n)$ with $s'_i = s_i$ for $i \in I$ and $s'_i = \perp$ for $i \notin I$.

Graph Notation. For a (directed) graph $G = (V, E)$, and sets $X, Y \subseteq V$, define $E_{X \to Y}$ as the set of edges from X to Y; i.e. $E_{X \to Y} = \{(v_1, v_2) \in E \mid v_1 \in X, v_2 \in Y\}$.

2.1 Chernoff Bounds

We rely on an extension of the standard Chernoff bounds to variables with negative correlation [PS97]. For example, this models sampling without replacement. The following definition and theorem are taken from [AD11].

Definition 1 (Negative Correlation). *Let $X_1, \ldots, X_n$ be binary random variables. We say that they are* negatively correlated *if for all $I \subseteq [n]$:*

$$\Pr\left[\bigwedge_{i \in I} \{X_i = 1\}\right] \leq \prod_{i \in I} \Pr[X_i = 1] \quad \textit{and} \quad \Pr\left[\bigwedge_{i \in I} \{X_i = 0\}\right] \leq \prod_{i \in I} \Pr[X_i = 0].$$

Theorem 1 ((Variant of) Chernoff-Hoeffding). *Let $X_1, \ldots, X_n$ be independent or negatively correlated binary random variables, let $X = \sum_{i=1}^{n} X_i$ and let $\mu = \mathbb{E}[X]$. Then for any $0 < \delta < 1$:*

$$\Pr[X < (1-\delta)\mu] \leq e^{-\delta^2\mu/2} \quad , \quad \Pr[X \geq (1+\delta)\mu] \leq e^{-\delta^2\mu/3}.$$

2.2 Hash Functions, Polynomial Evaluation

Definition 2 (Universal Hashing). *Let $\mathcal{H} = \{H_k : \mathcal{U} \to \mathcal{V}\}_{k \in \mathcal{K}}$ be family of hash functions. We say that $\mathcal{H}$ is ε-universal if for all $x, x' \in \mathcal{U}$ with $x \neq x'$ we have $\Pr_{k \leftarrow \mathcal{K}}[H_k(x) = H_k(x')] \leq \varepsilon$.*

Polynomial Evaluation. Let $\mathbb{F}$ be a finite field. Define the *polynomial evaluation* function $\mathsf{PEval} : \mathbb{F}^d \times \mathbb{F} \to \mathbb{F}$ as $\mathsf{PEval}(\boldsymbol{a}, x) = \sum_{i=1}^{d} a_i x^i$. We rely on two properties of this hash.

In appendix A.1 we analyze some useful properties of this function.

2.3 Graph Bisection

Let $G = (V, E)$ be an undirected graph. Let (V_1, V_2) be a partition of its edges. The *cross edges* of (V_1, V_2) are the edges in $E_{V_1 \to V_2}$.

Given an undirected graph $G = (V, E)$ with an even number of vertices $|V| = 2t$ a *graph bisection* for G is a partition (V_1, V_2) of V such that $|V_1| = t = |V_2|$. We also extend the notion of a graph bisection to graphs with an odd number of vertices $|V| = 2t + 1$ by defining a bisection to be a partition with $|V_1| = t, |V_2| = t + 1$.

Definition 3 (Approximate Graph Bisection Algorithm). *Let $G = (V, E)$ be an undirected graph with n vertices. Assume that G has a graph bisection with m cross edges. An algorithm Bisect that takes as input G and outputs a bisection U_1, U_2 with at most δm cross edges is called* δ-approximate graph bisection algorithm*.*

We remark that standard definitions of graphs bisection only consider the case where $n = 2t$ is even. However, given any δ-approximate graph bisection algorithm that works in the even case, we can generically adapt it to also work in the odd case $n = 2t+1$. In particular, given a graph $G = (V, E)$ with $|V| = 2t+1$ vertices, we can construct a graph $G' = (V \cup \{\perp\}, E)$ with an added dummy vertex $\perp$ that has no outgoing or incoming edges. We then run the δ-approximate graph bijection algorithm that works for an even number of vertices on G' to get a bisection U'_1, U'_2 where, without loss of generality, we assume that U'_1 contains the vertex $\perp$. By simply taking U_1 to be U'_1 with $\perp$ removed and $U_2 = U'_2$ we get a δ-approximate bijection for the graph G.

Theorem 2 ([Räc08] (Section 3, "Min Bisection.")). *There exists a polynomial-time $O(\log(n))$-approximate graph bisection algorithm, where n is the number of vertices.*

3 Definition of Robust Secret Sharing

Throughout the rest of the paper, we use the following notation:

- t denotes the number of players that are arbitrarily corrupt.
- $n = 2t + 1$ denotes the number of players in the scheme.
- $\mathcal{M}$ is the message space.

Definition 4 (Robust Secret Sharing). *A t-out-of-n, δ-robust secret sharing scheme over a message space $\mathcal{M}$ and share space $\mathcal{S}$ is a tuple $(\mathsf{Share}, \mathsf{Rec})$ of algorithms that run as follows:*

$\mathsf{Share}(\mathsf{msg}) \to (s_1, \ldots, s_n)$**:** *This is a randomized algorithm takes as input a message $\mathsf{msg} \in \mathcal{M}$ and outputs a sequence of shares $s_1, \ldots, s_n \in \mathcal{S}$.*

$\mathsf{Rec}(s_1, \ldots, s_n) \to \mathsf{msg}'$**:** *This is a deterministic algorithm takes as input n shares $(s_1, \ldots, s_n)$ with $s_i \in \mathcal{S} \cup \perp$ and outputs a message $\mathsf{msg}' \in \mathcal{M}$.*

We require perfect correctness, meaning that for all $\mathsf{msg} \in \mathcal{M}$: $\Pr[\mathsf{Rec}(\mathsf{Share}(\mathsf{msg})) = \mathsf{msg}] = 1$. Moreover, the following properties hold:

Perfect Privacy: *Any t out of n shares of a secret give no information on the secret itself. More formally, for any $\mathsf{msg}, \mathsf{msg}' \in \mathcal{M}$, any $I \subseteq [n]$ of size $|I| = t$, the distributions $\mathsf{Share}(\mathsf{msg})_I$ and $\mathsf{Share}(\mathsf{msg}')_I$ are identical.*

Perfect Reconstruction with Erasures: *The original secret can be reconstructed from any $t+1$ correct shares. More formally, for any $\mathsf{msg} \in \mathcal{M}$ and any $I \subseteq [n]$ with $|I| = t + 1$ we have $\Pr[\mathsf{Rec}(\mathsf{Share}(\mathsf{msg})_I) = \mathsf{msg}] = 1$.*

Adaptive δ-Robustness: *An adversary that modifies up to t shares can cause the wrong secret to be recovered with probability at most δ. We consider an adaptive version where the adversary can choose which shares to corrupt adaptively. More formally, we define the experiment $\boldsymbol{Exp}(\mathsf{msg}, \mathsf{Adv})$ with some secret $\mathsf{msg} \in \mathcal{M}$ an interactive adversary Adv.*

Exp$(\mathsf{msg}, \mathsf{Adv})$**:** *is defined as follows:*

E.1. Sample $\boldsymbol{s} = (s_1, \ldots, s_n) \leftarrow \mathsf{Share}(\mathsf{msg})$.

E.2. Set $I := \varnothing$. Repeat the following while $|I| \le t$.

– Adv *chooses* $i \in [n] \setminus I$.
– *Update* $I := I \cup \{i\}$ *and give* s_i *to* Adv.

E.3. Adv *outputs modified shares* $s'_i \ : \ i \in I$ *and we define* $s'_i := s_i$ *for* $i \notin I$.

E.4. Compute $\mathsf{msg}' = \mathsf{Rec}(s'_1, \ldots, s'_n)$.

E.5. If $\mathsf{msg}' \neq \mathsf{msg}$ *output 1 else 0.*

We require that for any (unbounded) adversary Adv *and any* $\mathsf{msg} \in \mathcal{M}$ *we have*

$$\Pr[\boldsymbol{Exp}(\mathsf{msg}, \mathsf{Adv}) = 1] \leq \delta.$$

Remarks. We note that since privacy and correctness are required to hold perfectly (rather than statistically) there is no difference between defining non-adaptive and adaptive variants. In other words, we could also define adaptive privacy where the adversary gets to choose which shares to see adaptively, but this is already implied by our non-adaptive definition of perfect privacy. We also note that when $n = 2t + 1$ then robustness implies a statistically secure reconstruction with erasures. However, since we can even achieve perfect reconstruction with erasures, we define it as a separate property.

Definition 5 (Non-Robust Secret Sharing). *We will say that a scheme is a* non-robust *t-out-of-n secret sharing scheme, if it satisfies the above definition with* $\delta = 1$.

Using Shamir secret sharing, we get a non-robust t-out-of-n secret sharing for any $t < n$ where the share size is the same as the message size.

4 The Building Blocks

In this section we introduce the building blocks of our robust secret sharing scheme: Robust Distributed Storage, Private MACs, and Graph Reconstruction.

4.1 Robust Distributed Storage

A robust distributed storage scheme allows us to store a public value among n parties, t of which may be corrupted. There is no secrecy requirement on the shared value. However, we require robustness: if the adversary adaptively corrupts t of the parties and modifies their shares, the reconstruction procedure should recover the correct value with overwhelming probability. In some sense, we can think of this as an error-correcting code where shares correspond to codeword symbols. However, the encoding procedure can be randomized and the adversary only sees the t corrupted positions when deciding on the errors. These restrictions allow us to achieve better parameters than what is normally possible with error-correcting codes.

Definition 6. *A* t-out-of-n, δ-robust distributed storage *over a message space* $\mathcal{M}$ *is a tuple of algorithms* $(\mathsf{Share}, \mathsf{Rec})$ *having the same syntax as robust secret* *sharing, and satisfying the δ-robustness property. However, it need not satisfy the privacy or perfect reconstruction with erasures properties.*

We would like to construct such schemes for $n = 2t+1$ and for a message of size m so that the share of each party is only $O(m/n)$ bits. These parameters are already beyond the reach of error-correcting codes for worst-case errors. We construct a simple robust distributed storage scheme by combining list-decoding and universal hashing.

List Decoding. In list-decoding, the requirement to decode to a unique codeword is relaxed, and it is only required to obtain a polynomially sized list of potential candidates that is guaranteed to include the correct codeword. We can simply use Reed-Solomon codes and the list-decoding algorithm provided by Sudan [Sud97] (better parameters are known but this suffices for our needs):

Theorem 3 ([Sud97]). *A Reed-Solomon code formed by evaluating a degree d polynomial on n points can be efficiently list-decoded to recover from $e < n - \sqrt{2dn}$ errors with a list of size $L \leq \sqrt{2n/d}$.*

Setting $d = \lfloor n/8 \rfloor$, we can then therefore recover from t out of $n = 2t+1$ errors and obtain a list of size $L \leq \sqrt{2n/d} = O(1)$.

Construction of Robust Distributed Storage. Let t be some parameter, let $n = 2t+1$, and let $\mathbb{F}$ be a field of size $|\mathbb{F}| = 2^u$ with $|\mathbb{F}| > n$. Let $\mathcal{H} = \{H_k : \mathbb{F}^{d+1} \to \mathbb{F}\}_{k \in \mathbb{F}}$ be an ε-universal hash function. For concreteness we can use polynomial evaluation $H_k(\boldsymbol{a}) = \mathsf{PEval}(\boldsymbol{a}, k))$ which achieves $\varepsilon = (d+1)/2^u$ (see Claim A.1). We use list-decoding for Reed Solomon with degree $d = \lfloor n/8 \rfloor$ which allows us to recover from t out of n errors with a list size $L = O(1)$. We construct a δ-robust distributed storage scheme with message space $\mathcal{M} = \mathbb{F}^{d+1}$, meaning that the messages have bit-size $m = u(d+1) = O(un)$, and with robustness $\delta = nL\varepsilon = O(n^2)/2^u$ and share size $3u$.

- $(s_1, \ldots, s_n) \leftarrow \mathsf{Share}(\mathsf{msg})$. Encodes $\mathsf{msg} \in \mathbb{F}^{d+1}$ using the Reed-Solomon code by interpreting it as a degree d polynomial and evaluating it on n points to get the Reed-Solomon codeword $(\hat{s}_1, \ldots, \hat{s}_n) \in \mathbb{F}^n$. Choose random values $k_1, \ldots, k_n \leftarrow \mathbb{F}$ and define the shares $s_i = (\hat{s}_i, k_i, H_{k_i}(\mathsf{msg})) \in \mathbb{F}^3$.
- $\mathsf{msg}' \leftarrow \mathsf{Rec}(s'_1, \ldots, s'_n)$. Parse $s'_i = (\hat{s}'_i, k'_i, y'_i)$. Use list-decoding on the codeword $(\hat{s}'_1, \ldots, \hat{s}'_n) \in \mathbb{F}^n$ to recover a list of $L = O(1)$ possible candidates $\mathsf{msg}^{(1)}, \ldots, \mathsf{msg}^{(L)} \in \mathbb{F}^{d+1}$ for the message. Output the first value $\mathsf{msg}^{(j)}$ that agrees with the majority of the hashes:

$$|\{i \in [n] \; : \; H_{k'_i}(\mathsf{msg}^{(j)}) = y'_i\}| \geq t+1.$$

Theorem 4. *The above scheme is a t-out-of-n, δ-robust distributed storage scheme for $n = 2t+1$ where, for any $u \geq \log n$ we have messages of length $m = \lfloor n/8 \rfloor u$, shares of length $3u$ and robustness $\delta = O(n^2)/2^u$.*

We prove 4 in appendix B.1.

4.2 Private Labeled MAC

As a tool in our construction of robust secret sharing schemes, we will use a new notion of an information-theoretic message-authentication code (MAC) that has additional privacy guarantees.

The message authentication code $\sigma = \mathsf{MAC}_{\mathsf{key}}(\mathsf{lab}, \mathsf{msg}, r)$ takes as input a *label* lab, a *message* msg, and some additional *randomness* r. The randomness is there to ensure privacy for the message msg even given key, σ.

Definition 7 (Private Labeled MAC). *An* (ℓ, ε) private MAC *is a family of functions* $\{\mathsf{MAC}_{\mathsf{key}} : \mathcal{L} \times \mathcal{M} \times \mathcal{R} \to \mathcal{T}\}_{\mathsf{key} \in \mathcal{K}}$ *with key-space* $\mathcal{K}$*, message space* $\mathcal{M}$*, label space* $\mathcal{L}$*, randomness space* $\mathcal{R}$*, and tag space* $\mathcal{T}$*. It has the following properties:*

Authentication: *For any* ℓ *values* $(\mathsf{lab}_i, \mathsf{msg}_i, r_i, \sigma_i) \in \mathcal{L} \times \mathcal{M} \times \mathcal{R} \times \mathcal{T} \ : \ i = 1, \ldots, \ell$ *such that the labels* lab_i *are distinct, and for any* $(\mathsf{lab}', \mathsf{msg}', r', \sigma') \in \mathcal{L} \times \mathcal{M} \times \mathcal{R} \times \mathcal{T}$ *such that* $(\mathsf{lab}', \mathsf{msg}', r') \notin \{(\mathsf{lab}_i, \mathsf{msg}_i, r_i)\}_{i \in [\ell]}$ *we have:*

$$\Pr_{\mathsf{key} \leftarrow \mathcal{K}}[\mathsf{MAC}_{\mathsf{key}}(\mathsf{lab}', \mathsf{msg}', r') = \sigma' \mid \{\mathsf{MAC}_{\mathsf{key}}(\mathsf{lab}_i, \mathsf{msg}_i, r_i) = \sigma_i\}_{i \in [\ell]}] \leq \varepsilon.$$

This implies that even after seeing the authentication tags σ_i for ℓ tuples $(\mathsf{lab}_i, \mathsf{msg}_i, r_i)$ with distinct labels lab_i, an adversary cannot come up with a valid tag σ' for any new tuple $(\mathsf{lab}', \mathsf{msg}', r')$.

Privacy Over Randomness: *For any* ℓ *distinct labels* $\mathsf{lab}_1, \ldots, \mathsf{lab}_\ell$*, any keys* $\mathsf{key}_1, \ldots, \mathsf{key}_\ell \in \mathcal{K}$*, and any* $\mathsf{msg} \in \mathcal{M}$*, the* ℓ *values* $\sigma_1 = \mathsf{MAC}_{\mathsf{key}_1}(\mathsf{lab}_1, \mathsf{msg}, r), \ldots, \sigma_\ell = \mathsf{MAC}_{\mathsf{key}_\ell}(\mathsf{lab}_\ell, \mathsf{msg}, r)$ *are uniformly random and independent in* $\mathcal{T}$ *over the choice of* $r \leftarrow \mathcal{R}$*.*

This says that the tags σ_i do not reveal any information about the message msg, or even about the labels lab_i and the keys key_i, as long as the randomness r is unknown.

Privacy Over Keys: *Let* $(\mathsf{lab}_i, \mathsf{msg}_i, r_i) \in \mathcal{L} \times \mathcal{M} \times \mathcal{R} \ : \ i = 1, \ldots, \ell$ *be* ℓ *values such that the labels* lab_i *are distinct. Then the* ℓ *values* $\sigma_1 = \mathsf{MAC}_{\mathsf{key}}(\mathsf{lab}_1, \mathsf{msg}_1, r_1), \ldots, \sigma_\ell = \mathsf{MAC}_{\mathsf{key}}(\mathsf{lab}_\ell, \mathsf{msg}_\ell, r_\ell)$ *are uniformly random and independent in* $\mathcal{T}$ *over a random* $\mathsf{key} \leftarrow \mathcal{K}$*.*

This says that the tags σ_i do not reveal any information about the values $(\mathsf{lab}_i, \mathsf{msg}_i, r_i)$ as long as key is unknown.

Construction. Let $\mathbb{F}$ and $\mathbb{F}'$ be finite fields such that $|\mathbb{F}'| \geq |\mathcal{L}|$ and $|\mathbb{F}| \geq |\mathbb{F}'| \cdot |\mathcal{L}|$. We assume that we can identify the elements of $\mathcal{L}$ as either a subset of $\mathbb{F}'$ or $\mathbb{F}$ and we can also efficiently identify tuples in $\mathbb{F}' \times \mathcal{L}$ as a subset of $\mathbb{F}$. Let $\mathcal{M} = \mathbb{F}^m$, $\mathcal{R} = \mathbb{F}^\ell$, $\mathcal{K} = \mathbb{F}^{\ell+1} \times (\mathbb{F}')^{\ell+1}$, $\mathcal{T} = \mathbb{F}$. Define $\mathsf{MAC}_{\mathsf{key}}(\mathsf{lab}, \mathsf{msg}, r)$ as follows:

- Parse $\mathsf{key} = (\mathsf{key}_1, \mathsf{key}_2)$ where $\mathsf{key}_1 \in (\mathbb{F}')^{\ell+1}, \mathsf{key}_2 \in \mathbb{F}^{\ell+1}$.
- Compute $\mathsf{key}_1^{\mathsf{lab}} := \mathsf{PEval}(\mathsf{key}_1, \mathsf{lab})$, $\mathsf{key}_2^{\mathsf{lab}} := \mathsf{PEval}(\mathsf{key}_2, \mathsf{lab})$ by identifying $\mathsf{lab} \in \mathcal{L}$ as an element of $\mathbb{F}'$ and $\mathbb{F}$ respectively.

– Output $\sigma := \mathsf{PEval}(\ (r, \mathsf{msg})\ ,\ (\mathsf{lab}, \mathsf{key}_1^{\mathsf{lab}})\ ) + \mathsf{key}_2^{\mathsf{lab}}$. Here we interpret $(r, \mathsf{msg}) \in \mathcal{R} \times \mathcal{M} = \mathbb{F}^{\ell+m}$ as a vector of coefficients in $\mathbb{F}$ and we identify $(\mathsf{lab}, \mathsf{key}_{\mathsf{lab}}^1) \in \mathcal{L} \times \mathbb{F}'$ as an element of $\mathbb{F}$.

Theorem 5. *The above construction is an (ℓ, ε) private MAC, where $\varepsilon = \frac{m+\ell}{|\mathbb{F}'|}$.*

We prove theorem 5 in appendix B.2

4.3 Graph Identification

Here, we define an algorithmic problem called the graph identification problem. We then prove that this problem can be solved efficiently.

Definition 8 (Graph Challenge). *A* graph challenge $\mathsf{Gen}^{\mathsf{Adv}}(n, t, d)$ *is a randomized process that outputs a directed graph $G = (V = [n], E)$, where each vertex $v \in V$ has out-degree d, along with a labeling $L : E \to \{\mathsf{good}, \mathsf{bad}\}$. The process is parameterized by an adversary* Adv *and proceeds as follows:*

Adversarial Components. *The adversary $\mathsf{Adv}(n, t, d)$ does the following:*

1. *It partitions $V = [n]$ into three disjoint sets H, A, P such that $V = H \cup A \cup P$ and $|A \cup P| = t$.*
2. *It chooses the set of edges $E_{A\cup P \to V}$ that originate from $A \cup P$ arbitrarily subject to each $v \in A \cup P$ having out-degree d and no self-loops.*
3. *It chooses the labels $L(e)$ arbitrarily for each edge $e \in E_{A\to(A\cup P)} \cup E_{(A\cup P)\to A}$.*

Honest Components. *The procedure* Gen *chooses the remaining edges and labels as follows:*

1. *It chooses the edges $E_{H\to V}$ that originate from H uniformly at random subject to each vertex having out-degree d and no self-loops. In particular, for each $v \in H$ it selects outgoing edges to a set of d vertices chosen uniformly at random (without replacement) from $V \setminus \{v\}$.*
2. *It sets $L(e) := \mathsf{bad}$ for all $e \in E_{H\to A} \cup E_{A\to H}$.*
3. *It sets $L(e) := \mathsf{good}$ for all $e \in E_{(H\cup P)\to(H\cup P)}$.*

Output. *Output $(G = (V, E), L, H, A, P)$.*

Theorem 6. *There exists an polynomial time algorithm* $\mathsf{GraphID}$*, called the* graph identification algorithm*, that takes as input a directed graph $G = (V = [n], E)$ a labeling $L : V \to \{\mathsf{good}, \mathsf{bad}\}$ and outputs a set $B \subseteq V$, such that for any* Adv*, and any t, $n = 2t + 1$, and any d we have*

$$\Pr[B = H \cup P \ :\ B \leftarrow \mathsf{GraphID}(G, L), (G, L, H, A, P) \leftarrow \mathsf{Gen}^{\mathsf{Adv}}(n, t, d)] \geq 1 - 2^{-\Omega(d/(\log^2 n))}$$

In Section 7, we prove Theorem 6 by providing an algorithm $\mathsf{GraphID}$ with the required properties.

5 Construction of Robust Secret Sharing

Let $t, n = 2t+1$ be parameters that are given to us, and let $\mathcal{M}$ be a message space. Let d be a parameter such that the t-out-of-n graph reconstruction problem with degree d has robustness δ_{gi}.

Let $(\mathsf{Share}_{nr}, \mathsf{Rec}_{nr})$ be a t-out-of-n *non-robust* secret sharing scheme with message space $\mathcal{M}$ and share space $\mathcal{S}_{nr} = \mathcal{M}$ (e.g., Shamir secret sharing).

Let $\{\mathsf{MAC}_{\mathsf{key}} : \mathcal{L} \times \mathcal{M}_{mac} \times \mathcal{R} \to \mathcal{T}\}_{\mathsf{key} \in \mathcal{K}}$ be an $(\ell, \varepsilon_{mac})$ *private MAC* with label space $\mathcal{L} = [n]^2 \times \{0,1\}$ and message space $\mathcal{M}_{mac} = \mathcal{M} \times [n]^d \times \mathcal{K}$, where $\ell \geq 3d$.

Finally, let $(\mathsf{Share}_{rds}, \mathsf{Rec}_{rds})$ be a t-out-of-n robust distributed storage (no privacy) with message space $\mathcal{M}_{rds} = \mathcal{T}^{2dn}$, share space $\mathcal{S}_{rds}$ and with robustness δ_{rds}.

Our robust secret sharing scheme $(\mathsf{Share}, \mathsf{Rec})$ is defined as follows.

$\mathsf{Share}(\mathsf{msg})$. On input a message $\mathsf{msg} \in \mathcal{M}$, the sharing procedure proceeds as follows:

S.1. Choose $(\widetilde{s}_1, \ldots, \widetilde{s}_n) \leftarrow \mathsf{Share}_{nr}(\mathsf{msg})$ to be a non-robust secret sharing of msg.

S.2. Choose a uniformly random directed graph $G = ([n], E)$ with out-degree d, in-degree at most $2d$ and no self-loops as follows:

(a) For each $i \in [n]$ choose a random set $E_i \subseteq [n] \setminus \{i\}$ of size $|E_i| = d$. Set

$$E := \{(i,j) \ : \ i \in [n], j \in E_i\}.$$

(b) Check if there is any vertex in G with in-degree $> 2d$. If so, go back to step (a). [5]

S.3. For each $i \in [n]$, sample a random MAC key $\mathsf{key}_i \leftarrow \mathcal{K}$ and MAC randomness $r_i \leftarrow \mathcal{R}$.
For each $j \in E_i$ define

$$\sigma_{i \to j} := \mathsf{MAC}_{\mathsf{key}_i}((i,j,0), (\widetilde{s}_j, E_j, \mathsf{key}_j), r_j) \quad , \quad \sigma_{i \leftarrow j} := \mathsf{MAC}_{\mathsf{key}_j}((i,j,1), (\widetilde{s}_i, E_i, \mathsf{key}_i), r_i).$$

where we treat $(i,j,0), (i,j,1) \in \mathcal{L}$ as a label, and we treat $(\widetilde{s}_j, E_j, \mathsf{key}_j) \in \mathcal{M}_{mac}$ as a message.

S.4. For each $i \in [n]$ define $\mathsf{tags}_i = \{(\sigma_{i \to j}, \sigma_{i \leftarrow j})\}_{j \in E_i} \in \mathcal{T}^{2d}$ and define $\mathsf{tags} = (\mathsf{tags}_1, \ldots, \mathsf{tags}_n) \in \mathcal{T}^{2nd}$. Choose $(p_1, \ldots, p_n) \leftarrow \mathsf{Share}_{rds}(\mathsf{tags})$ using the robust distributed storage scheme.

S.5. For $i \in [n]$, define $s_i = (\widetilde{s}_i, E_i, \mathsf{key}_i, r_i, p_i)$ to be the share of party i. Output $(s_1, \ldots, s_n)$.

[5] This happens with negligible probability. However, we include it in the description of the scheme in order to get perfect rather than statistical privacy.

$\mathsf{Rec}(s'_1, \ldots, s'_n)$. On input $s'_1, \ldots, s'_n$ with $s'_i = (\widetilde{s}'_i, E'_i, \mathsf{key}'_i, r'_i, p'_i)$ do the following.

R.0. If there is a set of exactly $t+1$ values $W = \{i \in [n] \ : \ s'_i \neq \bot\}$ then output $\mathsf{Rec}_{nr}((\widetilde{s}'_i)_{i \in W})$. Else proceed as follows.

R.1. Reconstruct $\mathsf{tags}' = (\mathsf{tags}'_1, \ldots, \mathsf{tags}'_n) = \mathsf{Rec}_{rds}(p'_1, \ldots, p'_n)$. Parse $\mathsf{tags}'_i = \{(\sigma'_{i \to j}, \sigma'_{i \leftarrow j})\}_{j \in E'_i}$.
R.2. Define a graph $G' = ([n], E')$ by setting $E' := \{(i, j) \ : \ i \in [n], j \in E'_i\}$.
R.3. Assign a label $L(e) \in \{\mathsf{good}, \mathsf{bad}\}$ to each edge $e = (i, j) \in E'$ as follows. If the following holds:

$$\sigma'_{i \to j} = \mathsf{MAC}_{\mathsf{key}'_i}((i,j,1), (\widetilde{s}'_j, E'_j, \mathsf{key}'_j), r'_j) \quad \text{and} \quad \sigma'_{i \leftarrow j} = \mathsf{MAC}_{\mathsf{key}'_j}((i,j,1), (\widetilde{s}'_i, E'_i, \mathsf{key}'_i), r'_i)$$

then set $L(e) := \mathsf{good}$, else set $L(e) := \mathsf{bad}$.
R.4. Call the graph identification algorithm to compute $B \leftarrow \mathsf{GraphID}(G', L)$.
R.5. Choose a subset $B' \subseteq B$ of size $|B'| = t+1$ arbitrarily and output $\mathsf{Rec}_{nr}((\widetilde{s}'_i)_{i \in B'})$.

Lemma 1. *The scheme* $(\mathsf{Share}, \mathsf{Rec})$ *satisfies perfect privacy.*

Proof. Let $I \subseteq [n]$ be of size $|I| = t$ and let $\mathsf{msg}, \mathsf{msg}' \in \mathcal{M}$ be any two values. We define a sequence of hybrids as follows:

Hybrid 0: This is $\mathsf{Share}(\mathsf{msg})_I = (s_i)_{i \in I}$. Each $s_i = (\widetilde{s}_i, E_i, \mathsf{key}_i, r_i, p_i)$.
Hybrid 1: In this hybrid, we change the sharing procedure to simply choose all tags $\sigma_{i \to j}$ and $\sigma_{j \leftarrow i}$ for any $j \notin I$ uniformly and independently at random. This is identically distributed by the "privacy over randomness" property of the MAC. In particular, we rely on the fact that the adversary does not see r_j and that there are at most $\ell = 3d$ tags of the form $\sigma_{i \to j}$ and $\sigma_{j \leftarrow i}$ for any $j \notin I$ corresponding to the total degree of vertex j. These are the only tags that rely on the randomness r_j and they are all created with distinct labels.
Hybrid 2: In this hybrid, we choose $(\widetilde{s}_1, \ldots, \widetilde{s}_n) \leftarrow \mathsf{Share}_{nr}(\mathsf{msg}')$. This is identically distributed by the perfect privacy of the non-robust secret sharing scheme. Note that in this hybrid, the shares $s_i \ : \ i \in I$ observed by the adversary do not contain any information about $\widetilde{s}'_j \ : \ j \notin I$.
Hybrid 3: This is $\mathsf{Share}(\mathsf{msg}')_I = (s_i)_{i \in I}$. Each $s_i = (\widetilde{s}_i, E_i, \mathsf{key}_i, r_i, p_i)$. This is identically distributed by the "privacy over randomness" property of the MAC, using same argument as going from Hybrid 0 to 1.

Lemma 2. *The scheme* $(\mathsf{Share}, \mathsf{Rec})$ *satisfies perfect reconstruction with erasures.*

Proof. This follows directly from the fact that the non-robust scheme $(\mathsf{Share}_{nr}, \mathsf{Rec}_{nr})$ satisfies perfect reconstruction with erasures and therefore step R.0 of reconstruction is guaranteed to output the correct answer when there are exactly t erasures.

Lemma 3. *The scheme* $(\mathsf{Share}, \mathsf{Rec})$ *is* δ*-robust for* $\delta = \delta_{rds} + \delta_{gi} + dn\varepsilon_{mac} + n2^{-d/3}$.

The formal proof is given in appendix B.3; here, we give a high-level overview. For simplicity, let's consider a non-adaptive robustness experiment where the adversary has to choose the set $I \subseteq [n]$, $|I| = t$ of parties to corrupt at the very beginning of the game (in the full proof, we handle adaptive security). Let $s_i = (\widetilde{s}_i, E_i, \mathsf{key}_i, r_i, p_i)$ be the shares created by the sharing procedure and let $s'_i = (\widetilde{s}'_i, E'_i, \mathsf{key}'_i, r'_i, p'_i)$ be the modified shares submitted by the adversary (for $i \notin I$, we have $s'_i = s_i$). Let us define the set of *actively modified* shares:

$$A = \{i \in I \ : \ (\widetilde{s}'_i, E'_i, \mathsf{key}'_i, r'_i) \neq (\widetilde{s}_i, E_i, \mathsf{key}_i, r_i)\}.$$

These are the shares where something was modified beyond (only) the p_i component. Define $H = [n] \setminus I$ and $P = I \setminus A$. To prove robustness, we show that the choice of H, P, A and the labeling created by the reconstruction procedure follow the same distribution as in the graph identification problem. Therefore the graph identification procedure outputs $B = H \cup P$. We show this by defining a sequence of "hybrid" distributions.

1. During reconstruction process, instead of recovering $\mathsf{tags}' = \mathsf{Rec}_{rds}(p'_1, \ldots, p'_n)$ we just set $\mathsf{tags}' = \mathsf{tags}$ to be the correct value chosen by the sharing procedure. By the security of the robust-distributed storage scheme, this modification only changes the experiment with probability δ_{rds}.
2. During the sharing procedure, we can change all of the tags $\sigma_{i \to j}, \sigma_{j \leftarrow i}$ with $j \in [n] \setminus I$ to uniformly random values. This is identically distributed by the "privacy over randomness" property of the MAC with $\ell = 3d$ since the adversary does not see r_j for any such j, and there are at most $3d$ such tags corresponding to the total degree of the vertex j. In particular, this means that such tags do not reveal any (additional) information to the adversary about E_j, key_j for $j \notin I$.
3. During the reconstruction process, when the labeling L is created, we automatically set $L(e) = \mathsf{bad}$ for any edge $e = (i, j)$ or $e = (j, i)$ in E' such that $i \in H, j \in A$ (i.e., one end-point honest and the other active). The only time this changes things is if the adversary manages to create a forged tag of a new value under the key key_i which was used to create at most $\ell = 3d$ tags with distinct labels. Therefore, by the authentication security of the MAC, the probability of this happening for any fixed edge e is ε_{mac}. By taking a union bound over all edges, we get $\leq dn\varepsilon_{mac}$ total probability of an incorrect labeling. Note that, by the definition of the labeling, we are also ensured that $L(e) = \mathsf{good}$ for any edge (i, j) where $i, j \in H \cup P$.
4. During the sharing procedure, we can change all of the tags $\sigma_{i \to j}, \sigma_{j \leftarrow i}$ with $i \in [n] \setminus I$ to uniformly random values. This is identically distributed by the "privacy over keys" property of the MAC with $\ell = 3d$ since the adversary does not see key_i for any such i, and there are at most $3d$ such tags corresponding to the total degree of the vertex i. In particular, this means that the adversary does not learn anything about the edges E_i for $i \in [n] \setminus I$.
5. When we choose the graph $G = ([n], E)$ during the sharing procedure, we no longer require that every vertex has in-degree $\leq 2d$. Instead, we just choose each set $E_i \subseteq [n] \setminus \{i\}$, $|E_i| = d$ uniformly at random. Since the expected

in-degree of every vertex is d, by a Chernoff bound and a union bound over all vertices, the probability that there is some vertex with in-degeee $> 2d$ is $< n2^{-d/3}$.

6. During reconstruction, instead of computing $B \leftarrow \mathsf{GraphID}(G', L)$ we set $B = H \cup P$. We notice that, in the previous hybrid, the distribution of G', L, H, A, P is exactly that of the graph reconstruction game $\mathsf{Gen}^{\mathsf{Adv}'}(n, t, d)$ with some adversary Adv'. In particular, the out-going edges from the honest set H are chosen uniformly at random and the adversary does not see any information about it. Furthermore, the labeling satisfies the properties of the graph identification game. Therefore, by the correctness of the graph identification algorithm, the above modification can only change the outcome of the experiment with probability δ_{gi}.

In the last hybrid, the last step of the reconstruction procedure runs $\mathsf{msg}' = \mathsf{Rec}_{nr}((\widetilde{s}'_i)_{i\in B})$ where $B = [n] \setminus A$ is of size $|B| \geq t+1$ and $\widetilde{s}'_i = \widetilde{s}_i$ for $i \in B$. Therefore $\mathsf{msg}' = \mathsf{msg}$ by the Perfect Reconstruction with Erasures property of the non-robust secret sharing scheme.

5.1 Parameters of Construction

Let $\mathcal{M} = \{0,1\}^m$ and $t, n = 2t+1$ be parameters. Furthermore, let λ be a parameter which we will relate to the security parameter k. We choose the degree of the graph $d = \lambda \log^2 n$, which gives $\delta_{gi} = 2^{-\Omega(\lambda)}$.

We instantiate the non-robust secret scheme $(\mathsf{Share}_{nr}, \mathsf{Rec}_{nr})$ using t-out-of-n Shamir secret sharing over a binary field of size $2^{\max\{m, \lfloor \log n \rfloor + 1\}} = 2^{m+O(\log n)}$.

We instantiate the MAC using the construction from Section 4.2. We choose the field $\mathbb{F}'$ to be a binary field of size $2^{\max\{2\lceil \log n \rceil + 1, \lambda\}} = 2^{O(\lambda + \log n)}$ which is enough to encode a label in $\mathcal{L} = [n]^2 \times \{0,1\}$. We choose the field $\mathbb{F}$ to be of size $|\mathbb{F}| = |\mathbb{F}'| 2^{2\lceil \log n \rceil + 1} = 2^{O(\lambda + \log n)}$ which is enough to encode an element of $\mathbb{F}' \times \mathcal{L}$. We set $\ell = 3d = O(\lambda \log^2 n)$. This means the randomness and keys have length $\log \mathcal{R}, \log \mathcal{K} = O(\lambda \log^2 n(\lambda + \log n))$ and the tags have length $\log \mathcal{T} = O(\lambda + \log n)$. We set the message space of the MAC to be $\mathcal{M}_{mac} = \mathbb{F}^{m_{mac}}$ which needs to be large enough to encode the Shamir share, edges, and a key and therefore we set $m_{mac} = \lceil (m + \log \mathcal{K} + d \log n)/\lambda \rceil = O(m/\lambda + \lambda \log^2 n + \log^3 n)$. This gives security $\varepsilon_{mac} = \frac{m_{mac} + \ell}{|\mathbb{F}'|} = 2^{-\Omega(\lambda + \log m + \log \log n)}$.

Finally, we instantiate the robust distributed storage scheme using the construction from Section 4.1. We need to set the message space $\mathcal{M}_{rds} = \mathcal{T}^{2dn}$ which means that the messages are of length $m_{rds} = 2dn \log \mathcal{T} = O(n\lambda \log n(\lambda + \log n))$. Since we set $m_{rds} = \lfloor n/8 \rfloor u$ we get share length $3u = O(\lambda \log n(\lambda + \log n))$ and we get security $\delta_{rds} = O(n^2)/2^u \leq 2^{-\lambda}$.

With the above we get security

$$\delta \leq \delta_{rds} + \delta_{gi} + dn\varepsilon_{mac} + n2^{-d/3} = 2^{\Omega(-\lambda + \log m + \log n)}$$

By choosing a sufficiently large $\lambda = O(k + \log m + \log n)$ we get security $\delta = 2^{-k}$ and share size

$$m + O(\lambda \log^2 n + \lambda^2 \log n) = m + O(k^2 \cdot \mathsf{polylog}(n+m)) = m + \widetilde{O}(k^2).$$

We show how to improve the above to $m + \widetilde{O}(k)$ in the next section.

5.2 Improved Parameters via Parallel Repetition

We saw how to achieve robust secret sharing with security $\delta = 2^{-k}$ at the cost of having a share size $m + \widetilde{O}(k^2)$. We now show how to improve this. We do so by instantiating the scheme from the previous section with smaller parameters that only provide weak robustness, say $\delta = \frac{1}{4}$. We then use parallel repetition of $q = \theta(k)$ independent copies of this weak scheme. The q copies of the recovery procedure recover q candidate messages, and we simply output the majority vote. A naive implementation of this idea, using q completely independent copies of the scheme, would result in share size $O(km) + \widetilde{O}(k)$ since we would have $q = \Theta(k)$ copies of the non-robust (e.g., Shamir) share which is as large as the message size m. However, we notice that we can reuse the same shares of the non-robust secret sharing scheme $(\widetilde{s}_1, \ldots, \widetilde{s}_n) \leftarrow \mathsf{Share}_{nr}(\mathsf{msg})$ across all q copies. This is because the robustness security held even for a worst-case choice of such shares, only over the randomness of the other components. Therefore, we only get a share size of $m + \widetilde{O}(k)$.

Construction. In more detail, let $(\mathsf{Share}, \mathsf{Rec})$ be our robust secret sharing scheme construction from above. For some random coins coins_{nr} of the non-robust secret sharing scheme, we let $\mathsf{Share}_{\mathsf{coins}_{nr}}(\mathsf{msg})$ denote the execution of the sharing procedure $\mathsf{Share}(\mathsf{msg})$ where step S.1 uses the randomness coins_{nr} to select the non-robust shares $(\widetilde{s}_1, \ldots, \widetilde{s}_n) \leftarrow \mathsf{Share}_{nr}(\mathsf{msg}; \mathsf{coins}_{nr})$ but steps S.2 – S.5 use fresh randomness. In particular, $\mathsf{Share}_{\mathsf{coins}_{nr}}(\mathsf{msg})$ remains a randomized algorithm.

We define the q-wise parallel repetition scheme $(\mathsf{Share}', \mathsf{Rec}')$ as follows:

$\mathsf{Share}'(\mathsf{msg})$**:** The sharing procedure proceeds as follows
- Choose uniformly random coins coins_{nr}.
- For $j = 1, \ldots, q$: sample $(s_1^j, \ldots, s_n^j) \leftarrow \mathsf{Share}_{\mathsf{coins}_{nr}}(\mathsf{msg})$ with $s_i^j = (\widetilde{s}_i, E_i^j, \mathsf{key}_i^j, r_i^j, p_i^j)$, where $\widetilde{s}_i$ is the same in all q iterations. Let $\hat{s}_i^j = (E_i^j, \mathsf{key}_i^j, r_i^j, p_i^j)$ be the fresh components.
- For $i \in [n]$, define $s_i = (\widetilde{s}_i, \hat{s}_i^1, \ldots, \hat{s}_i^q)$ and outputs $(s_1, \ldots, s_n)$.

$\mathsf{Rec}'(s_1, \ldots, s_n)$**:** The reconstruction procedure proceeds as follows
- For $i \in [n]$, parse $s_i = (\widetilde{s}_i, \hat{s}_i^1, \ldots, \hat{s}_i^q)$. For $j \in [q]$, define $s_i^j := (\widetilde{s}_i, \hat{s}_i^j)$.
- For $j \in [q]$, let $\mathsf{msg}_j := \mathsf{Rec}(s_1^j, \ldots, s_n^j)$. If there is a majority value msg such that $|\{j \in [q] \ : \ \mathsf{msg} = \mathsf{msg}_j\}| > q/2$ then output msg, else output $\perp$.

Theorem 7. *Assume that the parameters of* $(\mathsf{Share}, \mathsf{Rec})$ *are chosen such that the scheme is δ-robust for $\delta \leq \frac{1}{4}$. Then the q-wise parallel repetition scheme* $(\mathsf{Share}', \mathsf{Rec}')$ *is a δ'-robust secret sharing scheme with $\delta' = e^{-\frac{3}{128}q}$.*

We prove theorem 7 in appendix B.4.

Parameters. We choose the parameters of the underlying scheme $(\mathsf{Share}, \mathsf{Rec})$ to have security $\delta = \frac{1}{4}$. This corresponds to choosing a sufficiently large $\lambda = O(\log m + \log n)$. The overhead on the share size (ingnoring the m bit share of the non-robust scheme) is then $\mathsf{polylog}(n+m)$. By choosing a sufficiently large $q = O(k)$ and setting $(\mathsf{Share}', \mathsf{Rec}')$ to be the q-wise parallel repetition, we therefore get a scheme with robustness $\delta' = 2^{-k}$ and share size $m + O(k\mathsf{polylog}(n+m)) = m + \widetilde{O}(k)$.

6 Inefficient Graph Identification via Self-Consistency

We begin by showing a simple inefficient algorithm for the graph identification problem. In particular, we show that with overwhelming probability the set $H \cup P$ is the unique largest *self-consistent* set of vertices, meaning that there are no bad edges between vertices in the set.

Definition 9 (Self-Consistency). *Let $G = (V, E)$ be a directed graph and let $L : V \to \{\mathsf{good}, \mathsf{bad}\}$ be a labeling. We say that a subset of vertices $S \subseteq V$ is* self-consistent *if for all $e \in E_{S \to S}$ we have $L(e) = \mathsf{good}$. A subset $S \subseteq V$ is* max self-consistent *if $|S| \geq |S'|$ for every self-consistent $S' \subseteq V$.*

Note that there may not be a unique max self-consistent set in G. However, the next therorem shows that if the components are sampled as in the graph challenge game $\mathsf{Gen}^{\mathsf{Adv}}(n, t, d)$, then with overwhelming probability there is a unique max self-consistent set in G and it is $H \cup P$.

Lemma 4. *For any Adv, and for the distribution $(G, L, H, A, P) \leftarrow \mathsf{Gen}^{\mathsf{Adv}}(n, t, d)$, the set $H \cup P$ is the unique max self-consistent set in G with probability at least $1 - 2^{-\Omega(d - \log n)}$.*

Proof. We know that the set $H \cup P$ is self-consistent by the definition of the graph challenge game. Assume that it is not the unique max self-consistent set in G, which we denote by the event BAD. Then there exists some set $S \neq H \cup P$ of size $|S| = |H \cup P|$ such that S is self consistent. This means that S must contain at least $q \geq 1$ elements from A and at least $t+1-q$ elements from H. In other words there exists some value $q \in \{1, \ldots, t\}$ and some subsets $A' \subseteq S \cap A \subseteq A \subseteq A \cup P$ of size $|A'| = q$ and $H' \subseteq S \cap H \subseteq H$ of size $t + 1 - q$ such that $E_{H' \to A'} = \varnothing$. This is because, by the definition of the graph challenge game, every edge in $E_{H' \to A'} \subseteq E_{H \to A}$ is labeled bad and so it must be empty if S is consistent. For any fixed q, H', A' as above, if we take the probability over the random choice of d outgoing edges for each $v \in H'$, we get:

$$\Pr[E_{H' \to A'} = \varnothing] = \left(\frac{\binom{n-1-q}{d}}{\binom{n-1}{d}} \right)^{t+1-q} \leq \left(1 - \frac{q}{n-1}\right)^{d(t+1-q)} \leq e^{-\frac{d(t+1-q)q}{n}}.$$

By taking a union bound, we get

$$\Pr[\mathsf{BAD}] \leq \Pr\left[\exists \left\{ \begin{array}{c} q \in \{1,\ldots,t\} \\ A' \subseteq A \cup P : |A|' = q \\ H' \subseteq H, |H'| = t+1-q \end{array} \right\} \ : \ E_{H' \to A'} = \varnothing \right]$$

$$\leq \sum_{q=1}^{t} \binom{t+1}{t+1-q} \cdot \binom{t}{q} \cdot e^{-\frac{d(t+1-q)q}{n}} \leq \sum_{q=1}^{t} \binom{t+1}{t+1-q} \cdot \binom{t+1}{q} \cdot e^{-\frac{d(t+1-q)q}{n}}$$

$$\leq 2 \sum_{q=1}^{(t+1)/2} \binom{t+1}{q}^2 \cdot e^{-\frac{d(t+1-q)q}{n}}$$

(symmetry between q and $t+1-q$)

$$\leq 2 \sum_{q=1}^{(t+1)/2} (t+1)^{2q} \cdot e^{-\frac{d(t+1-q)q}{n}}$$

$$\leq 2 \sum_{q=1}^{(t+1)/2} e^{q\left(2\log_e(t+1) - \frac{d(t+1-q)}{n}\right)}$$

$$\leq 2 \sum_{q=1}^{(t+1)/2} e^{q\left(2\log_e(t+1) - \frac{(t+1)d}{2n}\right)} \qquad \text{(since } q \leq (t+1)/2\text{)}$$

$$\leq 2 \sum_{q=1}^{(t+1)/2} e^{q(2\log_e(t+1) - d/4)} \qquad \text{(since } t+1 > n/2\text{)}$$

$$\leq (t+1)e^{(2\log_e(t+1) - d/4)} \leq 2^{-\Omega(d - \log n)}$$

As a corollary of the above lemma, we get an inefficient algorithm for the graph identification problem, that simply tries every subset of vertices $S \subseteq V$, checks if S is self-consistent, and outputs the max consistent set.

Corollary 1. *There exists an inefficient algorithm* $\mathsf{GraphID}^{\mathsf{ineff}}$ *such that for any* Adv*, and any* t*,* $n = 2t+1$*, and any* d *we have*

$$\Pr[B = H \cup P \ : \ B \leftarrow \mathsf{GraphID}^{\mathsf{ineff}}(G, L), (G, L, H, A, P) \leftarrow \mathsf{Gen}^{\mathsf{Adv}}(n, t, d)] \geq 1 - 2^{-\Omega(d - \log n)}$$

7 Efficient Graph Identification

In this section, we prove Theorem 6 and given an efficient graph identification algorithm. We begin with an intuitive overview before giving the technical details.

7.1 Overview and Intuition

A Simpler Problem. We will reduce the problem of identifying the full set $H \cup P$ to the problem of only identifying a small set $Y \subseteq H \cup P$ such that

$Y \cap H$ is of size at least εn for $\varepsilon = 1/\Theta(\log n)$. If we are given such a Y, we can consider a larger set S defined as all vertices in $[n]$ with no bad incoming edge originating in Y. We observe that every vertex in $H \cup P$ is included in S, as there are no bad edges from $H \cup P$ to $H \cup P$. Since $Y \cap H$ is big enough, it is unlikely that a vertex in A could be included in S, as every vertex in A likely has an incoming edge from $|Y \cap H|$ that is labeled as bad. There is a bit of subtlety involved in applying this intuition, as it is potentially complicated by dependencies between the formation of the set Y and the distribution of the edges from Y to A. We avoid dealing with such dependencies by "splitting" the graph into multiple independent graphs and building Y from one of these graphs while constructing S in another. Ultimately, we obtain that with high probability, we will have $S = H \cup P$.

Now the task becomes obtaining such a set Y in the first place. We consider two cases depending on whether the set P is small ($|P| \leq \varepsilon n$) or large ($|P| > \varepsilon n$).

Small P. In this case, there is only a small number of good edges crossing between H and $A \cup P$ (only edges between H and P). Therefore there exists a bisection of the graph into sets H and $A \cup P$ of size $t+1$ and t respectively, where the number of good edges crossing this bisection is approximately εdn. By using an efficient $O(\log n)$-approximation algorithm for the graph bisection problem (on the good edges in G) we can get a bisection X_1, X_2 with only (say) $.1nd$ edges crossing between X_1 and X_2. This means that, with overwhelming probability, one of X_1 or X_2 has (say) .9 fraction of the honest vertices, as otherwise we'd expect more edges crossing this partition. We can then refine such an X to get a suitable smaller subset Y which is fully contained in $H \cup P$ similarly as above.

Large P. In this case, the intuition is that every active vertex is likely to have at least $d/2$ in-coming bad edges (from the honest vertices), but honest/passive vertices will have only at most $d(1/2 - \varepsilon)$ in-coming bad edges on average from the active vertices. So we can differentiate the two cases just by counting. This isn't precise since many active vertices can point bad edges at a single honest vertex to make it "look bad". However, intuitively, this cannot happen too often.

To make this work, we first start with the full set of vertices $[n]$ and disqualify any vertices that have more than $d/2$ out-going bad edges (all honest vertices remain since they only have $d(1/2-\varepsilon)$ outgoing bad edges on expectation). This potentially eliminates some active vertices. Let's call the remaining smaller set of vertices X. We then further refine X into a subset Y of vertices that do not have too many incoming bad edges (more than $d/2-\varepsilon/2$) originating in X. The active vertices are likely to all get kicked out in this step since we expect $d(1/2 - \varepsilon)$ incoming bad edges. On the other hand, we claim that not too many honest vertices get kicked out. The adversary has at most $(1/2 - \varepsilon)nd/2$ out-going bad edges in total under his control in the set $X \cap A$ and has to spend $d/2 - \varepsilon/2$ edges to kick out any honest party . Therefore there is a set of at least $\varepsilon n/2$ of the honest parties that must survive. This means that $Y \subseteq H \cup P$ and that Y contains $\Theta(n/\log n)$ honest parties as we wanted.

Unknown P. Of course, our reconstruction procedure will not know a priori whether P is relatively large or small or, in the case that P is small, which one of the bisection sets X_1 or X_2 to use. So it simply tries all of these possibilities and obtains three candidate sets Y_0, Y_1, Y_2, one of which has the properties we need but we do not know which one. To address this, we construct the corresponding sets S_i for each Y_i as described above, and we know that one of these sets S_i is $H \cup P$. From the previous section (Lemma 4), we also know that $H \cup P$ is the unique max self-consistent set in G. Therefore, we can simply output the largest one of the sets S_0, S_1, S_2 which is self-consistent in G and we are guaranteed that this is $H \cup P$.

7.2 Graph Splitting

Here, we describe a procedure GraphSplit that takes as input a directed graph $G = (V, E)$ with out-degree d for each vertex and outputs three directed graphs $(G^i = (V, E^i))_{i=1,2,3}$ such that $E^i \subset E$ and the out-degree of each vertex in each graph is $d/3$. Furthermore for vertex v in G whose d-outgoing edges $E_{v \to V}$ were uniformly random in G, the three sets $E^i_{v \to V}$ are distributed like 3 independently sampled sets of $d' = d/3$ outgoing edges each.

The procedure GraphSplit is defined as follows:

1. On input a labeled directed graph $G = (V, E, L)$ with d outgoing edges for each vertex, for each $v \in V$:
   (a) define $V_v := \{w \in V \mid (v, w) \in E\}$, the set of neighbors of v
   (b) sample three uniform and independent subsets $\{S^i_v\}_{i=1,2,3}$ of $V \setminus \{v\}$ with the constraint $|S^i_v| = d/3$.
   (c) sample a uniformly random injective function $\pi_v : \cup_{i=1,2,3} S^i_v \to V_v$
2. define $E^i := \{(v, w) \in E \mid w \in \pi_v(S^i_v)\}$,
3. for all $e \in E^i$, define $L^i(e) := L(e)$.

Lemma 5. *Let $G = (V, E)$ be a distribution over graphs with out-degree $d = 3d'$. Let $(G^i = (V, E^i))_{i=1,2,3} \leftarrow$ GraphSplit(G). For each $v \in V$ such that the distribution of*

$$V_v := \{w \in V \mid (v, w) \in E\}$$

is a uniform subset over $V \setminus \{v\}$ of size d and independent of $V_{v'}$ for all $v' \neq v$, we have that for each $i = 1, 2, 3$:

$$\left(V^i_v := \{w \in V \mid (v, w) \in E^i\}\right)$$

is a uniform subset of $V \setminus \{v\}$ of size d' and independent of $V^{i'}_{v'}$ for any $(v', i') \neq (v, i)$.

Proof. Let R_i $(i = 1, 2, 3)$ be an arbitrary subset of $V \setminus \{v\}$ of size $d/3$. We want to study the probability that $V_v^i = R_i$ for all i. We proceed as follows:

$$\begin{aligned}
\Pr[\forall i : V_v^i = R_i] &= \Pr[\forall i : V_v^i = R_i \mid \forall i : R_i \subset V_v] \cdot \Pr[\forall i : R_i \subset V_v] + \\
&\quad + \Pr[\forall i : V_v^i = R_i \mid \forall i : R_i \not\subset V_v] \cdot \Pr[\forall i : R_i \not\subset V_v] \\
&\qquad \text{(2nd summand equals zero)} \\
&= \Pr[\forall i : V_v^i = R_i \mid \forall i : R_i \subset V_v] \cdot \Pr[\forall i : R_i \subset V_v] \\
&= \Pr[\forall i : \pi_v(S_v^i) = R_i \mid \forall i : R_i \subset V_v] \cdot \Pr[\forall i : R_i \subset V_v] \\
&\qquad (\pi_v \text{ bijective}) \\
&= \Pr[\forall i : S_v^i = \pi_v^{-1}(R_i) \mid \forall i : R_i \subset V_v] \cdot \Pr[\forall i : R_i \subset V_v]
\end{aligned}$$

Now, we use the fact that the S_i's are independent, and that, given V_v, S_v^i is a uniform subset of $\pi_v^{-1}(V_v)$ of size $d/3$. Therefore, we have:

$$\begin{aligned}
\Pr[\forall i : V_v^i = R_i] &= \binom{d/3}{d}^3 \cdot \Pr[\forall i : R_i \subset V_v] \\
&= \binom{d/3}{d}^3 \cdot \binom{d}{n-1}^3 \\
&= \binom{d/3}{n-1}^3
\end{aligned}$$

This shows that the V_v^i's constitute a tuple of uniform and independent subsets of $V \setminus \{v\}$ of size $d/3$, which concludes the proof of lemma 5

7.3 Our Algorithm

We now define the efficient graph identification algorithm $B \leftarrow \mathsf{GraphID}(G, L)$.

Usage. Our procedure $\mathsf{GraphID}(G, L)$ first runs an initialization phase ***Initialize***, in which some parameters are generated, and the graph edges are partitioned into three sets. After that, it runs two procedures ***Small*** P and ***Large*** P sequentially, and uses the data these two procedures return to run the output phase **Output**.

Initialize.

1. Let b be a constant such that there exists a polynomial-time $b \log n$-approximate graph bisection algorithm Bisect, such as the one provided in [Räc08]. Let $c = \frac{800}{9} b$, and let $\varepsilon = 1/(c \cdot \log(n))$.
2. Run $(G^1, G^2, G^3) \leftarrow \mathsf{GraphSplit}(G)$. This produces three graphs having V as set of vertices, with the property that the out-degree of every vertex in each graph is $d' = d/3$, and the neighbors of honest nodes in each graph are distributed independently and uniformly at random (see section 7.2 for more details).

Small P**.**

1. Run $(X_0, X_1) \leftarrow \mathsf{Bisect}(G^*)$, where G^* is the undirected graph induced by the good edges of G^1,

2. For $i = 0, 1$: contract X_i to a set of *candidate good* vertices Y_i whose vertices have fewer than $0.4d'$ incoming bad edges in the graph G^2.

$$Y_i := \{v \in X_i \mid |\{e \in E^2_{X_i \to \{v\}} \mid L(e) = \mathsf{bad}\}| < 0.4d'\}.$$

3. For $i = 0, 1$: return Y_i.

Large P.

1. Define a set of *candidate legal* vertices X_2 as the set of vertices having fewer than $d'/2$ outgoing bad edges in G^1.

$$X_2 := \{v \in V \mid |\{e \in E^1_{\{v\} \to V} \mid L(e) = \mathsf{bad}\}| < d'/2\}.$$

2. Contract X_2 to a set of *candidate good* vertices Y_2, defined as the set of vertices in X_2 having fewer than $d'\,(1/2 - \varepsilon/2)$ incoming bad edges from legal vertices in the graph G^1.

$$Y_2 := \{v \in X_2 \mid |\{e \in E^1_{X_2 \to \{v\}} \mid L(e) = \mathsf{bad}\}| < d'\,(1/2 - \varepsilon/2)\}.$$

3. Return Y_2.

Output. This subprocedure takes as input the sets Y_0, Y_1 (generated by **Small** P), and Y_2 (generated by **Large** P) and outputs a single set B, according to the following algorithm.

1. For $i = 0, 1, 2$: define S_i as the set of vertices that only have incoming good edges from Y_i in G^3. Formally,

$$S_i := \{v \in V \mid \forall e \in E^3_{Y_i \to v} : L(e) = \mathsf{good}\}$$

2. For $i = 0, 1, 2$: if $L(e) = \mathsf{good}$ for all $e \in E_{S_i \to S_i}$, define $B_i := S_i$; otherwise, define $B_i = \varnothing$. This is a self-consistency check that rejects S_i if the subgraph induced by S_i contains at least one bad edge.
3. Output a set B defined as any of the largest sets among B_0, B_1, B_2.

7.4 Analysis of Correctness

In this section, we establish Theorem 6. We first fix an arbitrary adversary Adv and then define a *sufficient event*:

$$O := \text{“there exists } i \in \{0, 1, 2\} \text{ such that } |Y_i \cap H| \geq \varepsilon \cdot n/2 \text{ and } Y_i \subseteq H \cup P\text{”}$$

Our analysis is summarized in figure 1.

The three subsections below establish the following three lemmas, respectively.

Lemma 6. *The conditional probability that set B output by* GraphID *equals $H \cup P$, given the occurrence of event O, is $1 - 2^{-\Omega(d/\log n)}$.*

Lemma 7. *If $|P|$ is at most $\varepsilon \cdot n$, then the probability that O occurs is at least $1 - 2^{-\Omega(d/\log n)}$.*

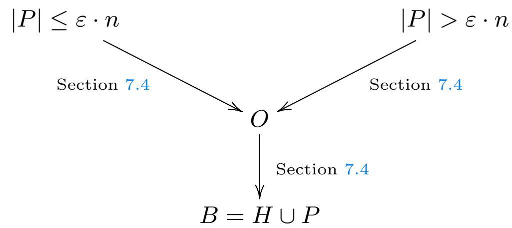

Fig. 1. Structure of our analysis: arrows denote logical implications (happening with high probability).

Lemma 8. *If $|P|$ is greater than $\varepsilon \cdot n$, then the probability that O occurs is at least* $1 - 2^{-\Omega(d/\log^2 n)}$.

In Lemma 6, the probability is over the random edge selection in G^3, while in Lemmas 7 and 8, the probability is over the random edge selections in G^1 and G^2. Note that the set P is selected by the adversary arbitrarily prior to the construction of the graph G. We now complete the proof of Theorem 6.

Proof of Theorem 6. By Lemmas 7 and 8, we obtain that the event O occurs with probability at least $1 - 2^{-\Omega(d/\log^2 n)}$. Putting together with Lemma 6, we obtain that the probability that the set B returned by the algorithm equals $H \cup P$ is at least $1 - 2^{-\Omega(d/\log^2 n)}$ completing the proof of the theorem.

Given the Sufficient Event O, $H \cup P$ is recovered

In this section, we prove Lemma 6. Assume that event O occurs. That is, there exists $i \in \{0, 1, 2\}$ such that $|Y_i \cap H| \geq \varepsilon \cdot n/2$ and $Y_i \subseteq H \cup P$.

First we show that $S_i = H \cup P$ with high probability, and later show that $B = S_i$. Note that the event O only depends on G^1, G^2 but is independent of the edges $E^3_{H \to V}$. All probabilities below are only taken over these edges.

Since $Y_i \subseteq H \cup P$, we have that $S_i \supseteq H \cup P$, because $L(e) = \mathsf{good}$ for each $e \in E_{H \cup P \to H \cup P}$. This means that we just need to prove that $A \cap S_i = \varnothing$ with high probability. We have

$$\begin{aligned}
\Pr[A \cap S_i \neq \varnothing] &\leq \sum_{v \in A} \Pr[v \in S_i] && \text{(definition of } S_i\text{)} \\
&= \sum_{v \in A} \Pr[\forall e \in E^3_{Y_i \to v} : L(e) = \mathsf{good}] && \text{(relaxing a constraint)} \\
&\leq \sum_{v \in A} \Pr[\forall e \in E^3_{Y_i \cap H \to v} : L(e) = \mathsf{good}] \\
& && (L(e) = \mathsf{bad} \text{ for } e \in E_{H \to A}) \\
&= \sum_{v \in A} \Pr[E^3_{Y_i \cap H \to v} = \varnothing] && \text{(uniformity of } E^3_{Y_i \cap H \to v}\text{)} \\
&\leq \sum_{v \in A} \left(1 - \frac{d'}{n}\right)^{|Y_i \cap H|} && \text{(by assumption on } O\text{)} \\
&\leq t \cdot \left(1 - \frac{d'}{n}\right)^{\varepsilon n/2} \leq t \cdot e^{-\varepsilon \cdot d'/2} = 2^{-\Omega(\varepsilon d)}
\end{aligned}$$

Now, we need to show that no set S_j ($j \neq i$) is larger than $H \cup P$: this immediately follows from lemma 4, which guarantees that $H \cup P$ is the largest self-consistent subset of vertices with probability $1 - 2^{-\Omega(d - \log n)}$. We thus have that conditioned on O, the probability that B is $H \cup P$ is at least $1 - 2^{-\Omega(d/\log n)}$.

Reaching the Sufficient Event O, if $|P| \leq \varepsilon \cdot n$

Assume that $|P| \leq \varepsilon \cdot n$. We prove Lemma 7 by combining three claims.

Claim. The probability that either $|X_0 \cap H| \geq 9|H|/10$ or $|X_1 \cap H| \geq 9|H|/10$ is at least $1 - e^{-\Omega(\varepsilon d)}$.

Proof. Let's first analyze the number of (non-directed) cross edges $|E^*_{P \cup A, H}|$ of the partition $(P \cup A, H)$ for G^*. We claim that with probability $1 - 2^{-\Omega(\varepsilon d)}$, we have

$$|E^*_{P \cup A, H}| = |E^*_{P,H}| = |E^*_{P \to H}| + |E^*_{H \to P}| \leq \varepsilon n d' + 2\varepsilon n d' = 3\varepsilon n d'.$$

In the above derivation, the first equality holds because there are no good edges from A to H (and vice versa), and the inequality follows from the facts that any vertex in P has at most d outgoing edges to H, and the number of outgoing edges from H to P is at most $2\varepsilon n d'$ with probability $1 - 2^{-\Omega(\varepsilon d)}$ by applying lemma 9 (with $H' = H$, $V' = P$, and δ such that $(1 + \delta)|H||P|d'/(n-1) = 2\varepsilon n d'$).

Since Bisect is $b \log n$-approximate for some constant b, we obtain that with probability $1 - 2^{-\Omega(\varepsilon d)}$

$$|E^*_{X_0, X_1}| \leq b \cdot \log n \cdot 3\varepsilon \cdot n \cdot d' = \frac{3b \log n}{c \log n} \cdot n \cdot d' \leq \frac{9}{800} \cdot n \cdot d', \tag{7.1}$$

where we use the fact that c is at least $\frac{800}{9} b$.

Let L_i denote the event that $|X_i \cap H|$ is less than $9|H|/10$. We will show that $\Pr[L_1 \text{ and } L_2]$ is at most $e^{-\Omega(d)}$. We first consider the event that L_1 holds, L_2 holds, and $|E^*_{X_0,X_1}| < 9nd'/800$. For this event to happen, it must be the case that there exists a set S of honest vertices of size s in $(|H|/10, |H|/2]$ such that the number of edges (all of which are labeled good) between S and $H - S$ is less than $9nd'/800$. The expected number of edges between S and $H - S$ is $s \cdot (t + 1 - s) \cdot d'/(n - 1)$, which is at least $9(t + 1)^2 d'/(100(n - 1)) \geq 9nd'/400$. For a given set S of size s, invoking lemma 9 with $H' = S$, $V' = H - S$, and $\delta = 1/2$, it follows that

$$\begin{aligned}\Pr\left[|E^*_{S,H-S}| < 9nd'/800\right] &\leq \Pr\left[|E^*_{S,H-S}| < s \cdot (t + 1 - s) \cdot d'/(2(n - 1))\right]\\ &\leq e^{-s(t+1-s)d'/(8(n-1))}.\end{aligned}$$

By taking a union bound, we get

$$\begin{aligned}&\Pr\left[L_1 \text{ and } L_2 \text{ and } |E^*_{X_0,X_1}| < 9nd'/800\right]\\ &\leq \Pr\left[\exists \left\{\begin{array}{c} s \in (|H|/10, |H|/2) \\ S \subseteq H : |S| = s \end{array}\right\} : E^*_{S,H-S} < 9nd'/800\right]\\ &\leq \sum_{s=\lceil (t+1)/10 \rceil}^{\lfloor (t+1)/2 \rfloor} \binom{t+1}{s} e^{-s(t+1-s)d'/8(n-1)} \leq \sum_{s=\lceil (t+1)/10 \rceil}^{\lfloor (t+1)/2 \rfloor} t^s \cdot e^{-\Omega(sd')} \quad (t \geq n/2)\\ &\leq \sum_{s=\lceil (t+1)/10 \rceil}^{\lfloor (t+1)/2 \rfloor} e^{s \ln t - \Omega(sd')} \leq e^{-\Omega(t)}. \qquad (d \text{ is } \Omega(\log n))\end{aligned}$$

We are now ready to show that $\Pr[L_1 \text{ and } L_2]$ is at most $2^{-\Omega(\varepsilon d)}$.

$$\begin{aligned}\Pr[L_1 \text{ and } L_2] &\leq \Pr\left[|E^*_{X_0,X_1}| \geq 9nd'/800\right] + \Pr\left[L_1 \text{ and } L_2 \text{ and } |E^*_{X_0,X_1}| < 9nd'/800\right]\\ &\leq e^{-\Omega(\varepsilon d)} + e^{-\Omega(t)} \qquad \text{(Equation 7.1)}\\ &= 2^{-\Omega(\varepsilon d)}.\end{aligned}$$

Claim. If $|X_i \cap H| \geq 9|H|/10$, then $|Y_i \cap H| \geq \varepsilon|H|$.

Proof. The set Y_i consists of all vertices in X_i that have fewer than $0.4d'$ bad edges from X_i. The number of active vertices in X_i is at most $t+1-9(t+1)/10 = (t+1)/10$. Therefore, the total number of bad edges into vertices in $X_i \cap H$ is at most $d'(t + 1)/10$. By an averaging argument, the number of vertices in $X_i \cap H$ that have at least $0.4d'$ bad edges is at most $d'(t + 1)/(0.4d' \cdot 10) \leq (t + 1)/4$. This implies that $Y_i \cap H$ has at least $(t + 1)(9/10 - 1/4) \geq \varepsilon|H|$.

Claim. If $|X_i \cap H| \geq 9|H|/10$, then $Y_i \cap A = \varnothing$ with probability at least $1-2^{-\Omega(d)}$.

Proof. Consider an active vertex v in X_i. By invoking Lemma 9 with $H' = X_i \cap H$, $V' = \{v\}$ and $\delta = 1/9$, the number of edges from $X_i \cap H$ into v is less than $0.4d'$ with probability at most $2^{-\Omega(d)}$. Taking a union bound over all active vertices in $X_i \cap H$ and noting that d is $\Omega(\log n)$, we obtain that the probability that any active vertex is in Y_i is at most $2^{-\Omega(d)}$, yielding the desired claim.

By Claims 7.4, 7.4, and 7.4, we obtain that when $|P| \le \varepsilon n$,

$$\Pr\left[\exists i : |Y_i \cap H| \ge \varepsilon n/2 \text{ and } Y_i \subseteq H \cup P\right] \ge 1 - 2^{-\Omega(\varepsilon d)},$$

which proves Lemma 7.

Reaching the Sufficient Event O, if $|P| > \varepsilon \cdot n$

Assume that $|P| > \varepsilon \cdot n$. We prove Lemma 8 by combining the following three claims. All probabilities in this analysis are only over the random choice of the edges $E^1_{H \to V}$.

Claim. $\Pr[H \subseteq X_2] \ge 1 - 2^{-\Omega(\varepsilon^2 d + \log n)}$

Proof. If it does not hold that $H \subseteq X_2$ then there exists some $u \in H$ such that u has $\ge d'/2$ outgoing bad edges in E^1, which means that $|E^1_{u \to A}| > d'/2$. By applying Lemma 9 with $H' = \{u\}$, $|V'| = |A|$ of size $|V'| \le t - \varepsilon n \le (\frac{1}{2} - \varepsilon)n$, and $\delta = (n-1)/(2|V'|) - 1$ so that $(1+\delta)|H'||V'|\frac{d'}{n-1} = d'/2$ we get

$$\begin{aligned}\Pr\left[|E^1_{u \to A}| > d'/2\right] &\le \Pr\left[|E^1_{u \to A}| > (1+\delta)|H'||V'|\frac{d'}{n-1}\right] \\ &\le e^{-\delta^2 |V'| \frac{d'}{3(n-1)}} \le e^{-\Omega(\varepsilon^2 d)}\end{aligned}$$

where we rely on the fact that $\delta > (2\varepsilon n - 1)/(2|V'|)$ to do the last calculation. Finally, by taking a union bound, we get

$$\Pr[\neg(H \subseteq X_2)] \le \sum_{u \in H} \Pr\left[|E^1_{u \to A}| > d'/2\right] \le e^{-\Omega(\varepsilon^2 d + \log n)}$$

which proves the claim.

Claim. Whenever $H \subseteq X_2$ occurs then $|Y_2 \cap H| \ge \varepsilon n/2$.

Proof. Let us assume $H \subseteq X_2$ occurs. Define the set $K = H \setminus Y_2$ to be the honest vertices that were "killed off" in the contraction from X_2 to Y_2. Define E^1_{BAD} to be the set of edges $e \in E^1$ with $L(e) = \mathsf{bad}$.

Every vertex in K must have $d'(1/2 - \varepsilon/2)$ incoming bad edges from X_2. Furthermore, since $K \subseteq H$, the only incoming bad edges can from A and therefore it must have $d(1/2 - \varepsilon/2)$ incoming bad edges to K can from $A \cap X_2$. Therefore for each $k \in K$, we have $E^1_{A \cap X_2 \to k} \cap E^1_{BAD} \ge d'(1/2 - \varepsilon/2)$ and $E^1_{A \cap X_2 \to K} \cap E^1_{BAD} \ge d'(1/2 - \varepsilon/2)|K|$. However, since the way that we define X_2 ensures that each vertex in $A \cap X_2$ has only $d'/2$ outgoing bad edges total, we have $E^1_{A \cap X_2 \to K} \cap E^1_{BAD} \le |A \cap X_2| d'/2 \le |A| d'/2$. Putting this together we get:

$$d'(1/2 - \varepsilon/2)|K| \le E^1_{A \cap X_2 \to K} \cap E^1_{BAD} \le |A| d'/2$$

and therefore $|K| \le |A|/(1-\varepsilon) \le (1/2 - \varepsilon)n/(1-\varepsilon) \le (1/2 - \varepsilon/2)n$.

Therefore $|Y_2 \cap H| \ge |H| - |K| \ge n/2 - (1/2 - \varepsilon/2)n \ge \varepsilon n/2$.

Claim. $\Pr[Y_2 \cap A \neq \varnothing] \leq 2^{-\Omega(\varepsilon^2 d + \log n)}$

Proof. Firstly, note that if the following two events occur:

1. $H \subseteq X_2$
2. for every $v \in A$ we have $|E^1_{H\to v}| \geq d'(1/2 - \varepsilon/2)$

then $Y_2 \cap A = \varnothing$. This is because, together, the above events imply that for every $v \in A$ $|E^1_{H\cap X_2 \to v}| \geq d'(1/2-\varepsilon/2)$, and by definition all of these edges are labeled bad. By construction, this means that $v \notin Y_2$. Therefore

$$\begin{aligned}\Pr[Y_2 \cap A \neq \varnothing] &\leq \Pr[\neg(H \subseteq X_2)] + \Pr[\exists v \in A \;:\; |E^1_{H\to v}| < d'(1/2-\varepsilon/2)] \\ &\leq 2^{-\Omega(\varepsilon^2 d + \log n)} + \sum_{v\in A} \Pr[|E^1_{H\to v}| < d'(1/2-\varepsilon/2)]\end{aligned}$$

where the first summand comes from Claim 7.4. To bound the second summand we rely on Lemma 9 with $H' = H$ and $V' = \{v\}$, where we set $\delta = 1 - (1/2 - \varepsilon/2)(n-1)/|H|$ so that $(1-\delta)|H|d'/(n-1) = d'(1/2-\varepsilon/2)$. We get:

$$\Pr[|E^1_{H\to v}| < d'(1/2-\varepsilon/2)] \leq e^{-\delta^2 |H| d'/(2n-1)} \leq 2^{-\Omega(\varepsilon^2 d)}$$

where we use the fact that $\delta > \varepsilon$ in the last step. Combining the above, we get the claim.

By combining Claims 7.4, 7.4, 7.4 we get that, when $|P| > \varepsilon \cdot n$, then

$$\Pr[|Y_2 \cap H| \geq \varepsilon n/2 \text{ and } Y_i \subseteq H \cup P] \geq 1 - 2^{-\Omega(\varepsilon^2 d + \log n)}$$

which proves Lemma 8.

A Useful Chernoff Type Bound on Edges

Several of the preceding lemmas make use of the following Chernoff type bounds on the number of edges crossing a cut of the graph.

Lemma 9. *Let $H' \subseteq H$ and $V' \subseteq V$ be arbitrary sets of vertices. Then for any $i \in \{1,2,3\}$ and for any $\delta > 0$ we have:*

$$\Pr\left[\; |E^i_{H'\to V'}| \geq (1+\delta)|H'||V'|\frac{d'}{n-1} \;\right] \leq e^{-\delta^2 |H'||V'| \frac{d'}{3(n-1)}}$$
$$\Pr\left[\; |E^i_{H'\to V'}| \leq (1-\delta)|H'|(|V'|-1)\frac{d'}{n-1} \;\right] \leq e^{-\delta^2 |H'|(|V'|-1) \frac{d'}{2(n-1)}}$$

Furthermore if $H' \cap V' = \varnothing$ then

$$\Pr\left[\; |E^i_{H'\to V'}| \leq (1-\delta)|H'||V'|\frac{d'}{n-1} \;\right] \leq e^{-\delta^2 |H'||V'| \frac{d'}{2(n-1)}}$$

where the probability is only over the choice of the edges $E^i_{H\to V}$ and independent of E^j for $j \neq i$.

Proof. For $u \in H', v \in V'$ define

$$\Delta_{u,v} = \begin{cases} 1 \text{ if } (u,v) \in E^i \\ 0 \text{ else} \end{cases}$$

Define $\Delta := |E^i_{H' \to V'}| = \sum_{u \in H', v \in V'} \Delta_{u,v}$. Since the expected value $\mathbb{E}[\Delta_{u,v}] = \frac{d}{n-1}$ for any $u \neq v$, and $\mathbb{E}[\Delta_{u,u}] = 0$ we have $\mathbb{E}[\Delta] = \sum_{u \in H', v \in V'} \mathbb{E}[\Delta_{u,v}] \in [|H'|(|V'|-1)\frac{d}{n-1}, |H'||V'|\frac{d}{n-1}]$. Furthermore, it's easy to check that the variables $\Delta_{u,v}$ are negatively correlated due to sampling without replacement (Definition 1) and therefore the lemma follows from the Chernoff bounds (Theorem 1).

8 Conclusion

We constructed an efficient robust secret sharing scheme for the maximal corruption setting with $n = 2t + 1$ parties with nearly optimal share size of $m + \widetilde{k}$ bits, where m is the length of the message and 2^{-k} is the failure probability of the reconstruction procedure with adversarial shares.

One open question would be to optimize the poly-logarithmic terms in our construction. Indeed, it should be relatively easy to improve on these terms which we did not analyze carefully, but it seems challenging and interesting to attempt to go all the way down to $m + O(k)$ or perhaps even just $m + k$ bits. We leave this as a challenge for future work.

References

AD11. Anne Auger and Benjamin Doerr. *Theory of Randomized Search Heuristics: Foundations and Recent Developments*. World Scientific Publishing Co., Inc., River Edge, NJ, USA, 2011.

Bla79. George Robert Blakley. Safeguarding cryptographic keys. In *Managing Requirements Knowledge, International Workshop on*, pages 313–317. IEEE Computer Society, 1979.

BS97. Carlo Blundo and Alfredo De Santis. Lower bounds for robust secret sharing schemes. *Inf. Process. Lett.*, 63(6):317–321, 1997.

CDD+15. Ronald Cramer, Ivan Bjerre Damgård, Nico Döttling, Serge Fehr, and Gabriele Spini. Linear secret sharing schemes from error correcting codes and universal hash functions. In Elisabeth Oswald and Marc Fischlin, editors, *Advances in Cryptology - EUROCRYPT 2015 - 34th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Sofia, Bulgaria, April 26-30, 2015, Proceedings, Part II*, volume 9057 of *Lecture Notes in Computer Science*, pages 313–336. Springer, 2015.

CDF01. Ronald Cramer, Ivan Damgård, and Serge Fehr. On the cost of reconstructing a secret, or VSS with optimal reconstruction phase. In Joe Kilian, editor, *Advances in Cryptology - CRYPTO 2001, 21st Annual International Cryptology Conference, Santa Barbara, California, USA, August 19-23, 2001, Proceedings*, volume 2139 of *Lecture Notes in Computer Science*, pages 503–523. Springer, 2001.

CDF+08. Ronald Cramer, Yevgeniy Dodis, Serge Fehr, Carles Padró, and Daniel Wichs. Detection of algebraic manipulation with applications to robust secret sharing and fuzzy extractors. *IACR Cryptology ePrint Archive*, 2008:30, 2008.

Cev11. Alfonso Cevallos. Reducing the share size in robust secret sharing. http://www.algant.eu/documents/theses/cevallos.pdf, 2011.

CFOR12. Alfonso Cevallos, Serge Fehr, Rafail Ostrovsky, and Yuval Rabani. Unconditionally-secure robust secret sharing with compact shares. In David Pointcheval and Thomas Johansson, editors, *EUROCRYPT*, volume 7237 of *Lecture Notes in Computer Science*, pages 195–208. Springer, 2012.

Che15. Mahdi Cheraghchi. Nearly optimal robust secret sharing. Cryptology ePrint Archive, Report 2015/951, 2015. http://eprint.iacr.org/.

CSV93. Marco Carpentieri, Alfredo De Santis, and Ugo Vaccaro. Size of shares and probability of cheating in threshold schemes. In Tor Helleseth, editor, *Advances in Cryptology - EUROCRYPT '93, Workshop on the Theory and Application of of Cryptographic Techniques, Lofthus, Norway, May 23-27, 1993, Proceedings*, volume 765 of *Lecture Notes in Computer Science*, pages 118–125. Springer, 1993.

FK02. Uriel Feige and Robert Krauthgamer. A polylogarithmic approximation of the minimum bisection. *SIAM J. Comput.*, 31(4):1090–1118, 2002.

GJS76. M. R. Garey, David S. Johnson, and Larry J. Stockmeyer. Some simplified np-complete graph problems. *Theor. Comput. Sci.*, 1(3):237–267, 1976.

Gol98. Oded Goldreich. *Modern Cryptography, Probabilistic Proofs and Pseudorandomness*, volume 17 of *Algorithms and Combinatorics*. Springer, 1998.

IK10. Russell Impagliazzo and Valentine Kabanets. Constructive proofs of concentration bounds. In Maria J. Serna, Ronen Shaltiel, Klaus Jansen, and José D. P. Rolim, editors, *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques, 13th International Workshop, APPROX 2010, and 14th International Workshop, RANDOM 2010, Barcelona, Spain, September 1-3, 2010. Proceedings*, volume 6302 of *Lecture Notes in Computer Science*, pages 617–631. Springer, 2010.

LP14. Allison Bishop Lewko and Valerio Pastro. Robust secret sharing schemes against local adversaries. *IACR Cryptology ePrint Archive*, 2014:909, 2014.

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.

Räc08. Harald Räcke. Optimal hierarchical decompositions for congestion minimization in networks. In Cynthia Dwork, editor, *Proceedings of the 40th Annual ACM Symposium on Theory of Computing, Victoria, British Columbia, Canada, May 17-20, 2008*, pages 255–264. ACM, 2008.

RB89. Tal Rabin and Michael Ben-Or. Verifiable secret sharing and multiparty protocols with honest majority (extended abstract). In David S. Johnson, editor, *Proceedings of the 21st Annual ACM Symposium on Theory of Computing, May 14-17, 1989, Seattle, Washigton, USA*, pages 73–85. ACM, 1989.

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

Sud97. Madhu Sudan. Decoding of reed solomon codes beyond the error-correction bound. *J. Complexity*, 13(1):180–193, 1997.

A Additional Material

A.1 Properties of Polynomial Evaluation

Here, we list some useful properties of the polynomial evaluation defined in section 2.2

Claim (ℓ-wise Independence.). For any $\ell \le d$, any $d-\ell$ values $a_{\ell+1},\dots,a_d \in \mathbb{F}$, any ℓ distinct values $x_1,\dots,x_\ell \in \mathbb{F}^*$, and any $y_1,\dots,y_\ell \in \mathbb{F}$ we have:

$$\Pr[\mathsf{PEval}(\boldsymbol{a},x_1)=y_1,\dots,\mathsf{PEval}(\boldsymbol{a},x_\ell)=y_\ell \;:\; (a_1,\dots,a_\ell)\leftarrow \mathbb{F}^\ell, \boldsymbol{a}=(a_1,\dots,a_d)] = \frac{1}{|\mathbb{F}|^\ell}$$

In other words, the values $\mathsf{PEval}(\boldsymbol{a},x_1),\dots,\mathsf{PEval}(\boldsymbol{a},x_\ell)$ are uniformly random and independent when the first ℓ components of $\boldsymbol{a}$ are chosen at random.

Proof. There is a unique choice of $a_1,\dots,a_\ell$ such that $\mathsf{PEval}(\boldsymbol{a},x_1)=y_1,\dots,\mathsf{PEval}(\boldsymbol{a},x_\ell)=y_\ell$. In particular, $a_0=0,a_1,\dots,a_\ell$ are the coefficients of unique degree ℓ polynomial $p(x)=\sum_{i=0}^{\ell} a_i x^i$ such that $p(0)=0$ and $p(x_i)=y_i-\sum_{i=\ell+1}^{d} a_i x^i$.

Claim ((XOR) Universality). Let $\mathcal{K}\subseteq\mathbb{F}$. Then for any $\boldsymbol{a}\neq\boldsymbol{a}'\in\mathbb{F}^d$ and any $y\in\mathbb{F}$ we have $\Pr_{x\leftarrow\mathcal{K}}[\mathsf{PEval}(\boldsymbol{a},x)-\mathsf{PEval}(\boldsymbol{a}',x)=y]\le\frac{d}{|\mathcal{K}|}$. [6] In particular, the hash family $\mathcal{H}=\{H_x \;:\; \mathbb{F}^d\to\mathbb{F}\}_{x\in\mathbb{F}}$ defined by $H_x(\boldsymbol{a})=\mathsf{PEval}(\boldsymbol{a},x)$ is $\varepsilon=\frac{d}{|\mathbb{F}|}$-universal.

Proof. The event $\mathsf{PEval}(\boldsymbol{a},x)-\mathsf{PEval}(\boldsymbol{a}',x)=y$ occurs iff x is a root of the non-zero degree $\le d$ polynomial $\sum_{i=1}^{d}(a_i-a_i')x^i-y=0$. Since there are at most d roots of this polynomial, the probability of the above happening over a random choice of $x\leftarrow\mathcal{K}$ is at most $\varepsilon=\frac{d}{|\mathcal{K}|}$.

B Proofs

B.1 Proof of Theorem 4

Proof. Let Adv be any unbounded adversary and $\mathsf{msg}\in\mathcal{M}$ be some value. Consider the robustness game $\mathbf{Exp}(\mathsf{msg},\mathsf{Adv})$ from Definition 4. We show that $\Pr[\mathbf{Exp}(\mathsf{msg},\mathsf{Adv})=1]\le\delta$. Let $(s_1,\dots,s_n)$ be the original shares created by the sharing algorithm in step E.1 of the experiment with $s_i=(\hat{s}_i,k_i,y_i)$. Let $I\subseteq[n]$, $|I|=t$ be the set of corrupted parties at the end of step E.2 of the experiment, and let $s_i'=(\hat{s}_i',k_i',y_i')$ be the modified shares at the end of step E.3 of the experiment. Let $\mathsf{msg}^{(1)},\dots,\mathsf{msg}^{(L)}$ be the list of recovered messages in step E.4 of the experiment. By the correctness of list-decoding with t errors we know that $\mathsf{msg}=\mathsf{msg}^{(j^*)}$ for some j^* and that we have $|\{i\in[n] \;:\; H_{k_i'}(\mathsf{msg}^{(j^*)})=y_i'\}|\ge|[n]\setminus I|\ge t+1$. Therefore, the only way that the wrong message is recovered is if there exists some $j\in[L]$, $j\neq j^*$ such that $|\{i\in[n] \;:\; H_{k_i'}(\mathsf{msg}^{(j)})=y_i'\}|\ge t+1$.

[6] We can think of this as a generalization of XOR universality over a an arbitrary field $\mathbb{F}$.

This only happens if there exists some $i \in [n] \setminus I$ such that $H_{k_i}(\mathsf{msg}^{(j)}) = y_i$ (note that for $i \notin I$, $k'_i = k_i, y'_i = y_i$). The key k_i for $i \notin I$ is random and independent of the adversary's view in the experiment and therefore also of the message $\mathsf{msg}^{(j)}$ which is completely determined by the adversary's view. Therefore, by the ε-universality of the hash, the probability of the above happening for any $i \in [n] \setminus I, j \in [L] \setminus \{j^*\}$ is at most ε. By a union bound over all such i, j the probability that the wrong message is recovered is $\Pr[\mathbf{Exp}(\mathsf{msg}, \mathsf{Adv}) = 1] \leq nL\varepsilon = \delta$.

B.2 Proof of Theorem 5

Proof. We prove the three properties separately.

Authentication. Let $(\mathsf{lab}_i, \mathsf{msg}_i, r_i, \sigma_i) \;:\; i = 1, \ldots, \ell$ be such that the labels lab_i are distinct, and let $(\mathsf{lab}', \mathsf{msg}', r', \sigma')$ be such that $(\mathsf{lab}', \mathsf{msg}', r') \notin \{(\mathsf{lab}_i, \mathsf{msg}_i, r_i)\}_{i \in [\ell]}$. We consider two cases:

Case 1: $\mathsf{lab}' \notin \{\mathsf{lab}_i\}_{i \in [\ell]}$. In this case $\mathsf{key}_1^{\mathsf{lab}'}, \mathsf{key}_2^{\mathsf{lab}'}$ are random and independent of $\{\mathsf{key}_1^{\mathsf{lab}_i}, \mathsf{key}_2^{\mathsf{lab}_i}\}_{i \in [\ell]}$. This follows by the $(\ell+1)$-wise indepdnence of $\mathsf{PEval}(\mathsf{key}, \cdot)$ from Claim A.1. Therefore:

$$\begin{aligned}
&\Pr_{\mathsf{key} \leftarrow \mathcal{K}}[\mathsf{MAC}_{\mathsf{key}}(\mathsf{lab}', \mathsf{msg}', r') = \sigma' \mid \{\mathsf{MAC}_{\mathsf{key}}(\mathsf{lab}_i, \mathsf{msg}_i, r_i) = \sigma_i\}_{i \in [\ell]}] \\
&\leq \Pr_{\mathsf{key} \leftarrow \mathcal{K}}[\mathsf{key}_2^{\mathsf{lab}'} = \sigma' - \mathsf{PEval}(\ (r', \mathsf{msg}')\ ,\ (\mathsf{lab}', \mathsf{key}_1^{\mathsf{lab}'})\ )\ \mid \{\mathsf{MAC}_{\mathsf{key}}(\mathsf{lab}_i, \mathsf{msg}_i, r_i) = \sigma_i\}_{i \in [\ell]}] \\
&\leq 1/|\mathbb{F}| \leq \varepsilon.
\end{aligned}$$

where the last line relies on the fact that the values $\mathsf{MAC}_{\mathsf{key}}(\mathsf{lab}_i, \mathsf{msg}_i, r_i)$ only depend on $\{\mathsf{key}_1^{\mathsf{lab}_i}, \mathsf{key}_2^{\mathsf{lab}_i}\}_{i \in [\ell]}$.

Case 2: $\mathsf{lab}' = \mathsf{lab}_i$ for some $i \in [\ell]$. In this case $\mathsf{key}_1^{\mathsf{lab}_i}, \mathsf{key}_2^{\mathsf{lab}_i}$ are random and independent of $\mathsf{key}_1^{\mathsf{lab}_j}$, $\mathsf{key}_2^{\mathsf{lab}_j}$ for $j \in [\ell] \setminus \{i\}$. This follows by the $(\ell+1)$-wise indepdnence of $\mathsf{PEval}(\mathsf{key}, \cdot)$ from Claim A.1. Therefore:

$$\begin{aligned}
&\Pr_{\mathsf{key} \leftarrow \mathcal{K}}[\mathsf{MAC}_{\mathsf{key}}(\mathsf{lab}', \mathsf{msg}', r') = \sigma' \mid \{\mathsf{MAC}_{\mathsf{key}}(\mathsf{lab}_i, \mathsf{msg}_i, r_i) = \sigma_i\}_{j \in [\ell]}] \\
&\leq \Pr_{(\mathsf{key}_1^{\mathsf{lab}_i}) \leftarrow \mathbb{F}'} \left[\begin{matrix} \mathsf{PEval}(\ (r', \mathsf{msg}')\ ,\ (\mathsf{lab}_i, \mathsf{key}_1^{\mathsf{lab}_i})\ ) \\ -\mathsf{PEval}(\ (r_i, \mathsf{msg}_i)\ ,\ (\mathsf{lab}_i, \mathsf{key}_1^{\mathsf{lab}_i})\ ) \end{matrix} = \sigma' - \sigma_i \right] \\
&\leq \frac{m + \ell}{|\mathbb{F}'|} = \varepsilon
\end{aligned}$$

where the second line follows since the events $\{\mathsf{MAC}_{\mathsf{key}}(\mathsf{lab}_i, \mathsf{msg}_i, r_i) = \sigma_i\}_{j \in [\ell]}$ are independent of $\mathsf{key}_1^{\mathsf{lab}_i}$ and the third line follows from Claim A.1 where the value $(\mathsf{lab}_i, \mathsf{key}_1^{\mathsf{lab}_i}) \in \mathbb{F}$ is uniformly random over a subset of size $|\mathbb{F}'|$.

Privacy over Keys. Let $(\mathsf{lab}_i, \mathsf{msg}_i, r_i) \in \mathcal{L} \times \mathcal{M} \times \mathcal{R} \;:\; i = 1, \ldots, \ell$ be ℓ fixed values such that the labels lab_i are distinct. Then the ℓ values $\{\mathsf{key}_2^{\mathsf{lab}_i} =$

$\mathsf{PEval}(\mathsf{key}_2, \mathsf{lab}_i)\}_{i\in[\ell]}$ are ℓ-wise independent (Claim A.1) over the random choice of key_2. Therefore so are the ℓ values $\sigma_i = \mathsf{MAC}_{\mathsf{key}}(\mathsf{lab}_i, \mathsf{msg}_i, r_i) = \mathsf{PEval}(\ (r_i, \mathsf{msg}_i)\ ,\ (\mathsf{lab}_i, \mathsf{key}_1^{\mathsf{lab}_i})\ ) + \mathsf{key}_2^{\mathsf{lab}_i}$.

Privacy over Randomness. Fix any $\mathsf{msg} \in \mathcal{M}$, any ℓ distinct labels $\mathsf{lab}_1, \ldots, \mathsf{lab}_\ell$, and any keys $\mathsf{key}_1, \ldots, \mathsf{key}_\ell \in \mathcal{K}$ with $\mathsf{key}_i = (\mathsf{key}_{i,1}, \mathsf{key}_{i,2})$. Let $\mathsf{key}_1^{\mathsf{lab}_i} = \mathsf{PEval}(\mathsf{key}_{i,1}, \mathsf{lab}_i), \mathsf{key}_2^{\mathsf{lab}_i} = \mathsf{PEval}(\mathsf{key}_{i,2}, \mathsf{lab}_i)$. Let $\sigma_i = \mathsf{MAC}_{\mathsf{key}_i}(\mathsf{lab}_i, \mathsf{msg}, r) = \mathsf{PEval}(\ (r, \mathsf{msg})\ ,\ (\mathsf{lab}_i, \mathsf{key}_1^{\mathsf{lab}_i})\ ) + \mathsf{key}_2^{\mathsf{lab}_i}$ for $i = 1, \ldots, \ell$ be random variables over the choice of $r \leftarrow \mathcal{R}$. Then $\{\sigma_i\}_{i\in[\ell]}$ are random and independent by the ℓ-wise independence of $\mathsf{PEval}(\ (r, \mathsf{msg})\ ,\ (\mathsf{lab}_i, \mathsf{key}_1^{\mathsf{lab}_i})\ )$ over the random choice of r (Claim A.1) and the fact that the labels lab_i are distinct.

B.3 Proof of Lemma 3

Proof. Let Adv be any unbounded adversary in the robustness game (Definition 4). We show that $\Pr[\mathbf{Exp}(\mathsf{msg}, \mathsf{Adv}) = 1] \leq \delta$. Our proof follows by defining several hybrid distributions.

Hybrid 0: This is the original robustness game $\mathbf{Exp}(\mathsf{msg}, \mathsf{Adv})$ with a message msg and an adversary Adv as in Definition 4.

Hybrid 1: This is a modified game $\mathbf{Exp}^{(1)}(\mathsf{msg}, \mathsf{Adv})$ which is the same as $\mathbf{Exp}(\mathsf{msg}, \mathsf{Adv})$ except that we modify how the shares are reconstructed in step E.4 of the experiment. In particular, instead of running step R.1 of reconstruction to compute $\mathsf{tags}' = \mathsf{Rec}_{rds}(p'_1, \ldots, p'_n)$ we now set $\mathsf{tags}' := \mathsf{tags}$ to be the original value chosen by the sharing procedure in step E.1 of the experiment. By the security of the robust distributed storage, it's easy to see:

$$\Pr[\mathbf{Exp}^{(1)}(\mathsf{msg}, \mathsf{Adv})] \geq \Pr[\mathbf{Exp}(\mathsf{msg}, \mathsf{Adv})] - \delta_{rds}.$$

Hybrid 2: This is a modified game $\mathbf{Exp}^{(2)}(\mathsf{msg}, \mathsf{Adv})$ which is the same as $\mathbf{Exp}^{(1)}(\mathsf{msg}, \mathsf{Adv})$ except that we modify how the shares are created in step E.1 of the experiment. In particular, we modify step S.3 of the sharing procedure to do the following:

- For each $i \in [n]$ choose $\mathsf{key}_i \leftarrow \mathcal{K}$.
- For each $(i, j) \in E$ choose $(\sigma_{i\to j}) \leftarrow \mathcal{T}$ and $(\sigma_{i\leftarrow j}) \leftarrow \mathcal{T}$ uniformly at random.
- For each $i \in [n]$ choose r_i uniformly at random from $\mathcal{R}$ *conditioned on*

$$\begin{Bmatrix} \forall (j,i) \in E & : & \mathsf{MAC}_{\mathsf{key}_j}((j,i,0), (\widetilde{s}_i, E_i, \mathsf{key}_i), r_i) = \sigma_{j\to i} \\ \forall (i,j) \in E & : & \mathsf{MAC}_{\mathsf{key}_j}((i,j,1), (\widetilde{s}_i, E_i, \mathsf{key}_i), r_i) = \sigma_{i\leftarrow j} \end{Bmatrix} \tag{B.1}$$

We claim that the joint distribution of keys, tags, and randomness:

$$\{\mathsf{key}_i\}_{i\in[n]} \quad , \quad \{\sigma_{i\to j}\ ,\ \sigma_{i\leftarrow j}\}_{(i,j)\in E} \quad , \quad \{r_i\}_{i\in[n]} \tag{B.2}$$

is identical in Hybrid 1 and 2, and therefore the Hybrids are altogether identical.

Firstly, let's only consider the joint distribution over only the keys and tags:

$$\{\mathsf{key}_i\}_{i\in[n]} \quad , \quad \{\sigma_{i\to j}, \sigma_{i\leftarrow j}\}_{(i,j)\in E} \tag{B.3}$$

In Hybrid 1, the tags are the output of the MAC whereas in Hybrid 2 they are uniformly random. However, for every $i \in [n]$ there are at most $2d$ tags of the form $\sigma_{j\to i}$ (corresponding to the in-degree of i) and exactly d tags of the form $\sigma_{i\leftarrow j}$ (corresponding to the out-degree of i). In Hybrid 1, all of these tags are created by computing the MAC using distinct labels and these are the only tags that rely on the randomness r_i. Therefore by the "privacy over randomness" of the MAC with $\ell = 3d$, these tags are uniformly random and independent over the choice of r_i (for any choice of keys). This shows that the distribution of keys and tags (B.3) is identical in Hybrid 1 and 2.
Once we fix any choice of keys, tags the values r_i then follows the same conditional distribution in Hybrids 1 and 2. In particular in both cases they are uniformly random over $\mathcal{R}$ subject to satisfying (B.1). This shows that the distribution of (B.2) is identical in Hybrid 1 and 2. Therefore the experiments $\mathbf{Exp}^{(2)}(\mathsf{msg}, \mathsf{Adv})$ and $\mathbf{Exp}^{(1)}(\mathsf{msg}, \mathsf{Adv})$ are identically distributed and we have:

$$\Pr[\mathbf{Exp}^{(2)}(\mathsf{msg}, \mathsf{Adv})] = \Pr[\mathbf{Exp}^{(1)}(\mathsf{msg}, \mathsf{Adv})]$$

Hybrid 3: This is a modified game $\mathbf{Exp}^{(3)}(\mathsf{msg}, \mathsf{Adv})$ which is the same as $\mathbf{Exp}^{(2)}(\mathsf{msg}, \mathsf{Adv})$ except that we modify how the shares are reconstructed in step E.4 of the experiment. Let $I \subseteq [n]$ be the set of parties corrupted by the adversary at the end of step E.2. Let $s'_i = (\widetilde{s}'_i, E'_i, \mathsf{key}'_i, r'_i, p'_i)$ be the modified shares submitted by the adversary in step E.3 and let $s_i = (\widetilde{s}_i, E_i, \mathsf{key}_i, r_i, p_i)$ be the original shares created by the sharing procedure. Let us define the set $A \subseteq I$ as:

$$A = \{i \in I \ : \ (\widetilde{s}'_i, E'_i, \mathsf{key}'_i, r'_i) \neq (\widetilde{s}_i, E_i, \mathsf{key}_i, r_i)\}.$$

We refer to A as the *active* corruptions. We define $P = I \setminus A$ as the *passive* corruptions and $H = [n] \setminus I$ as the *honest* parties. In Hybrid 3, we modify step R.3 of the reconstruction procedure to do the following

- Set $L(e) := \mathsf{bad}$ for all edges $e \in E'_{A\to H} \cup E'_{H\to A}$. These are the edges where one end point is honest and the other active.
- Set $L(e) := \mathsf{good}$ for all edges $e \in E'_{H\cup P\to H\cup P}$. These are the edges where neither end-point is actively corrupted.
- For all other edges e compute the labeling $L(e)$ as previously by verifying the MAC tags.

The only way that Hybrid 2 and 3 could differ is if one of the following $\leq nd$ "forgery events" occurs:

- For $(i,j) \in E'$ with $i \notin I, j \in A$: Event $\mathsf{MAC}_{\mathsf{key}_i}((i,j,0), (\widetilde{s}'_j, E'_j, \mathsf{key}'_j), r'_j) = \sigma'_{i\to j}$.
- For $(j,i) \in E'$ with $i \notin I, j \in A$: Event $\mathsf{MAC}_{\mathsf{key}_i}((j,i,1), (\widetilde{s}'_j, E'_j, \mathsf{key}'_j), r'_j) = \sigma'_{j\leftarrow i}$.

Note that for each $i \notin I$, the key key_i is uniformly random from the point of view of the adversary after the corruption stage (step E.2) subject to the $\leq 3d$ constraints: [7]

$$\forall (i,j) \in E \quad : \quad \mathsf{MAC}_{\mathsf{key}_i}((i,j,0),(\widetilde{s}_j, E_j, \mathsf{key}_j), r_j) = \sigma_{i \to j}$$

$$\forall (j,i) \in E \quad : \quad \mathsf{MAC}_{\mathsf{key}_i}((j,i,1),(\widetilde{s}_j, E_j, \mathsf{key}_j), r_j) = \sigma_{j \leftarrow i}$$

Therefore, by the authentication property of the MAC, the probability of any specific forgery event occurring is at most ε_{mac}. By the union bound, the probability that some such event occurs is at most $2nd\varepsilon_{mac}$.
This shows:

$$\Pr[\mathbf{Exp}^{(3)}(\mathsf{msg},\mathsf{Adv}) = 1] \geq \Pr[\mathbf{Exp}^{(2)}(\mathsf{msg},\mathsf{Adv}) = 1] - 2nd\varepsilon_{mac}.$$

Hybrid 4: We define Hybrid 4 via the experiment $\mathbf{Exp}^{(4)}(\mathsf{msg},\mathsf{Adv})$ which is the same as $\mathbf{Exp}^{(3)}(\mathsf{msg},\mathsf{Adv})$ except that we modify how the shares are created in step E.1 of the experiment. In particular, we modify steps S.2 and S.3 of the sharing procedure to do the following:

- In step S.2, we now omit condition (b) and no longer re-sample the graph G is some vertex has in-degree $> 2d$. In particular, we now just choose $G = ([n], E)$ by choosing $E_i \subseteq [n] \setminus \{i\}$, $|E_i| = d$ uniformly at random and define $E = \{(i,j) \ : \ i \in [n], j \in E_i\}$. Let INDEG be the event that the graph we sample has some vertex with in-degree $> 2d$.
- If the event INDEG does *not* occur, then step S.3 proceeds the same way as before. If INDEG does occur, then we modify step S.3 to choose the keys, tags, and randomness:

$$\{\mathsf{key}_i\}_{i \in [n]}, \{\sigma_{i \to j}, \sigma_{i \leftarrow j}\}_{(i,j) \in E}, \{r_i\}_{i \in [n]}$$

  uniformly at random and independently of each other.

Hybrids 3 and 4 can only differ if in Hybrid 4 the event INDEG does occur. This means that there is some vertex $j \in [n]$ with in-degree $> 2d$. For each $i \in [n] \setminus j$, the probability that the edge (i,j) is in E is $d/(n-1)$ and therefore the expected in-degree of vertex j is $\leq d$. These events are independent and therefore, by Chernoff, the probability of vertex j having in-degree $> 2d$ is $< 2^{-d/3}$. By the union bound over all $j \in [n]$, the probability that there exists some vertex $j \in [n]$ with in-degree $> 2d$ is $< n2^{-d/3}$. Therefore we have:

$$\Pr[\mathbf{Exp}^{(4)}(\mathsf{msg},\mathsf{Adv}) = 1] \geq \Pr[\mathbf{Exp}^{(3)}(\mathsf{msg},\mathsf{Adv}) = 1] - n2^{-d/3}.$$

Hybrid 5: We observe that in Hybrid 4, the adversary does not learn anything about the sets $E_i \ : \ i \notin I$ during the corruption stage (at the end of step E.2). In other words, from the point of the adversary, each such set is uniformly

[7] For this argument, we can even condition on a worst-case choice of all tags $\sigma_{i \to j}, \sigma_{i \leftarrow j}$ and the edges E as well as all other keys $\{\mathsf{key}_j \ : \ j \neq i\}$.

random over $[n] \setminus \{i\}$ subject to $|E_i| = d$. To see this, note that when the event INDEG occurs in Hybrid 4 than the adversary truly does not see any information about $E_i \ : \ i \notin I$ at the end of step E.2. On the other hand when the event INDEG does occur then the only information available to the adversary about each set E_i and key key_i for $i \notin I$ at the end of step E.2 in Hybrid 4 is:[8]

$$\left\{ \begin{array}{lcl} \forall j \in E_i & : & \mathsf{MAC}_{\mathsf{key}_i}((i,j,0),(\widetilde{s}_j, E_j, \mathsf{key}_j), r_j) = \sigma_{i \to j} \\ \forall j \text{ s.t. } i \in E_j : & : & \mathsf{MAC}_{\mathsf{key}_i}((j,i,1),(\widetilde{s}_j, E_j, \mathsf{key}_j), r_j) = \sigma_{j \leftarrow i} \end{array} \right\} \quad \text{(B.4)}$$

By the privacy over keys property of the MAC, for any choice of E_i with $|E_i| = d$ the probability of (B.4) happening over the random choice of key_i is exactly $1/|\mathcal{T}|^{d'}$ where $d' \leq 3d$ is the total degree of vertex i. In particular, this probability is the same for every choice of E_i, $|E_i| = d$. Therefore, the adversary's views at the end of step E.2 in Hybrid 4 is independent of $E_i \ : \ i \notin I$.

We formalize this by defining the hybrid experiment $\mathbf{Exp}^{(5)}(\mathsf{msg}, \mathsf{Adv})$ which is the same as $\mathbf{Exp}^{(4)}(\mathsf{msg}, \mathsf{Adv})$ except that we modify how the shares are reconstructed in step E.4 of the experiment. In step R.2 we re-sample a completely fresh set $E'_i \subseteq [n] \setminus \{i\}$ of size $|E'_i| = d$ uniformly at random for each $i \notin I$ instead of using the set $E'_i = E_i$ that was chosen by the sharing procedure. We use these freshly re-sampled sets to define the graph $G' = ([n], E')$.

By the above argument

$$\Pr[\mathbf{Exp}^{(5)}(\mathsf{msg}, \mathsf{Adv}) = 1] \geq \Pr[\mathbf{Exp}^{(6)}(\mathsf{msg}, \mathsf{Adv}) = 1]$$

Hybrid 6. This is a modified game $\mathbf{Exp}^{(6)}(\mathsf{msg}, \mathsf{Adv})$ which is the same as $\mathbf{Exp}^{(5)}(\mathsf{msg}, \mathsf{Adv})$ except that we modify how the shares are reconstructed in step E.4 of the experiment. In particular, we change step R.4 of the reconstruction procedure so that, instead of outputting $B \leftarrow \mathsf{GraphID}(G', L)$, it sets $B = H \cup P$, where the sets A, H, P are defined the same way as in Hybrid 3.

We observe that the distribution of (G', L, H, A, P) in Hybrid 5 is exactly that of $\mathsf{Gen}^{\mathsf{Adv}'}(n, t, d)$ where the adversary Adv' is defined as follows:

- Run steps E.1, E.2 and E.3 of $\mathbf{Exp}^{(5)}(\mathsf{msg}, \mathsf{Adv})$. This defines the values s_i, s'_i and the sets A, H, P as described above.
- Let $E'_{A \cup P \to [n]} = \bigcup_{i \in I} E'_i$.
- For each edge $e = (i,j) \in E'_{A \to A \cup P} \cup E'_{A \cup P \to A}$ define $L(e) = \mathsf{good}$ if

  $$\sigma'_{i \to j} = \mathsf{MAC}_{\mathsf{key}'_i}((i,j,1),(\widetilde{s}'_j, E'_j, \mathsf{key}'_j), r'_j) \quad \text{and} \quad \sigma'_{i \leftarrow j} = \mathsf{MAC}_{\mathsf{key}'_j}((i,j,1),(\widetilde{s}'_i, E'_i, \mathsf{key}'_i), r'_i)$$

  and otherwise $L(e) = \mathsf{bad}$.

[8] For this argument, we can even condition on a worst-case choice of all tags and as well as all other keys and edges except for key_i, E_i.

We observe that, in Hybrid 5 just like in the game $\mathsf{Gen}^{\mathsf{Adv}'}(n,t,d)$, the edges $E'_{H\to[n]}$ are chosen uniformly at random subject to the out-degree being d. Furthermore, in both cases we define $L(e) = \mathsf{good}$ for all $e \in E'_{H\cup P\to H\cup P}$ and $L(e) = \mathsf{bad}$ for all $e \in E'_{A\to H} \cup E'_{H\to A}$. Therefore (G', L, H, A, P) in Hybrid 5 has the distribution of $\mathsf{Gen}^{\mathsf{Adv}'}(n,t,d)$. By the guarantee of the graph identification algorithm we therefore have that, when we execute $B \leftarrow \mathsf{GraphID}(G', L)$ in step R.4 of the reconstruction process in Hybrid 5, then:

$$\Pr[B = H \cup P] \leq 1 - \delta_{gi}.$$

Therefore,

$$\Pr[\mathbf{Exp}^{(6)}(\mathsf{msg}, \mathsf{Adv}) = 1] \geq \Pr[\mathbf{Exp}^{(5)}(\mathsf{msg}, \mathsf{Adv}) = 1] - \delta_{gi}.$$

Conclusion. Finally, we observe that in Hybrid 6 we have

$$\Pr[\mathbf{Exp}^{(6)}(\mathsf{msg}, \mathsf{Adv}) = 1] = 0.$$

This is because, when the shares are reconstructed in step E.5 of the experiment, we are guaranteed that $\widetilde{s}'_i = \widetilde{s}_i$ for $i \in B$. Furthermore, since $B = H \cup P$, we know that $|B| \geq t+1$. Therefore, by the perfect reconstruction with erasures property of the non-robust secret sharing scheme we are guaranteed that for any $B' \subseteq B$ of size $|B'| = t+1$ we have $\mathsf{Rec}_{nr}((\widetilde{s}'_i)_{i\in B'}) = \mathsf{msg}$.

Combining the above, we get

$$\Pr[\mathbf{Exp}(\mathsf{msg}, \mathsf{Adv}) = 1] \leq \delta_{rds} + \delta_{gi} + 2dn\varepsilon_{mac} + n2^{-d/3}$$

which concludes the proof.

B.4 Proof of Theorem 7

Proof. The fact that the scheme $(\mathsf{Share}', \mathsf{Rec}')$ satisfies perfect privacy follows the same argument as in Lemma 1, and the fact that it satisfies perfect reconstruction with erasures follows the same argument as in Lemma 2.

Therefore, we are left to analyze robustness. Let us consider the robustness experiment $\mathbf{Exp}(\mathsf{msg}, \mathsf{Adv})$ for the original scheme $(\mathsf{Share}, \mathsf{Rec})$. Let us define $\mathbf{Exp}_{\mathsf{coins}_{nr}}(\mathsf{msg}, \mathsf{Adv})$ to be the experiment when using some fixed choice of coins_{nr} for the non-robust secret sharing scheme in step S.1 of the sharing procedure. It follows from the proof of Lemma 3 that for any $\mathsf{msg} \in \mathcal{M}$, any choice of coins_{nr} and for all adversaries Adv we have $\Pr[\mathbf{Exp}_{\mathsf{coins}_{nr}}(\mathsf{msg}, \mathsf{Adv}) = 1] \leq \delta$. In particular, the proof of Lemma 3 did not rely on the randomness of coins_{nr} anywhere and works equally well if we fix them to some worst-case value.

Let us define $\mathbf{Exp}'(\mathsf{msg}, \mathsf{Adv}')$ to be the robustness experiment for the q-wise parallel repetition scheme defined above. Assume that there exists some adversary Adv' and message $\mathsf{msg} \in \mathcal{M}$ such that $\Pr[\mathbf{Exp}'(\mathsf{msg}, \mathsf{Adv}') = 1] = \delta'$. We can think of Adv' as participating in q parallel copies of the game $\mathbf{Exp}_{\mathsf{coins}_{nr}}(\mathsf{msg}, \cdot)$, where coins_{nr} are chosen randomly once but are re-used in every copy of the game. The adversary Adv' has to win in more than $q/2$ of the copies to win in $\mathbf{Exp}'(\mathsf{msg}, \mathsf{Adv}')$. In particular, there must exist some choice of coins_{nr} such that Adv' has probability δ' in winning in at least $q/2$ out of q parallel independent copies of the interactive game $\mathbf{Exp}_{\mathsf{coins}_{nr}}(\mathsf{msg}, \cdot)$. Therefore, by employing Chernoff bounds for the parallel-repetition of information theoretic interactive games (i.e., interactive proofs – see e.g., [Gol98,IK10]) the probability of the above happening is bounded by $\delta' \leq e^{-\frac{3}{128}q}$.