Robust Secret Sharing with Almost Optimal Share Size and Security Against Rushing Adversaries

Serge Fehr[1,2] and Chen Yuan[1(✉)]

[1] CWI, Amsterdam, The Netherlands
{serge.fehr,chen.yuan}@cwi.nl
[2] Mathematical Institute, Leiden University, Leiden, The Netherlands

Abstract. We show a robust secret sharing scheme for a maximal threshold $t < n/2$ that features an optimal overhead in share size, offers security against a rushing adversary, and runs in polynomial time. Previous robust secret sharing schemes for $t < n/2$ either suffered from a suboptimal overhead, offered no (provable) security against a rushing adversary, or ran in superpolynomial time.

1 Introduction

Background. Robust secret sharing is a version of secret sharing that enables the reconstruction of the shared secret s in the presence of *incorrect* shares: given all n shares but with t of them possibly incorrect, and of course without knowing which ones are incorrect, it should still be possible to recover s. If $t < n/3$ then this can be achieved by standard error-correction techniques, while for $t \geq n/2$ the task is impossible. When $n/3 \leq t < n/2$, robust secret sharing is possible, but only if one accepts a small failure probability and an overhead in the share size, i.e., shares of bit size *larger* than the bit size of s. The goal then is to minimize the overhead in the share size for a given (negligible) failure probability 2^{-k}. Following up on earlier work on the topic [2,4–7,10], Bishop et al. proposed a scheme with optimal overhead $O(k)$ in the share size, neglecting polylogarithmic terms (in n and k and the bit size of s) [3]. In particular, their scheme was the first robust secret sharing with an overhead that is *independent* of n (neglecting *polylog*(n) terms). However, as pointed out by Fehr and Yuan [8], the Bishop et al. scheme does not (appear to) offer security in the presence of a *rushing* adversary that may choose the incorrect shares depending on the shares of the honest parties. This is in contrast to most of the earlier schemes, which do offer security against such rushing attacks (but are less efficient in terms of

R. Pass and K. Pietrzak (Eds.): TCC 2020, LNCS 12552, pp. 470–498, 2020.
https://doi.org/10.1007/978-3-030-64381-2_17

share size).[1] Towards recovering security against a rushing adversary, Fehr and Yuan [8] proposed a new robust secret sharing scheme that features security against a rushing adversary *and* an overhead "almost independent" of n, i.e., $O(n^\epsilon)$ for an arbitrary $\epsilon > 0$. Furthermore, a variation of their scheme offers security against a rushing adversary and an overhead that is *truly independent* of n (neglecting polylogarithmic terms), but this version of the scheme has a running time that is superpolynomial.

Our Result. In this work, we close the final gap left open in [8]: we propose and analyze a new robust secret sharing scheme that is secure against a rushing adversary, has an overhead independent of n as in [3] (i.e., independent up to the same poly-logarithmic $O(\log^4 n + \log n \log m)$ term as in [3], where m is the bit size of the secret), and has a polynomial running time.

Our new scheme recycles several of the ideas and techniques of [8]. The basic idea, which goes back to [3], is to have each share s_i be authenticated by a *small randomly chosen subset* of the other parties. Following [8], our approach here differs from [3] in that the keys for the authentication are not authenticated. Indeed, this "circularity" of having the authentication keys authenticated causes the solution in [3] to not allow a rushing adversary; on the other hand, by not authenticating the authentication keys, we give the dishonest parties more flexibility in lying, making the reconstruction harder.

The reconstruction is in terms of a careful (and rather involved) inspection of the resulting consistency graph, exploiting that every honest party can verify the correctness of the shares of a small but *random* subset of parties, and that these choices of random "neighborhoods" become known to the adversary only *after* having decided about which shares s_i to lie about. As a matter of fact, in our scheme, every honest party can verify the correctness of the shares of *several* randomly chosen small neighborhoods, giving rise to several global verification graphs. Furthermore, to ensure "freshness" of each such neighborhood conditioned on the adversary's behavior so far, these neighborhoods are revealed *sequentially* in subsequent rounds of communication during the reconstruction phase.

As in [8], in our scheme the reconstructor first learns from the consistency graph whether the number p of "passive" parties, i.e., dishonest parties that did not lie about the actual share s_i (but possibly about other pieces of information), is "large" or "small". For p small, we can recycle the solution from [8], which happens to also work for the tighter parameter setting we consider here. When p is large though, the solution in [8] is to exploit the given redundancy in the shares s_i by means of applying list decoding, and then to find the right candidate from the list by again resorting to the consistency graph. However, this list decoding

[1] In order to achieve security against a rushing adversary when $n/3 \leq t < n/2$, it is necessary that the shares are disclosed part-by-part in subsequent rounds of communication; for the adversary to be *rushing* then means that he can choose his messages in each communication round depending on the parts of the shares of the honest parties that are communicated in this round (and earlier ones), but not depending on what the honest parties will communicate in the upcoming rounds.

technique only works in a parameter regime that then gives rise to the $O(n^\epsilon)$ overhead obtained in [8]. To overcome this in our solution, we invoke a new technique for dealing with the case of a large p.

We quickly explain this new part on a very high level. The idea is to design a procedure that works *assuming* the exact value of p is known. This procedure is then repeated for every possible choice of p, leading to a list of possible candidates; similarly to how the scheme in [8] finds the right candidate from the list produced by the list decoding, we can then find the right one from the list. As for the procedure assuming p is known, exploiting the fact that p is large and known, we can find subsets V and V_1 so that either we can recover the shared secret from the shares of the parties in $V \cup V_1$ by standard error correction (since it happens that there is more redundancy than errors in this collection of shares), or we can argue that the complement of V is a set for which the small-p case applies and thus we can again resort to the corresponding technique in [8].

One technical novelty in our approach is that we also invoke one layer of random neighborhoods that are publicly known. In this case, the adversary can corrupt parties *depending* on who can verify whose share, but the topology of the global verification graph is fixed and cannot be modified by dishonest parties that lie about their neighborhoods.

Following [3,8], we point out that it is good enough to have a robust secret sharing scheme with a constant failure probability and a (quasi-)constant overhead; a scheme with 2^{-k} failure probability and a (quasi-) $O(k)$ overhead can then be obtained by means of parallel repetition. This is what we do here as well: at the core is a scheme where each party is given a *quasi-constant* number of bits on top of the actual share s_i (i.e., the size of the authentication keys and the size of the random neighborhoods are chosen to be quasi-constant), and we show that this scheme has a constant failure probability.

Concurrent Work. In concurrent and independent work [9], a very similar result as ours was obtained (using rather different techniques though). They also show an optimal (up to poly-logarithmic terms) robust secret sharing scheme with security against a rushing adversary. Compared to our scheme, their scheme has a slightly better poly-logarithmic dependency on n: $O(\log^2 n + \log m \log n)$. On the other hand, in a setting where the reconstruction is towards an external reconstructor R, our scheme works simply by revealing the shares to R (over multiple rounds) and R doing some local computation, whereas their scheme requires interaction *among* the shareholders and, as far as we can see, the shareholders will then learn the shared secret as well. For instance in the context of robust storage, the latter is undesirable.

2 Preliminaries

2.1 Graph Notation

We follow the graph notation in [8], which we briefly recall. Let $G = ([n], E)$ be a graph with vertex set $[n] := \{1, \ldots, n\}$ and edge set E. By convention, $(v, w) \in E$

represents an edge directed from v to w is . We let $G|_S$ be the restriction of G to S for any $S \subseteq [n]$, i.e., $G|_S = (S, E|_S)$ with $E|_S = \{(u,v) \in E : u, v \in S\}$.

For vertex $v \in [n]$, we set

$$N^{\mathsf{out}}(v) = \{w \in [n] : (v,w) \in E\} \quad \text{and} \quad N^{\mathsf{in}}(v) = \{w \in [n] : (w,v) \in E\}.$$

We use E_v as a short hand for $N^{\mathsf{out}}(v)$, the *neighborhood* of v. For $S \subseteq [n]$, we set

$$N^{\mathsf{out}}_S(v) = N^{\mathsf{out}}(v) \cap S \quad \text{and} \quad N^{\mathsf{in}}_S(v) = N^{\mathsf{in}}(v) \cap S.$$

This notation is extended to a *labeled* graph, i.e., when G comes with a function $L : E \to \{\mathtt{good}, \mathtt{bad}\}$ that labels each edge. Namely, for $v \in [n]$ we set

$$N^{\mathsf{out}}(v, \mathtt{good}) = \{w \in N^{\mathsf{out}}(v) : L(v,w) = \mathtt{good}\},$$
$$N^{\mathsf{in}}(v, \mathtt{good}) = \{w \in N^{\mathsf{in}}(v) : L(w,v) = \mathtt{good}\},$$

and similarly $N^{\mathsf{out}}(v, \mathtt{bad})$ and $N^{\mathsf{in}}(v, \mathtt{bad})$. Also, $N^{\mathsf{out}}_S(v, \mathtt{good})$, $N^{\mathsf{in}}_S(v, \mathtt{good})$, $N^{\mathsf{out}}_S(v, \mathtt{bad})$ and $N^{\mathsf{in}}_S(v, \mathtt{bad})$ are defined accordingly for $S \subseteq [n]$. Finally, we set

$$n^{\mathsf{out}}(v) = |N^{\mathsf{out}}(v)| \quad \text{and} \quad n^{\mathsf{in}}_S(v, \mathtt{bad}) = |N^{\mathsf{in}}_S(v, \mathtt{bad})|$$

and similarly for all other variations.

2.2 Random Graphs

We call a graph $G = ([n], E)$ a *randomized* graph if each edge in E is actually a random variable. We are particularly interested in randomized graphs where (some or all of) the E_v's are uniformly random and independent subsets $E_v \subset [n] \setminus \{v\}$ of a given size d. For easier terminology, we refer to such neighborhoods E_v as being *random and independent*. G is called a *random degree-d graph* if E_v is a random subset of size d in the above sense *for all* $v \in [n]$. The following properties are direct corollaries of the Chernoff-Hoeffding bound: the first follows from Chernoff-Hoeffding with independent random variables, and the latter from Chernoff-Hoeffding with negatively correlated random variables (see Appendix A).[2]

Corollary 1. *Let $G = ([n], E)$ be a randomized graph with the property that, for some fixed $v \in [n]$, the neighborhood E_v is a random subset of $[n] \setminus \{v\}$ of size d. Then, for any fixed subset $T \subset [n]$, we have*

$$\Pr\left[n^{\mathsf{out}}_T(v) \geq \mu + \Delta\right] \leq 2^{-\frac{\Delta^2}{3\mu}} \quad \text{and} \quad \Pr\left[n^{\mathsf{out}}_T(v) \leq \mu - \Delta\right] \leq 2^{-\frac{\Delta^2}{2\mu}},$$

where $\mu := \frac{|T|d}{n}$.

[2] We refer to [1] for more details, e.g., for showing that the random variables $X_j = 1$ if $j \in E_v$ and 0 otherwise are negatively correlated for E_v as in Corollary 1.

Corollary 2. *Let $G = ([n], E)$ be a randomized graph with the property that, for some fixed $T \subset [n]$, the neighborhoods E_v for $v \in T$ are random and independent of size d (in the sense as explained above). Then, for any $v \notin T$, we have*

$$\Pr\left[n_T^{\mathsf{in}}(v) \geq \mu + \Delta\right] \leq 2^{-\frac{\Delta^2}{3\mu}} \quad \text{and} \quad \Pr\left[n_T^{\mathsf{in}}(v) \leq \mu - \Delta\right] \leq 2^{-\frac{\Delta^2}{2\mu}},$$

where $\mu := \frac{|T|d}{n}$.

We will also encounter a situation where the set T may *depend* on the graph G; this will be in the context of a random but publicly known verification graph, where the adversary can then influence T dependent on G. The technical issue then is that *conditioned* on the set T, the neighborhood E_v may *not* be random anymore, so that we cannot apply the above two corollaries. Instead, we will then use the following properties, which require some more work to prove.

Lemma 1. *Let $G = ([n], E)$ be a random degree-d graph. Then, there exists no $\gamma \in \frac{1}{n}\mathbb{Z} \cap \left[0, \frac{1}{2}\right]$ and $T \subset [n]$ of size $|T| \geq (\gamma - \alpha)n$ for $\alpha^2 d = 24 \log n$ with the property that*

$$|\{v \in [n] : n_T^{\mathsf{in}}(v) < d(\gamma - 2\alpha)\}| \geq \frac{\gamma n}{2},$$

except with probability n^{1-5n}.[3]

Proof. See appendix.

Lemma 2. *Let $G = ([n], E)$ be a random degree-d graph. Then, there exists no $\gamma \in \frac{1}{n}\mathbb{Z} \cap \left[\frac{1}{\log n}, \frac{1}{2}\right]$ and $T \subset [n]$ of size $|T| \leq (\gamma - 3\alpha)n$ for $\alpha^2 d = 24 \log n$ with the property that*

$$|\{v \in [n] : n_T^{\mathsf{in}}(v) \geq d(\gamma - 2\alpha)\}| \geq \frac{\gamma n}{2},$$

except with probability n^{1-3n}.

The proof goes along the very same lines as for Lemma 1.

2.3 Robust Secret Sharing

A robust secret sharing scheme consists of two interactive protocols: the *sharing* protocol **Share** and the *reconstruction* protocol **Rec**. There are three different roles in this scheme, a *dealer* D, a *receiver* R and n parties labeled $1, \ldots, n$. The sharing protocol is executed by D and n parties: D takes as input a message **msg**, and each party $i \in \{1, \ldots, n\}$ obtains as output a *share*. Typically, D generates these shares locally and then sends to each party the corresponding share. The reconstruction protocol is executed by R and the n parties: each party is supposed to use its share as input, and the goal is that R obtains **msg**

[3] We emphasize that γ and T are allowed to depend on G.

as output. Ideally, the n parties simply send their shares to R—possibly using multiple communication rounds—and R then performs some local computation to reconstruct the message.[4]

We want a robust secret sharing scheme to be secure in the presence of an active adversary who can corrupt up to t of n parties. Once a party is corrupted, the adversary can see the share of this party. In addition, in the reconstruction protocol, the corrupt parties can arbitrarily deviate from the protocol. The following captures the formal security requirements of a robust secret sharing scheme.

Definition 1 (Robust Secret Sharing). *A pair* $(\mathbf{Share}, \mathbf{Rec})$ *of protocols is called a* (t, δ)-robust secret sharing scheme *if the following properties hold for any distribution of* **msg** *(from a given domain).*

- Privacy: *Before* **Rec** *is started, the adversary has no more information on the shared secret* **msg** *than he had before the execution of* **Share**.
- Robust reconstructability: *At the end of* **Rec**, *the reconstructor* R *outputs* $\mathbf{msg}' = \mathbf{msg}$ *except with probability at most* δ.

As for the precise corruption model, we consider an adversary that can corrupt up to t of the n parties (but not the dealer and receiver). We consider the adversary to be *rushing*, meaning that the messages sent by the corrupt parties during any communication round in the reconstruction phase may depend on the messages of the honest parties sent in that round. Also, we consider the adversary to be *adaptive*, meaning that the adversary can corrupt parties *one by one* (each one depending on the adversary's current view) and *between any two rounds of communication*, as long as the total number of corrupt parties is at most t. We point out that we do not allow the adversary to be "*corruption-rushing*", i.e., to corrupt parties *during* a communication round, depending on the messages of (some of) the honest parties in this round, and to then "rush" and modify this round's messages of the freshly corrupt parties.[5]

2.4 Additional Building Blocks

We briefly recall a couple of techniques that we use in our construction. For more details, see Appendix B.

Message Authentication Codes. The construction uses unconditionally secure message authentication codes (MAC) that satisfy the usual authentication security, but which also feature a few additional properties: (1) an authentication tag σ is computed in a *randomized* way as a function $MAC_{key}(m, r)$ of the message m, the key key, and freshly chosen randomness r, (2) it is ensured that for any ℓ keys $key_1, \ldots, key_\ell$ (with ℓ a parameter), the list of

[4] This is not the case in [9]; see our discussion at the very end of Sect. 1.

[5] It is not fully clear to us what the impact would be of such a "corruption-rushing" adversary to our scheme.

tags $MAC_{key_1}(m,r),\ldots,MAC_{key_\ell}(m,r)$ is independent of m over the choice of random string r , and (3) for any message m and fixed randomness r, the tag $MAC_{key}(m,r)$ is uniformly distributed (over the random choice of the key). The specific construction we use is polynomial-evaluation construction

$$MAC_{(x,y)} : \mathbb{F}^a \times \mathbb{F}^\ell \to \mathbb{F}, (m,r) \mapsto \sum_{i=1}^{a} m_i x^{i+\ell} + \sum_{i=1}^{\ell} r_i x^i + y,$$

with $\mathbb{F}$ a finite field of appropriate size and the key being $key = (x,y) \in \mathbb{F}^2$.

Robust Distributed Storage. Following [3,8], the tags in the construction of our robust secret sharing scheme will be stored *robustly* yet *non-privately*; the latter is the reason why the extra privacy property (2) for the MAC is necessary. This design ensures that cheaters cannot lie about the tags that authenticate their shares to, say, provoke disagreement among honest parties about the correctness of the share of a dishonest party.

Formally, a *robust distributed storage scheme* is a robust secret sharing scheme but without the privacy requirement, and it can be achieved using a list-decodable code (see Appendix B or [8] for more details). Important for us will be that the share of each party i consists of two parts, p_i and q_i, and robustness against a rushing adversary is achieved by first revealing p_i and only then, in a second communication round, q_i. Furthermore, we can do with p_i and q_i that are (asymptotically) smaller than the message by a fraction $1/n$, and with correct reconstruction except with probability $2^{-\Omega(\log^2 n)}$.

3 The Robust Secret Sharing Scheme

3.1 The Sharing Protocol

Let t be an arbitrary positive integer and $n = 2t+1$. Let $d = 600\log^3 n$.[6] We consider the message **msg** to be shared to be m bits long. We let $\mathbb{F}$ be a field with $\log|\mathbb{F}| = \log m + 3\log n$, and we set $a := \frac{m}{\log m + 3\log n}$ so that $\mathbf{msg} \in \mathbb{F}^a$. Our robust secret sharing scheme uses the following three building blocks. A linear secret sharing scheme **Sh** that corresponds to a Reed-Solomon code of length n and dimension $t+1$ over an extension field $\mathbb{K}$ over $\mathbb{F}$ with $[\mathbb{K} : \mathbb{F}] = a$,[7] together with its corresponding error-correcting decoding algorithm **Dec**, the MAC construction from Theorem 6 with $\ell = 10d$, and the robust distributed storage scheme from Theorem 7. On input $\mathbf{msg} \in \mathbb{F}^a$, our sharing protocol **Share**(**msg**) works as follows.

1. Let $(s_1,\ldots,s_n) \leftarrow \mathbf{Sh}(\mathbf{msg})$ to be the non-robust secret sharing of **msg**.
2. Sample MAC randomness $r_1,\ldots,r_n \leftarrow \mathbb{F}^{10d}$ and repeat the following 5 times.

[6] We are not trying to optimize this constant. We specify the constant 600 for the convenience of probability estimate.

[7] So that we can identify $\mathbf{msg} \in \mathbb{F}^a$ with $\mathbf{msg} \in \mathbb{K}$.

(a) For each $i \in [n]$, choose a random set $E_i \subseteq [n] \setminus \{i\}$ of size d. If there exists $j \in [n]$ with in-degree more than $2d$, do it again.[8]
(b) For each $i \in [n]$, sample a random MAC keys $key_{i,j} \in \mathbb{F}^2$ for each $j \in E_i$, and set $\mathcal{K}_i = (key_{i,j})_{j \in E_i}$.
(c) Compute the MACs[9]

$$\sigma_{i \to j} = MAC_{key_{i,j}}(s_j, r_j) \in \mathbb{F} \quad \forall j \in E_i$$

and set $\mathbf{tag}_i = (\sigma_{i \to j})_{j \in E_i} \in \mathbb{F}^d$.

Let $E_i^{(m)}$, $\mathcal{K}_i^{(m)}$ and $\mathbf{tag}_i^{(m)}$ be the resulting choices in the m-th repetition.

3. Set $\mathbf{tag} = (\mathbf{tag}_i^{(m)})_{m \in [5], i \in [n]} \in \mathbb{F}^{5nd}$, and use the robust distributed storage scheme to store $\mathbf{tag}$ *together with* $E^{(2)}$. Party i gets p_i and q_i.
4. For $i \in [n]$, define $\mathbf{s}_i = \left(s_i, E_i^{(1)}, E_i^{(3)}, E_i^{(4)}, E_i^{(5)}, \mathcal{K}_i^{(1)}, \ldots, \mathcal{K}_i^{(5)}, r_i, p_i, q_i\right)$ to be the share of party i. Output $(\mathbf{s}_1, \ldots, \mathbf{s}_n)$.

We emphasize that the topology of the graph G_2, determined by the random neighborhoods $E_i^{(2)}$, is stored robustly (yet non-private). This means that the adversary will know $G^{(2)}$ but dishonest parties cannot lie about it. For G_1, G_3, G_4, G_5 it is the other way round: they remain private until revealed (see below), but a dishonest party i can then lie about $E_i^{(m)}$.

3.2 The Reconstruction Protocol

The reconstruction protocol **Rec** works as follows. First, using 5 rounds of communication, the different parts of the shares $(\mathbf{s}_1, \ldots, \mathbf{s}_n)$ are gradually revealed to the reconstructor R:

Round 1: Every party i sends (s_i, r_i, p_i) to the reconstructor R.
Round 2: Every party i sends $(q_i, E_i^{(1)}, \mathcal{K}_i^{(1)}, \mathcal{K}_i^{(2)})$ to the reconstructor R.
Round 3: Every party i sends $(E_i^{(3)}, \mathcal{K}_i^{(3)})$ to the reconstructor R.
Round 4: Every party i sends $(E_i^{(4)}, \mathcal{K}_i^{(4)})$ to the reconstructor R.
Round 5: Every party i sends $(E_i^{(5)}, \mathcal{K}_i^{(5)})$ to the reconstructor R.

Remark 1. We emphasize that since the keys for the authentication tags are announced *after* the Shamir/Reed-Solomon shares s_i, it is ensured that the MAC does its job also in the case of a rushing adversary. Furthermore, it will be crucial that also the $E_i^{(1)}$'s are revealed in the second round only, so as to ensure that once the (correct and incorrect) Shamir shares are "on the table", the $E_i^{(1)}$'s for the honest parties are still random and independent. Similarly for the $E_i^{(m)}$'s in the m-th round for $m = 3, 4, 5$. The graph G_2 is stored robustly; hence, the adversary knows all of it but cannot lie about it.

Then, second, having received the shares of n parties, the reconstructor R locally runs the reconstruction algorithm given in the box below.

[8] This is for privacy purposes.
[9] The same randomness r_j is used for the different i's and the 5 repetitions.

Local reconstruction algorithm

Collecting the data:

1. R collects $\mathbf{s} := (s_1, \ldots, s_n)$ and $(r_1, \ldots, r_n)$, and, round by round, all the authentication keys $key_{i,j}^{(m)}$ and the graphs G_1, G_3, G_4, G_5.
2. R reconstructs all the tags $\sigma_{i \to j}^{(m)}$ and the graph G_2 from $(p_i, q_i)_{i \in [n]}$.
3. R turns $G_1, \ldots, G_5$ into *labeled* "consistency" graphs by marking any edge $(i,j) \in E^{(m)}$ as `good` for which $\sigma_{i \to j}^{(m)} = MAC_{key_{i,j}^{(m)}}(s_j, r_j)$.

Exploring the consistency graphs:

1. Estimate the number p of "passive parties" by running $\mathrm{Check}(G_1, \frac{n}{\log n})$.
2. If the output is `yes` (indicating a "large" p) then compute

$$\mathbf{c}_\gamma := \mathrm{BigP}(G_1, G_2, G_3, G_4, \gamma, \mathbf{s})$$

for every $\gamma \in \Gamma := [\frac{1}{\log n}, \frac{1}{4}] \cap \frac{1}{n}\mathbb{Z}$, set $\mathbf{c}_1 = \mathbf{Dec}(\mathbf{s})$, and output

$$\mathbf{c} := \mathrm{Cand}\big(\{\mathbf{c}_\gamma\}_{\gamma \in \Gamma} \cup \{\mathbf{c}_1\}, G_5, \mathbf{s}\big)\,.$$

3. Otherwise, i.e., if the output is `no` (indicating a "small" p), compute

$$\mathbf{c}_i = \mathrm{GraphB}\big(G_3, G_4, \tfrac{4n}{\log n}, i, \mathbf{s}\big)$$

for every $i \in [n]$, and output $\mathbf{c} := \mathrm{maj}(\mathbf{c}_1, \ldots, \mathbf{c}_n)$, the majority.

In a first step, this reconstruction algorithm considers the graphs G_1, G_2, G_3, G_4 and all the authentication information, and turns there graphs into *labeled* graphs by marking edges as `good` or `bad` depending on whether the corresponding authentication verification works out. Then, makes calls to various subroutines; we will describe and analyze them at a time. As indicated in the description of the reconstruction algorithm, the overall approach is to first find out if the number p of passive parties[10] is small or large, i.e., if there is either lots of redundancy or many errors in the Shamir shares, and then use a procedure that is tailored to that case. Basically speaking, there are three subroutines to handle p in three different ranges. The unique decoding algorithm $\mathbf{Dec}(\mathbf{s})$ handles the case $p \geq \frac{n}{4}$ where there is sufficient redundancy in the shares to *uniquely* decode (this is the trivial case, which we do not discuss any further below but assume for the remainder that $p \leq \frac{n}{4}$). The graph algorithm GraphB handles the case $p \leq \frac{4n}{\log n}$, and the algorithm BigP deals with $p \in [\frac{n}{\log n}, \frac{n}{4}]$; there is some overlap in those two ranges as will not be able to pinpoint the range precisely.

[10] Formally, p is defined as t minus the number of active parties; thus, we implicitly assume that t parties are corrupt (but some of them may behave honestly).

In order to complete the description of the reconstruction procedure and to show that it does its job (except with at most constant probability), we will show the following in the upcoming sections.

1. An algorithm Check that distinguishes "small" from "large" p.
2. An algorithm BigP that, when run with $\gamma = p/n$ and given that p is "large", outputs a valid codeword $\mathbf{c}$ for which $c_i = s_i$ for all honest i, and thus which decodes to s. Given the p is not know, this algorithm is run with all possible choices for p, and all the candidates for $\mathbf{c}$ are collected.
3. An algorithm Cand that finds the right $\mathbf{c}$ in the above list of candidates.
4. An algorithm GraphB that, when run with an *honest* party i and given that p is "small", outputs the codeword corresponding to the correct secret s. This algorithm very much coincides with the algorithm used in [8] to deal with the case of a "small" p, except for an adjustment of the parameters. We defer description of this algorithm to our appendix as the security analysis is quite similar to the graph algorithm in BigP.

3.3 "Active" and "Passive" Dishonest Parties

As in previous work on the topic, for the analysis of our scheme, it will be convenient to distinguish between corrupt parties that announce the correct Shamir share s_i and the correct randomness r_i in the first round of the reconstruction phase (but may lie about other pieces of information) and between corrupt parties that announce an incorrect s_i or r_i. Following the terminology of previous work on the topic, the former parties are called *passive* and the latter are called *active* parties, and we write P and A for the respective sets of passive and active parties, and we write H for the set of honest parties.

A subtle issue is the following. While the set A of active parties is determined and fixed after the first round of communication, the set of passive parties P may increase over time, since the adversary may keep corrupting parties as long as $|A \cup P| \leq t$, and make them lie in later rounds. Often, this change in P is no concern since many of the statements are in terms of $H \cup P$, which is fixed like A. In the other cases, we have to be explicit about the communication round we consider, and P is then understood to be the set of passive parties *during* this communication round.

3.4 The Consistency Graphs

As in [8], using a lazy sampling argument, it is not hard to see that after every communication round (including the subsequent "corruption round") in the reconstruction procedure, the following holds. Conditioned on anything that can be computed from the information announced up to that point, the neighbourhoods $E_i^{(m)}$ of the currently honest parties that are then announced in the *next* round are still random and independent. For example, conditioned on the set A of active parties and the set P of passive parties after the first round, the $E_i^{(1)}$'s announced in the second round are random and independent for all

$i \in H = [n] \setminus (A \cup P)$. Whenever we make probabilistic arguments, the randomness is drawn from these random neighbourhoods. The only exception is the graph G_2, which is robustly but non-privately stored, and which thus has the property that the $E_i^{(2)}$'s are random and independent for *all* parties, but not necessarily anymore when conditioned on, say, A and/or P.

Furthermore, by the security of the robust distributed storage of **tag** (Theorem 7) and the MAC (Theorem 6) with our choice of parameters, it is ensured that all of the labeled graphs $G_1, \ldots, G_5$ satisfy the following property except with probability $O\big(\log^3(n)/n^2\big)$. For any edge (i, j) in any of these graphs G_m, if i is honest at the time it announces $E_i^{(m)}$ then (i, j) is labled `good` whenever j is honest or passive.[11] Also, (i, j) is labeled `bad` whenever j is active.

These observations give rise to the following definition, given a partition $[n] = H \cup P \cup A$ into disjoint subsets with $|H| \geq t + 1$.

Definition 2. *A randomized labeled graph $G = ([n], E)$ is called a degree-d* consistency graph *(w.r.t. the given partition) if the following two properties hold.*

(Randomness) *The neighborhoods $E_i = \{j \,|\, (i, j) \in E\}$ of the vertices $i \in H$ are uniformly random and independent subsets of $[n] \setminus \{i\}$ of size d.*

(Labelling) *For any edge $(i, j) \in E$ with $i \in H$, if $j \in H \cup P$ then $L(i, j) =$* `good` *and if $j \in A$ then $L(i, j) =$* `bad`*.*

In order to emphasize the randomness of the neighborhoods E_i *given* the partition $[n] = H \cup P \cup A$ (and possibly of some other information X considered at a time), we also speak of a *fresh* consistency graph (w.r.t. to the partition and X). When we consider a variant of a consistency graph that is a random degree-d graph, i.e., the randomness property holds for *all* $i \in [n]$, while the partition $[n] = H \cup P \cup A$ (and possibly of some other information X considered at a time) may *depend* on the choice of the random edges, we speak of a *random but non-fresh* consistency graph.

Using this terminology, we can now capture the above remarks as follows:

Proposition 1. *The graphs G_1, G_3, G_4, G_5, as announced in the respective communication rounds, are fresh consistency graphs w.r.t. the partition $[n] = H \cup P \cup A$ given by the active and (at that time) passive parties and w.r.t. any information available to R or the adversary prior to the respective communication round, except that the labeling property may fail with probability $O\big(\log^3(n)/n^2\big)$ (independent of the randomness of the edges). On the other hand, G_2 is a random but non-fresh consistency graph (where, again, the labeling property may fail with probability $O\big(\log^3(n)/n^2\big)$).*

In the following analysis we will suppress the $O\big(\log^3(n)/n^2\big)$ failure probability for the labeling property; we will incorporate it again in the end. Also, we take it as understood that the partition $[n] = H \cup P \cup A$ always refers to the honest, the passive and the active parties, respectively.

[11] Note that we are exploiting here the fact that the authentication tags are *robustly* stored; thus, passive parties cannot lie about them.

3.5 The Check Subroutine

Let A be the set of active parties (well defined after the first communication round), and let $p := t - |A|$, the number of (potential) passive parties. The following subroutine distinguishes between $p \geq \frac{n}{\log n}$ and $p \leq \frac{4n}{\log n}$. This very subroutine was already considered and analyzed in [8]; thus, we omit the proof. The intuition is simply that the number of good outgoing edges of the honest parties reflects the number of active parties.

Check(G, ϵ)

Output `yes` if

$$|\{i \in [n] \,:\, n^{\mathsf{out}}(i, \mathtt{good}) \geq \frac{d}{2}(1+\epsilon)\}| \geq t+1\,;$$

otherwise, output `no`.

Proposition 2 [8]. *Except with probability* $\epsilon_{check} \leq 2^{-\Omega(\epsilon d)}$, Check$(G, \epsilon)$ *outputs* `yes` *if* $p \geq \epsilon n$ *and* `no` *if* $p \leq \epsilon n/4$ *(and either of the two otherwise).*

3.6 The Cand Subroutine

For simplicity, we next discuss the algorithm Cand. Recall that the set of correct Shamir sharings form a (Reed-Solomon) code with minimal distance t, and $\mathbf{s}$ collected by R is such a codeword, but with the coordinates in A (possibly) altered. The task of Cand is to find "the right" codeword $\mathbf{c}$, i.e., the one with $c_i = s_i$ for all $i \notin A$, out of a given list $\mathcal{L}$ of codewords. The algorithm is given access to a "fresh" consistency graph, i.e., one that is still random when conditioned on the list $\mathcal{L}$, and it is assumed that p is not too small.

Cand$(\mathcal{L}, G, \mathbf{s})$

For each codeword $\mathbf{c} \in \mathcal{L}$, set

$$S := \{i \in [n] \,|\, c_i = s_i\} \quad \text{and} \quad T := \{v \in S : n^{\mathsf{out}}_{[n]\setminus S}(v, \mathtt{good}) = 0\}$$

until $|T| \geq t+1$, and output $\mathbf{c}$ then.

Proposition 3. *If* $p \geq \frac{n}{\log n}$, $\mathcal{L}$ *is a set of codewords of cardinality* $O(n^2)$ *for which there exists* $\mathbf{c} \in \mathcal{L}$ *with* $c_i = s_i$ *for all* $i \in H \cup P$, *and* G *is a fresh consistency graph, then Cand*$(\mathcal{L}, G, \mathbf{s})$ *outputs this* $\mathbf{c} \in \mathcal{L}$ *except with probability* $\epsilon_{cand} \leq e^{-\Omega(\log^2 n)}$.

Proof. Consider first a codeword $\mathbf{c} \in \mathcal{L}$ for which $c_i \neq s_i$ for some $i \in H \cup P$. Then, due to the minimal distance of the code, $|(H \cup P) \cap S| \leq t$. Therefore,

$$|(H \cup P) \setminus S| \geq |H \cup P| - t > p \geq \frac{n}{\log n}.$$

By the properties of G and using Corollary 1, this implies that for any $v \in H$

$$\Pr[n^{\mathsf{out}}_{[n]\setminus S}(v, \mathtt{good}) = 0] \leq \Pr[n^{\mathsf{out}}_{(H\cup P)\setminus S}(v, \mathtt{good}) = 0] \leq 2^{-\Omega(\frac{d}{\log n})},$$

which is $2^{-\Omega(\log^2 n)}$ by the choice of d. Taking a union bound over all such $\mathbf{c} \in \mathcal{L}$ does not affect this asymptotic bound.

Next, if $\mathbf{c} \in \mathcal{L}$ with $c_i = s_i$ for all $i \in H \cup P$, i.e., $[n] \setminus S \subseteq A$, then, by the properties of G,

$$n^{\mathsf{out}}_{[n]\setminus S}(v, \mathtt{good}) \leq n^{\mathsf{out}}_{A}(v, \mathtt{good}) = 0$$

for any $v \in H \subseteq S$. This proves the claim. □

4 The Algorithm for Big p

We describe and discuss here the algorithm BigP, which is invoked when p is large. We show that BigP works, i.e., outputs a codeword $\mathbf{c}$ for which $c_i = s_i$ for all $i \in H \cup P$, and thus which decodes to the correct secret s, *if* it is given p as input. Since, p is not known, in the local reconstruction procedure BigP is run with all possible choices for p, producing a list of codewords, from which the correct one can be found by means of Cand, as shown above.

BigP$(G_1, G_2, G_3, G_4, \gamma, \mathbf{s})$

1. *Find V with no active parties and many honest parties:*

$$V := \mathrm{Filter}(G_1, \gamma)\,.$$

2. *Find a correct codeword assuming $V \cap P$ to be small or V to be large:*

$$\mathbf{c} := \mathrm{Find}(G_2, V, \gamma, \mathbf{s})\,.$$

3. *Find a list of candidate codewords otherwise:* Let $W := [n] \setminus V$ and

$$\mathbf{c}_i := \mathrm{Graph}(G_3, G_4, W, \gamma, i) \text{ for } i \in W\,.$$

4. Output $\{\mathbf{c}\} \cup \{\mathbf{c}_1, \ldots, \mathbf{c}_{|W|}\}$.

Below, we describe the different subroutines of BigP and show that they do what they are supposed to do. Formally, we will prove the following.

Theorem 1. *If the number* $p := t - |A|$ *of passive parties satisfies* $\frac{n}{\log n} \leq p \leq \frac{n}{4}$, *and* $\gamma := \frac{p}{n}$, *then BigP will output a list that contains the correct codeword except with probability* $\epsilon_{bigp} \leq O(n^{-3})$. *Moreover, it runs in time poly*(n, m).

For the upcoming description of the subroutines of BigP, we define the global constant

$$\alpha := \frac{1}{5 \log n} \qquad \text{so that} \qquad \alpha^2 d = \frac{600 \log^3 n}{25 \log^2 n} = 24 \log n.$$

Also, recall that $\frac{1}{\log n} \leq \gamma = \frac{p}{n} \leq \frac{1}{4}$.

4.1 Filter Out Active Parties

The goal of the algorithm Filter is to find a set V with no active parties and many honest parties. It has access to γ and to a fresh consistency graph.

Filter(G, γ)

Compute

$$T := \left\{ v \in [n] : n^{\mathsf{out}}_{[n]}(v, \mathtt{bad}) \leq \tfrac{d(1-2\gamma+\alpha)}{2} \right\}$$

and

$$V := \left\{ v \in T : n^{\mathsf{in}}_{T}(v, \mathtt{bad}) \leq \tfrac{d(1-\alpha)}{2} \right\}$$

and output V.

Proposition 4. *If* $\gamma = (t - |A|)/n$ *and* G *is a fresh consistency graph then Filter*(G, γ) *outputs a set* V *that satisfies*

$$|V \cap H| \geq |H| - t + (\gamma - \alpha)n \geq (\gamma - \alpha)n \qquad \text{and} \qquad V \cap A = \emptyset \tag{1}$$

except with probability $O(n^{-3})$.

We point out that the statement holds for the set of honest parties H as it is *before* Round 2 of the reconstruction procedure, but lower bound $(\gamma - \alpha)n$ will still hold *after* Round 2, since $|H|$ remains larger than t.

Proof. By the property of G and using Corollary 1, recalling that $\frac{|A|}{n} \leq \frac{1-2\gamma}{2}$, we have

$$\Pr\left[n^{\mathsf{out}}(v, \mathtt{bad}) \geq \tfrac{d(1-2\gamma+\alpha)}{2}\right] = \Pr\left[n^{\mathsf{out}}_{A}(v, \mathtt{bad}) \geq \tfrac{d(1-2\gamma+\alpha)}{2}\right] \leq 2^{-\alpha^2 d/6} = n^{-4}$$

for all $v \in H$. Taking a union bound over all honest parties, we conclude that all $v \in H$ are contained in T, except with probability n^{-3}.

In order for an honest party $v \in H$ to fail the test for being included in V, there must be $d(1 - \alpha)/2$ bad incoming edges, coming from dishonest parties

in T. However, there are at most t dishonest parties in T, each one contributing at most $d(1-2\gamma+\alpha)/2$ bad outgoing edges; thus, there are at most

$$\frac{td(1-2\gamma+\alpha)}{d(1-\alpha)} \leq t(1-2\gamma+2\alpha) = t-(\gamma-\alpha)n$$

honest parties excluded from V, where the inequality holds because

$$\frac{1-2\gamma+\alpha}{1-2\gamma+2\alpha} \leq \frac{(1-2\gamma+\alpha)+2(\gamma-\alpha)}{(1-2\gamma+2\alpha)+2(\gamma-\alpha)} = 1-\alpha,$$

using $\gamma-\alpha \geq 0$. This proves the claim on the number of honest parties in V.

For an active party $v \in A$, again by the properties of G but using Corollary 2 now, it follows that

$$\Pr\left[n_T^{\mathsf{in}}(v,\mathtt{bad}) \leq \tfrac{d(1-\alpha)}{2}\right] \leq \Pr\left[n_H^{\mathsf{in}}(v,\mathtt{bad}) \leq \tfrac{d(1-\alpha)}{2}\right] \leq 2^{-\alpha^2 d/4} = n^{-6},$$

recalling that $\frac{|H|}{n} \geq \frac{1}{2}$ and $H \subseteq T$. Taking the union bound, we conclude that V contains no active party, except with probability $O(n^{-5})$.

4.2 Find the Correct Codeword—In Some Cases

On input the set V as produced by Filter above, the goal of Find is to find the correct decoding of $\mathbf{s}$. Find is given access to γ and to a modified version of a consistency graph G. Here, the consistency graph has uniformly random neighbourhoods E_i for *all* parties, but the set V as well as the partition of $[n]$ into honest, passive and active parties may *depend* on the topology of G. Indeed, this is the property of the graph G_2, on which Find is eventually run.

Find$(G, V, \gamma, \mathbf{s})$

If $|V| < (2\gamma+2\alpha)n$ then set

$$V_1 := \left\{v \in [n] \setminus V : n_V^{\mathsf{in}}(v,\mathtt{good}) \geq d(\gamma-2\alpha)\right\},$$

while

$$V_1 := \left\{v \in [n] \setminus V : n_V^{\mathsf{in}}(v,\mathtt{good}) \geq d(\gamma+2\alpha)\right\}$$

otherwise. Then, run the unique decoding algorithm on the shares s_i for $i \in V_1 \cup V$, and output the resulting codeword $\mathbf{c}$.

We will show that the algorithm Find succeeds as long as

$$|V \cap P| \leq (\gamma-3\alpha)n \qquad \text{or} \qquad |V| \geq (2\gamma+2\alpha)n. \tag{2}$$

This condition implies that honest parties outnumber passive parties by at least $2\alpha n$ in V. We notice that $2\alpha n$ is a very narrow margin which may become useless if passive parties in V can lie about their neighbours by directing all

their outgoing edges to active parties. This behaviour may result in many active parties admitted to V_1. To prevent passive parties in V from lying about their neighbours, we introduce a non-fresh consistency graph G whose topology is publicly known but can not be modified. With the help of this graph G, we first prove that under condition (2), $V \cup V_1$ contains many honest and passive parties with high probability. Then, we further prove that under the same condition, $V \cup V_1$ contains very few active parties with high probability.

We stress that in the following statement, the partition $[n] = H \cup P \cup A$ (and thus γ) and the set V may depend on the choice of the (random) edges in G.

Lemma 3. *For $\gamma = (t - |A|)/n$, $V \subseteq [n]$ and G a random but non-fresh consistency graph, the following holds except with probability $2^{-\Omega(n)}$. If*

$$|V \cap H| \geq (\gamma - \alpha)n \qquad \text{and} \qquad |V| < (2\gamma + 2\alpha)n,$$

or

$$|V| \geq (2\gamma + 2\alpha)n,$$

then V_1 produced by Find(G, V, γ) satisfies $|(H \cup P) \setminus (V \cup V_1)| \leq \frac{\gamma n}{2}$.

Proof. Consider $T := V \cap H$. Note that for $v \in H \cup P$

$$n_T^{\mathsf{in}}(v) = n_{V \cap H}^{\mathsf{in}}(v) = n_{V \cap H}^{\mathsf{in}}(v, \mathtt{good}) \leq n_V^{\mathsf{in}}(v, \mathtt{good}),$$

and thus

$$B := \left\{v \in H \cup P : n_V^{\mathsf{in}}(v, \mathtt{good}) < d(\gamma - 3\alpha)\right\} \subseteq \left\{v \in H \cup P : n_T^{\mathsf{in}}(v) < d(\gamma - 3\alpha)\right\}.$$

By Lemma 1 the following holds, except with probability $2^{-\Omega(n)}$. If $|V \cap H| \geq (\gamma - \alpha)n$ then $|B| < \frac{\gamma n}{2}$. But also, by definition of V_1 in case $|V| < (2\gamma + 2\alpha)n$, $(H \cup P) \setminus (V \cup V_1) \subseteq B$. This proves the claim under the first assumption on V.

The proof under the second assumption goes along the same lines, noting that the lower bound on $|V|$ then implies that $|V \cap H| \geq |V| - |P| \geq (\gamma + 2\alpha)n$, offering a similar gap to the condition $n_V^{\mathsf{in}}(v, \mathtt{good}) < d(\gamma + \alpha)$ in the definition of V_1 then.

We proceed to our second claim.

Lemma 4. *For $\gamma = (t - |A|)/n$, $V \subseteq [n]$ and G a random but non-fresh consistency graph, the following holds except with probability $2^{-\Omega(n)}$. If $V \cap A = \emptyset$, as well as*

$$|V \cap P| \leq (\gamma - 3\alpha)n \qquad \text{and} \qquad |V| < (2\gamma + 2\alpha)n$$

or

$$|V| \geq (2\gamma + 2\alpha)n,$$

then V_1 produced by Find(G, V, γ) satisfies $|V_1 \cap A| \leq \frac{\gamma n}{2}$.

Proof. Consider $T := V \cap P$. Note that for $v \in A$

$$n_T^{\mathsf{in}}(v) = n_{V\cap P}^{\mathsf{in}}(v) \geq n_{V\cap P}^{\mathsf{in}}(v, \mathtt{good}) = n_V^{\mathsf{in}}(v, \mathtt{good}),$$

and thus

$$C := \left\{v \in A : n_V^{\mathsf{in}}(v, \mathtt{good}) \geq d(\gamma - 2\alpha)\right\} \subseteq \left\{v \in A : n_T^{\mathsf{in}}(v) \geq d(\gamma - 2\alpha)\right\}.$$

By Lemma 2 the following holds, except with probability $2^{-\Omega(n)}$. If $|V \cap P| \leq (\gamma - 3\alpha)n$ then $|C| < \frac{\gamma n}{2}$. But also, by definition of V_1 in case $|V| < (2\gamma + 2\alpha)n$, $V_1 \cap A \subseteq C$. This proves the claim under the first assumption on V.

The proof under the second assumption goes along the same lines, noting that $|V \cap P| \leq |P| \leq \gamma n$ offers a similar gap to the condition $n_V^{\mathsf{in}}(v, \mathtt{good}) < d(\gamma + \alpha)$ in the definition of V_1 then.

The following theorem is a consequence of Lemma 3 and Lemma 4. The statement holds for P and H after Round 2 in the reconstruction procedure.

Proposition 5. *The following holds except with probability $2^{-\Omega(n)}$. If (1) is satisfied, i.e., $|V \cap H| \geq (\gamma - \alpha)n$ and $V \cap A = \emptyset$, and additionally*

$$|V \cap P| \leq (\gamma - 3\alpha)n \qquad \text{or} \qquad |V| \geq (2\gamma + 2\alpha)n$$

holds, and if G is a non-fresh consistency graph, then Find$(G, V, \gamma, \mathbf{s})$ will output the correct codeword (determined by the s_i for $i \in H$).

Proof. It follows from Lemma 3 and Lemma 4 that, except with the claimed probability, $|(V \cup V_1) \cap A| \leq \frac{\gamma n}{2}$ and $|(V \cup V_1) \cap (P \cup H)| \geq t + 1 + \frac{\gamma n}{2}$. Therefore, the punctured codeword, obtained by restricting to the coordinates in $V \cup V_1$, has more redundancy than errors, thus unique decoding works and produces the correct codeword.

Remark 2. Given that (1), i.e., $|V \cap H| \geq (\gamma - \alpha)n$ and $V \cap A = \emptyset$, is promised to be satisfied (except with small probability), the only case when Find fails is $|V| < (2\gamma + 2\alpha)n$ yet $|V \cap P| > (\gamma - 3\alpha)n$, where P is the set of passive parties before the third communication round. These conditions together with (1) imply that

$$|V| = |V \cap H| + |V \cap P| \geq (\gamma - \alpha)n + (\gamma - 3\alpha)n = (2\gamma - 4\alpha)n$$

and

$$|V \cap H| = |V| - |V \cap P| \leq |V| - |V \cap P| \leq (2\gamma + 2\alpha)n - (\gamma - 3\alpha)n \leq (\gamma + 5\alpha)n$$

This holds for set of honest parties H even before Round 4 as the set of honest parties before Round 4 is a subset of that before Round 3. Combine this observation with Proposition 4, we come to conclusion that $|V \cap H| \in [(\gamma - \alpha)n, (\gamma + 5\alpha)n]$ holds for the set of honest parties H before Round 4, i.e., the number of honest parties within V is in the above range.

We can thus conclude that if Find fails then the set $W := [n] \setminus V$ satisfies

$$(1 - 2\gamma - 2\alpha)n \leq |W| \leq (1 - 2\gamma + 4\alpha)n$$

and, given that $|W \cap H| = t + 1 - |V \cap H|$,

$$(\frac{1}{2} - \gamma - 5\alpha)n \leq |W \cap H| \leq (\frac{1}{2} - \gamma + \alpha)n + 1.$$

As we mention before, this holds for the set of honest parties H before Round 4. Moreover,

$$|W \cap P| = |P| - |V \cap P| = |P \cup H| - |H| - |V \cap P| \leq \gamma n - (\gamma n - 3\alpha n) \leq 3\gamma n.$$

as $|P| \leq \gamma n$. We point out that the statement $|W \cap P| \leq 3\gamma n$ holds even for the set of passive parties P before Round 4 as $|P \cup H| = t + 1 + \gamma n$, $|H|$ remains bigger than $t + 1$ and $|V \cap P|$ remains bigger than $\gamma - 3\alpha n$. In the following section we show that if W satisfies the above constraints then the algorithm Graph finds the correct decoding of $\mathbf{s}$ (when initiated with an honest party v and two fresh consistency graphs).

4.3 Graph Algorithm

Recall that n'^{out}_W refers to n^{out}_W but for the graph G' rather than G, and similarly for n'^{in}_W. This graph algorithm resembles the one in [8], due to that they share the same goal of finding a subset of parties that contains all honest parties and a few dishonest parties whose majority are the passive parties. The differences lie in the range of parameters due to that the graph algorithm in this paper takes the subset of n parties as an input instead of n parties and honest parties may not be a majority in this subset.

The algorithm Graph(G, G', W, γ, v)

i. Set $X := \{v\}$.

ii. *Expand X to include more honest parties:*

$$\text{While } |X| \leq \frac{\alpha t}{2d} \text{ do } X := \text{Expan}(G, W, X, \tfrac{1}{2} - \gamma).$$

iii. *Include all honest parties into V:*

$$V := V \cup \left\{v \in W \setminus X : n_X^{\mathsf{in}}(v, \mathtt{good}) \geq \tfrac{d|X|}{2n}\right\}.$$

iv. *Remove all active parties from V (and maybe few honest parties):*

$$U := \left\{v \in X : n_X^{\mathsf{in}}(v, \mathtt{bad}) \geq \tfrac{d}{10}\right\} \quad \text{and} \quad X := X \setminus U.$$

v. 1. *Bound the degree of parties in X:*

$$V := V \setminus \left\{v \in X : n'^{\mathsf{out}}_U(v) \geq \tfrac{d}{8}\right\}.$$

2. *Include the honest parties from U (and perhaps few active parties):*

$$X := X \cup \left\{v \in U : n'^{\mathsf{in}}_V(v, \mathtt{good}) \geq \tfrac{d}{6}\right\}.$$

3. *Error correction:*
Run the unique decoding algorithm algorithm on the shares of parties in $X \cup ([n] \setminus W)$ and output the result.

In this section, we assume that the graph algorithm $\text{Graph}(G, G', W, \epsilon, v)$ starts with an honest party v. Set $c = \frac{1}{2} - \gamma$ and we have $c \in [\frac{1}{4}, \frac{1}{2}]$ as $\gamma \leq \frac{1}{4}$. Note that P and H now become the set of passive parties and honest parties before Round 3. According to Remark 2, it suffices to prove the correctness of this graph algorithm under the condition that

$$|W| \in [(2c-2\alpha)n, (2c+4\alpha)], \quad |W \cap H| \in [c-5\alpha, c+\alpha], \quad |W \cap P| \leq 3\alpha n. \quad (3)$$

Recall that $\alpha = \frac{1}{5 \log n}$ and $\alpha^2 d = 24 \log n$. We also note that by Remark 2, the above condition also holds for the set of passive parties and honest parties before Round 4. In what follows, when we claim that some event happens with high probability, we mean this holds for all set W, P and H in above range.

Let $H_W = H \cap W$ and $P_W = P \cap W$. The subset of active parties in W is still A. The out-degree of vertices in $G|_W$ and $G'|_W$ is expected to be $d\frac{|W|}{n} \in [(2c-2\alpha)d, (2c+4\alpha)d]$ and that (due to the MAC's) the edges from honest parties to active parties are labeled bad, and the edges from honest parties to honest or passive parties are labeled good.

We also recall that whether a corrupt party i is *passive* or *active*, i.e., in P or in A, depends on s_i and r_i only, as announced in the first communication round in the reconstruction phase. Note that a passive party may well lie about, say, his

neighborhood E_i. Our reasoning only relies on the neighborhoods of the honest parties, which are random and independent conditioned on the adversary's view, as explained in Proposition 1.

Theorem 2. *Under the claim of Proposition 1, and assuming that v is honest and W satisfies (3), the algorithm will output a correct codeword except with probability $\epsilon_{graph} \leq n^{-15}$. Moreover, it runs in time poly(m, n).*

The proof follows almost literally the one of [8] adjusted to the parameter regime considered here. For completeness, we provide the proof of this theorem. The proof of Theorem 2 consists of the analysis of Step ii to Step v and the Graph expansion algorithm. The analysis of Step ii to Step v is deferred to the Appendix.

4.4 Graph Expansion

We start by analyzing the expansion property of $G|_{H_W}$, the subgraph of G restricted to the set of honest parties H_W.

Lemma 5 (Expansion property of $G|_{H_W}$). *If $H' \subset H_W$ is so that $|H'| \leq \frac{\alpha|H_W|}{2d}$ and the E_v's for $v \in H'$ are still random and independent in G when given H' and H, then*

$$n_H^{\mathsf{out}}(H') := \left| \bigcup_{v \in H'} N_H^{\mathsf{out}}(v) \right| \geq (c - 7\alpha)d|H'|$$

except with probability $O(n^{-23})$.

Graph expansion algorithm Expan(G, W, X, c)

Set $X' = \emptyset$. For each vertex $v \in X$ do the following:

if $n_W^{\mathsf{out}}(v, \mathsf{good}) \leq d(c + 5\alpha)$ then $X' := X' \cup N_W^{\mathsf{out}}(v, \mathsf{good})$.

Then, output $X' \cup X$.

Proof. By Remark 2, we know that the size of H_W is at least $(c - 5\alpha)n$. By assumption on the E_i's and by Corollary 1, for any vertex $v \in H'$, $\Pr[n_{H_W}^{\mathsf{out}}(v) < (c-6\alpha)d] \leq 2^{-\alpha^2 d/2c} = O(n^{-24})$ as $\alpha^2 d = 24 \log n$ and $c \leq 1/2$. Taking the union bound, this hold for all $v \in H'$ except with probability $O(n^{-23})$. In the remainder of the proof, we may thus assume that $N_{H_W}^{\mathsf{out}}(v)$ consist of $d' := (c-6\alpha)d$ random outgoing edges.

Let $N := |H_W|$, $N' := |H'|$, and let $v_1, \ldots, v_{d'N'}$ denote the list of neighbours of all $v \in H'$, with repetition. To prove the conclusion, it suffices to bound the probability p_f that more than $\alpha d N'$ of these $d'N'$ vertices are repeated.

The probability that a vertex v_i is equal to one of $v_1, \ldots, v_{i-1}$ is at most

$$\frac{i}{N-1} \leq \frac{d'N'}{N-1} = (c-6\alpha)d \cdot \frac{\alpha|N|}{2d} \cdot \frac{1}{N-1} \leq \frac{\alpha}{4}.$$

as $c \leq \frac{1}{2}$.

Taking over all vertex sets of size $\alpha d N'$ in these $d'N'$ neighbours, the union bound shows that p_f is at most

$$\begin{aligned}&\binom{d'N'}{\alpha dN'}\left(\frac{\alpha}{4}\right)^{\alpha dN'} \leq \binom{dN'}{\alpha dN'}\left(\frac{\alpha}{4}\right)^{\alpha dN'} \leq 2^{dN'H(\alpha)+\alpha dN'(\log\alpha-2)}\\ &\leq 2^{\alpha dN'(\frac{1}{\ln 2}-2+O(\alpha))} \leq 2^{-\Omega(\alpha dN')} \leq 2^{-\Omega(\log^2 n)}.\end{aligned}$$

The first inequality is due to that $\binom{n}{k} \leq 2^{nH(\frac{k}{n})}$ and the second due to

$$H(\alpha) = -\alpha\log\alpha - (1-\alpha)\log(1-\alpha) = -\alpha\log\alpha + \frac{\alpha}{\ln 2} + O(\alpha^2)$$

for $\alpha = \frac{1}{5\log n}$ and the Taylor series $\ln(1-\alpha) = \alpha + O(\alpha^2)$.

5 Parallel Repetition

The failure probability δ of our local reconstruction scheme includes the failure probability of recovering tag ϵ_{tag}, the failure probability of labelling of consistency graph ϵ_{mac}, the failure probability of algorithm GraphB ϵ_{graph}, the failure probability of algorithm BigP ϵ_{bigP}, the failure probability of algorithm Cand ϵ_{Cand} and the failure probability of Check ϵ_{check}. Therefore, we have

$$\delta = \epsilon_{mac} + \epsilon_{tag} + \epsilon_{check} + (t+1)\epsilon_{graph} + \epsilon_{bigP} + \epsilon_{cand} = O(\frac{\log^3 n}{n^2}).$$

Note that our graph has degree $d = \Omega(\log^3 n)$ and $\mathbb{F}$ is a finite field with mn^3 elements. The total share size is $m + O(d(\log n + \log m)) = m + O(\log^4 n + \log m \log^3 n)$. We summarize our result as follows.

Theorem 3. *The scheme* (**Share**, **Rec**) *is a* $2t+1$*-party* $(t, O(\frac{\log^3 n}{n^2}))$*-robust secret sharing scheme with running time poly*(m,n) *and share size* $m + O(\log^4 n + \log m \log^3 n)$.

The error probability can be made arbitrarily small by several independent executions of (**Share**, **Rec**), except that the same Shamir shares s_i would be used in all the instances. This could be done in a same manner in [8] or [3]. We skip the details but refer interested reader to [8] or [3]. In conclusion, we obtain the following main result.

Theorem 4. *For any set of positive integers t, n, κ, m with $t < n/2$, there exists a n-party $(t, 2^{-\kappa})$-robust secret sharing scheme against rushing adversary with secret size m, share size $m + O(\kappa(\log^4 n + \log^3 n \log m))$, and running time $poly(m, n)$.*

Acknowledgments. Chen Yuan has been funded by the ERC-ADG-ALGSTRONG CRYPTO project. (no. 740972)

A Chernoff Bound

Like for [3], much of our analysis relies on the Chernoff-Hoeffding bound, and its variation to "sampling without replacement". Here and throughout, $[n]$ is a short hand for $\{1, 2, \ldots, n\}$.

Definition 3 (Negative Correlation [1]). *Let $X_1, \ldots, X_n$ be binary random variables. We say that they are negatively correlated if for all $I \subset [n]$:*

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

Theorem 5 (Chernoff-Hoeffding Bound). *Let $X_1, \ldots, X_n$ be random variables that are independent and in the range $0 \leq X_i \leq 1$, or binary and negatively correlated, and let $u = E\left[\sum_{i=1}^{n} X_i\right]$. Then, for any $0 < \delta < 1$:*

$$\Pr\left[\sum_{i=1}^{n} X_i \leq (1-\delta)u\right] \leq 2^{-\delta^2 u/2} \quad \textit{and} \quad \Pr[\left[\sum_{i=1}^{n} X_i \geq (1+\delta)u\right] \leq 2^{-\delta^2 u/3}.$$

B Building Blocks

B.1 MAC Construction

We adopt the definition as well as the construction of message authentication codes (MAC) from [8].

Definition 4. *A message authentication code (MAC) for a finite message space $\mathcal{M}$ consists of a family of functions $\{MAC_{key} : \mathcal{M} \times \mathcal{R} \to \mathcal{T}\}_{key \in \mathcal{K}}$. This MAC is said to be (ℓ, ϵ)-secure if the following three conditions hold.*

1. Authentication security: *For all $(m, r) \neq (m', r') \in \mathcal{M} \times \mathcal{R}$ and all $\sigma, \sigma' \in \mathcal{T}$,*

$$\Pr_{key \leftarrow \mathcal{K}}[MAC_{key}(m', r') = \sigma' | MAC_{key}(m, r) = \sigma] \leq \epsilon.$$

2. Privacy over Randomness: *For all $m \in \mathcal{M}$ and $key_1, \ldots, key_\ell \in \mathcal{K}$, the distribution of ℓ values $\sigma_i = MAC_{key_i}(m, r)$ is independent of m over the choice of random string $r \in \mathcal{R}$, i.e.,*

$$\Pr_{r \leftarrow \mathcal{R}}[(\sigma_1, \ldots, \sigma_\ell) = \mathbf{c} | m] = \Pr_{r \leftarrow \mathcal{R}}[(\sigma_1, \ldots, \sigma_\ell) = \mathbf{c}]$$

for any $\mathbf{c} \in \mathcal{T}^\ell$.

3. Uniformity: *For all* $(m, r) \in \mathcal{M} \times \mathcal{R}$*, the distribution of* $\sigma = MAC_{key}(m, r)$ *is uniform at random over the random element key* $\in \mathcal{K}$*.*

The following variation of the standard polynomial-evaluation MAC construction meets the requirements.

Theorem 6 (Polynomial Evaluation [8]). *Let* $\mathbb{F}$ *be a finite field. Let* $\mathcal{M} = \mathbb{F}^a$*,* $\mathcal{R} = \mathbb{F}^\ell$ *and* $\mathcal{T} = \mathbb{F}$ *such that* $\frac{a+\ell}{|\mathbb{F}|} \leq \epsilon$*. Define the family of MAC functions* $\{MAC_{(x,y)} : \mathbb{F}^a \times \mathbb{F}^\ell \to \mathbb{F}\}_{(x,y)\in\mathbb{F}^2}$ *such that*

$$MAC_{(x,y)}(\mathbf{m}, \mathbf{r}) = \sum_{i=1}^{a} m_i x^{i+\ell} + \sum_{i=1}^{\ell} r_i x^i + y$$

for all $\mathbf{m} = (m_1, \ldots, m_a) \in \mathbb{F}^a$*,* $\mathbf{r} = (r_1, \ldots, r_\ell) \in \mathbb{F}^\ell$ *and* $(x, y) \in \mathbb{F}^2$*. Then, this family of MAC functions is* (ℓ, ϵ)*-secure.*

B.2 Robust Distributed Storage

Following [3] and [8], the authentication tags in the construction of our robust secret sharing scheme will be stored *robustly* yet *non-privately*; indeed, the latter is the reason why the extra privacy property 2 in Definition 4 is necessary. The purpose of this design is to make sure that dishonest parties can not lie about the tags that authenticate their share e.g., to provoke disagreement among honest parties about the correctness of the share of a dishonest party.

Formally, a *robust distributed storage scheme* is a robust secret sharing scheme as in Definition 1 but without the privacy requirement. Such a scheme can easily be obtained as follows; we refer the interested readers to [3] or [8] for details. First of all, a list-decodable code is used to store the messages robustly. Then, the share of each party i consists of p_i and q_i, where p_i is the i-th component of list-decodable encoding of the message, and q_i is a hash-key and the hash of the message. Reconstruction works in the obvious way: a list of candidate messages is obtained by applying list decoding, and the correct message is then filtered out by means of checking the hashes. The robustness against a rushing adversary is achieved by first revealing p_i and only then, in a second communication round, q_i.

The following summarizes the result obtained by adapting the scheme in [8] to our parameter setting.

Theorem 7. *For any* $n = 2t + 1$ *and* $u = O(\log^3 n)$*, there exists a robust distributed storage against rushing adversary with messages of length* $m = \Omega(nu)$*, shares of length* $O(u)$ *that can recover the message with probability* $1 - 2^{-\Omega(\log^2 n)}$ *up to* t *corruptions.*

C Graph Algorithm for Small p

The graph algorithm GraphB that is invoked for small p is exactly the same as that in [8] (for completeness, we recall it below); GraphB is also very similar to the graph algorithm Graph appearing inside the algorithm BigP. Thus, we omit the analysis of GraphB and rely on Theorem 8 from [8], re-stated below using our terminology and instantiated with our choice of parameters. Note that n'^{out}_W refers to n^{out}_W but for the graph G' rather than G, and similarly for n'^{in}_W.

Theorem 8 (Theorem 8, [8]). *If G is a fresh consistency graph, and G' is a fresh consistency graphy w.r.t. G', and if $|P| \leq \frac{4 \log n}{n}$ and v is an honest party, then GraphB$(G, G', \epsilon, v, \mathbf{s})$ will output the correct secret except with probability $\epsilon_{graph} \leq 2^{-\Omega(d/\log^2 n)} = O(n^{-3})$. Moreover, it runs in time poly(n, m).*

The algorithm GraphB$(G, G', \epsilon, v, \mathbf{s})$ for small p

i. Input $G = ([n], E, L), G' = ([n], E', L'), d, \epsilon$ and $v \in [n]$.

ii. Expand set $V = \{v\}$ to include more honest parties:

$$\text{While } |V| \leq \frac{\epsilon t}{d} \text{ do } T = \{v \in V : n^{\mathsf{out}}(v, \mathtt{good}) \leq \frac{d}{2}(1 + 3\epsilon)\}$$

$$\text{and } V := V \cup \bigcup_{v \in T} N^{\mathsf{out}}(v, \mathtt{good}).$$

iii. Include all honest parties into V:

$$V := V \cup \left\{v \notin V : n^{\mathsf{in}}_V(v, \mathtt{good}) \geq \tfrac{d|V|}{2n}\right\}.$$

iv. Remove all active parties from V (and maybe few honest parties as well):

$$W := \left\{v \in V : n^{\mathsf{in}}_V(v, \mathtt{bad}) \geq \tfrac{d}{4}\right\} \quad \text{and} \quad V := V \setminus W.$$

v. 1. Bound the degree of parties in V:

$$V := V \setminus \left\{v \in V : n'^{\mathsf{out}}_W(v) \geq \tfrac{d}{8}\right\}.$$

2. Include the honest parties from W (and perhaps few active parties):

$$V := V \cup \left\{v \in W : n'^{\mathsf{in}}_V(v, \mathtt{good}) \geq \tfrac{d}{4}\right\}.$$

3. Error correction: run the unique decoding algorithm algorithm on the shares s_i of parties in V and output the result.

D Proof of Lemma 1

For *fixed* $\gamma \in \frac{1}{n}\mathbb{Z} \cap \left[0, \frac{1}{2}\right]$ and $T \subseteq [n]$ with $|T| = (\gamma - \alpha)n$, by Corollary 2,

$$\Pr[n_T^{\mathsf{in}}(v) < d(\gamma - 2\alpha)] \leq 2^{-\frac{\alpha^2 d}{2\gamma}}.$$

Thus, setting $\Sigma(T) := \{v \in [n] : n_T^{\mathsf{in}}(v) < d(\gamma - 2\alpha)\}$ and considering another arbitrary but fixed subset $S \subseteq [n]$, we have (by negative correlation)

$$\Pr[S \subseteq \Sigma(T)] = \Pr[n_T^{\mathsf{in}}(v) < d(\gamma - 2\alpha)\, \forall\, v \in S] \leq 2^{-\frac{\alpha^2 d}{2\gamma}|S|}.$$

Therefore, noting that $\Sigma(T) \subseteq \Sigma(T')$ for $T' \subseteq T$,

$$\begin{aligned}
&\Pr\left[\exists\, \gamma, T : |T| \geq (\gamma - \alpha)n \wedge |\Sigma(T)| \geq \tfrac{\gamma n}{2}\right] \\
&\quad = \Pr\left[\exists\, \gamma, T : |T| = (\gamma - \alpha)n \wedge |\Sigma(T)| \geq \tfrac{\gamma n}{2}\right] \\
&\quad = \Pr\left[\exists\, \gamma, T, S : |T| = (\gamma - \alpha)n \wedge |S| = \tfrac{\gamma n}{2} \wedge S \subseteq \Sigma(T)\right].
\end{aligned}$$

Taking union bound by summing over all $\gamma \in \frac{1}{n}\mathbb{Z} \cap \left[0, \frac{1}{2}\right]$ and $T, S \subseteq [n]$ with $|T| = (\gamma - \alpha)n \wedge |S| = \frac{\gamma n}{2}$, this is bounded by

$$\begin{aligned}
&\leq \sum_{\gamma} \sum_{S,T} \Pr[S \subseteq \Sigma(T)] \leq \sum_{\gamma} \binom{n}{\frac{\gamma n}{2}} \binom{n}{(\gamma - \alpha)n} 2^{-\frac{\alpha^2 d}{4} n} \\
&\leq \sum_{\gamma} n^{\frac{\gamma n}{2}} \cdot n^{\gamma n} \cdot n^{-6n} \leq n^{1-5n},
\end{aligned}$$

proving the claim. □

E Analysis of Step ii to Step v

E.1 Analysis of Step ii

Set $\epsilon = \frac{1}{\log\log n}$ and recall that $\alpha = \frac{1}{5 \log n}$. The following shows that after Step ii, at most an $O(\epsilon)$-fraction of the parties in X is dishonest.

Proposition 6. *At the end of Step ii, with probability at least* $1 - O(n^{-15})$, *X is a set of size $\Omega(\epsilon n)$ with $|H_W \cap X| \geq (1 - O(\epsilon))|X|$ and $|(P \cup A) \cap X| \leq O(\epsilon |X|)$.*

Proof. First of all, we observe that for every honest party v, the number of its good outgoing edges is expected to be $\frac{|P_W \cup H_W| d}{n} \leq (c + 4\alpha)d$ since only honest parties and passive parties can pass the verification of v. By assumption on the E_v's and by Corollary 1, we have

$$\Pr[n_W^{\mathsf{out}}(v, \mathtt{good}) \geq d(c + 5\alpha)] \leq 2^{\alpha^2 d/3c} = O(n^{-16}).$$

Taking an union bound over all $v \in H_W$ leads to the claim that except with probability $O(n^{-15})$, all honest parties in X will be included in the expansion of $\mathrm{Expan}(G, W, X, c)$.

Recall that $\mathrm{Expan}(G, W, X, c)$ has been invoked multiple times. Let X_i be the set X after Expan has been invoked i times, $X_0 = \{v\}$, $X_1 = \mathrm{Expan}(G, W, X_0, c)$ etc., and let $H_0 = \{v\}$ and $H_1 = \mathrm{Expan}(G, W, H_0, c) \cap H$, etc. be the corresponding sets when we include only honest parties into the sets.

Using a similar lazy-sampling argument as for Proposition 1, it follows that conditioned on $H_0, H_1, \ldots, H_i$, the E_v's for $v \in H_i \setminus H_{i-1}$ are random and independent for any i. Therefore, we can apply Lemma 5 to $H'_i = H_i \setminus H_{i-1}$ to obtain that $|H_{i+1}| \geq |H'_i| d(c - 7\alpha)$. It follows that $|H_i| \geq (d(1 - 7\alpha))^i$ except with probability $O(n^{-23})$. Our algorithm jumps out of Step ii when X is of size $\Omega(\alpha n)$. We next bound the number of rounds in this step. For $i = \frac{\log n}{\log \log n}$, noting that $d \geq \log^3 n$ and $c \geq \frac{1}{4}$, it thus follows that

$$|X_i| \geq |H_i| \geq \Big(d(c - 7\alpha)\Big)^i \geq (\log^3 n)^{\frac{\log n}{\log \log n}} (c - 7\alpha)^{\frac{\log n}{\log \log n}}$$
$$\geq n^3 \cdot c^{\frac{\log n}{\log \log n}} \geq \Omega(n^3).$$

This means $\mathrm{Expan}(G, W, X, c)$ is invoked $r \leq \frac{\log n}{\log \log n}$ times assuming n is large enough.

On the other hand, we trivially have $|X_r| \leq (d(c + 5\alpha))^r$ by specification of Expan. Thus,

$$|X_r| - |H_r| \leq \Big(d(c + 5\alpha)\Big)^r - \Big(d(c - 7\alpha)\Big)^r$$
$$= 12\alpha d \left(\sum_{i=0}^{r-1} \big((d(c + 5\alpha)\big)^i \big(d(c - 7\alpha)\big)^{r-1-i} \right) \leq 12\alpha r d \Big((d(c + 5\alpha)\Big)^{r-1}$$
$$= O(\frac{1}{\log \log n} |X_r|) = O(\epsilon |X_r|),$$

as $\epsilon = \frac{1}{\log \log n}$. The first equality is due to $a^n - b^n = (a - b)(\sum_{i=0}^{n-1} a^i b^{n-1-i})$ and the last one is due to $r \leq \frac{\log n}{\log \log n}$ and $\alpha = \frac{1}{5 \log n}$.

This upper bound implies that there are at least $|X_r|(1 - O(\epsilon))$ honest parties in X_r while the number of dishonest parties is at most $O(\epsilon |X_r|)$.

E.2 Analysis of Step iii

The analysis of Step iii is based on the intuition that every honest party v outside $H_W \setminus X$ will get sufficient support from parties in X as X consists almost entirely of honest parties in H_W. In particular, any such v is expected have have close to $\frac{d}{n}|X|$ good incoming edges from the parties in X.

Proposition 7. *At the end of Step iii, with probability at least* $1 - 2^{-\Omega(\alpha d)}$, X *contains* all *honest parties in* H_W *and* $O(\epsilon n)$ *dishonest parties.*

Proof. Recall that conditioned on H_r, the E_v's for $v \in H_r \setminus H_{r-1}$ are random and independent.

Set $\tilde{H} := H_r \setminus H_{r-1}$ and $d_1 := \frac{|X|d}{n} = \Omega(\alpha d)$. This implies $|\tilde{H}| = (1 - o(1))|H_r| = (1 - o(1))|X|$ as $H_{r-1} = o(H_r)$ and X contains at most $O(\epsilon|X|) = O(\frac{|X|}{\log\log n})$ dishonest parties. Using Corollary 2 for the final bound, it follows that for a given honest party $v \in H_W/\tilde{H}$,

$$\Pr\left[n_X^{\mathsf{in}}(v, \mathtt{good}) < \frac{d_1}{2}\right] \leq \Pr\left[n_{\tilde{H}}^{\mathsf{in}}(v, \mathtt{good}) < \frac{d_1}{2}\right]$$
$$= \Pr\left[n_{\tilde{H}}^{\mathsf{in}}(v) < \frac{d_1}{2}\right] \leq 2^{-\Omega(\alpha d)}.$$

By union bound over all honest parties in $H_W \setminus \tilde{H}$, all these honest parties are added to X except with probability at most $2^{-\Omega(\alpha d)}$.

On the other hand, Admitting any active party w outside X requires at least $\frac{d_1}{2}$ good incoming edges. Recall that only dishonest party can verify active party. Therefore, these edges must be directed from dishonest parties in X. Since there are at most $O(\epsilon)|X|$ dishonest parties in X and each of them contributes to at most d good incoming edges, the number of active parties admitted to X is at most $\frac{O(\epsilon)|X|d}{d_1/2} = O(\epsilon n) = O(\frac{n}{\log\log n})$.

E.3 Analysis of Step iv

The goal of Step iv is to remove all active parties from X. This can be done by exploiting the fact that the vast majority of X are honest parties.

Proposition 8. *At the end of Step iv, with probability at least $1 - 2^{-\Omega(d)}$, X consists of $|H_W| - O(\epsilon n)$ honest parties and no active parties, and U consists of the rest of honest parties in H_W and $O(\epsilon n)$ dishonest parties.*

Proof. Observe that $\frac{|H_W|d}{n} \geq (c - 5\alpha)d \geq \frac{d}{5}$ as $c \geq \frac{1}{4}$ and $\alpha = \frac{1}{5\log n}$. It follows, again using Corollary 2, that for an active party v in X, we have

$$\Pr\left[n_X^{\mathsf{in}}(v, \mathtt{bad}) < \frac{d}{10}\right] \leq \Pr\left[n_{H_W}^{\mathsf{in}}(v, \mathtt{bad}) < \frac{d}{10}\right] = \Pr\left[n_{H_W}^{\mathsf{in}}(v) < \frac{d}{10}\right] \leq 2^{-\Omega(d)}.$$

By union bound over all active parties in X, all of them are removed from X with probability at least $1 - t2^{-\Omega(d)} = 1 - 2^{-\Omega(d)}$.

On the other hand, if the honest party v is removed from X, v must receive at least $\frac{d}{10}$ bad incoming edges from dishonest parties in X. Since the number of dishonest parties is at most $a := O(\epsilon n)$, there are at most $\frac{ad}{d/10} = O(\epsilon n)$ honest parties removed from X.

E.4 Analysis of Step v

From the analysis of Step iv, we learn that $|U| = O(\epsilon n)$ and X has size $|H_W| - O(\epsilon n) = cn - O(\epsilon n)$. In order to analyze this step, we introduce the following notation. Let P and H be the set of passive parties and honest parties before the

fourth communication round respectively. According to Remark 2, the condition (3) still holds. Then, $H_W = H \cap W$ has size $cn - O(\alpha n)$ and $H_P = P \cap W$ has size at most $3\alpha n$. By Proposition 8, this implies that X consists of $cn - O(\epsilon n)$ honest parties and a few passive parties and U is as set of size $O(\epsilon)$ which contains all honest parties in $H_W \setminus X$. Let $X = X_H \cup X_P$ where X_H is a of honest parties and X_P is a set of passive parties. Let $U = U_H \cup U_C$ where U_H is a set of honest parties and U_C is a set of dishonest parties. Then, we have $X_H \cup U_H = H_W$ and $|U_H| \leq |U| = O(\epsilon n)$ as U and X together contains all honest parties in H_W.

Note that in Step v, we introduce a new fresh consistency graph G'. Given the adversary's strategy, all the previous steps of the graph algorithm are determined by the graph G and thus independent of the new fresh consistency graph G'. Therefore, by Proposition 1, at this point in the algorithm the E'_v's for $v \in H_W$ are still random and independent conditioned on X_H, X_P, U_C, U_H.

Proposition 9. *Except with probability $2^{-\Omega(d)}$, after Step v the set X will contain all honest parties in H_W and at least as many passive parties as active ones. Therefore, Step v will output the correct codeword with probability at least $1 - 2^{-\Omega(d)}$.*

Proof. **Step v.1.** For any $i \in X_H$, ${n'_U}^{\mathsf{out}}(i)$ is expected to be $\frac{|U|d}{n} = O(\epsilon d)$. By Corollary 1 we thus have

$$\Pr\left[{n'_U}^{\mathsf{out}}(i) \geq \frac{d}{8}\right] \leq 2^{-\Omega(d)}.$$

Hence, by union bound, all honest parties in X remain in X except with probability $2^{-\Omega(d)}$.

Let X'_P be the set of passive parties left in X after this step, and set $p := |X'_P|$. Note that $n_U^{\mathsf{out}}(v) \leq d/8$ for every $v \in X$.

Step v.2. Observe that $\frac{d|X_H|}{n} = (c - O(\epsilon))d \geq (\frac{1}{4} - O(\epsilon))d$ as $d \leq \frac{1}{4}$. It follows from Corollary 2 that for any honest party $i \in U_H$,

$$\Pr\left[{n'_X}^{\mathsf{in}}(i, \mathtt{good}) \leq \frac{d}{6}\right] \leq \Pr\left[{n'_{X_H}}^{\mathsf{in}}(i, \mathtt{good}) \leq \frac{d}{6}\right] = \Pr\left[{n'_{X_H}}^{\mathsf{in}}(i) \leq \frac{d}{6}\right] \leq 2^{-\Omega(d)}.$$

Thus, all honest parties in U are added to X, except with probability $2^{-\Omega(d)}$.

On the other hand, the good incoming edges of the active parties must be directed from passive parties in X'_P. Observe that each party in X is allowed to have at most $\frac{d}{8}$ outgoing neighbours in U. This implies there are at most $\frac{|X_P|d/8}{d/6} = \frac{3|X_P|}{4}$ active parties admitted to X in this step, proving the first part of the statement.

Step v.3. Observe that the set $[n] \setminus W$ consists of the rest of honest parties and passive parties. This implies that $([n] \setminus W) \cup X$ contains all honest parties and is a set where the number of active parties is less than the number of passive parties. The shares of the parties in $([n] \setminus W) \cup X$ form a Reed-Solomon code. Since the number of errors is less than the redundancy of this code, the unique decoding algorithm will output a correct codeword.

References

1. Auger, A., Doerr, B.: Theory of Randomized Search Heuristics. World Scientific, Singapore (2011)
2. Bishop, A., Pastro, V.: Robust secret sharing schemes against local adversaries. In: Cheng, C.-M., Chung, K.-M., Persiano, G., Yang, B.-Y. (eds.) PKC 2016. LNCS, vol. 9615, pp. 327–356. Springer, Heidelberg (2016). https://doi.org/10.1007/978-3-662-49387-8_13
3. Bishop, A., Pastro, V., Rajaraman, R., Wichs, D.: Essentially optimal robust secret sharing with maximal corruptions. In: Fischlin, M., Coron, J.-S. (eds.) EUROCRYPT 2016. LNCS, vol. 9665, pp. 58–86. Springer, Heidelberg (2016). https://doi.org/10.1007/978-3-662-49890-3_3
4. Carpentieri, M., De Santis, A., Vaccaro, U.: Size of shares and probability of cheating in threshold schemes. In: Helleseth, T. (ed.) EUROCRYPT 1993. LNCS, vol. 765, pp. 118–125. Springer, Heidelberg (1994). https://doi.org/10.1007/3-540-48285-7_10
5. Cevallos, A., Fehr, S., Ostrovsky, R., Rabani, Y.: Unconditionally-secure robust secret sharing with compact shares. In: Pointcheval, D., Johansson, T. (eds.) EUROCRYPT 2012. LNCS, vol. 7237, pp. 195–208. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-29011-4_13
6. Cramer, R., Damgård, I., Fehr, S.: On the cost of reconstructing a secret, or VSS with optimal reconstruction phase. In: Kilian, J. (ed.) CRYPTO 2001. LNCS, vol. 2139, pp. 503–523. Springer, Heidelberg (2001). https://doi.org/10.1007/3-540-44647-8_30
7. Cramer, R., Damgård, I.B., Döttling, N., Fehr, S., Spini, G.: Linear secret sharing schemes from error correcting codes and universal hash functions. In: Oswald, E., Fischlin, M. (eds.) EUROCRYPT 2015. LNCS, vol. 9057, pp. 313–336. Springer, Heidelberg (2015). https://doi.org/10.1007/978-3-662-46803-6_11
8. Fehr, S., Yuan, C.: Towards optimal robust secret sharing with security against a rushing adversary. In: Ishai, Y., Rijmen, V. (eds.) EUROCRYPT 2019. LNCS, vol. 11478, pp. 472–499. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-17659-4_16
9. Manurangsi, P., Srinivasan, A., Vasudevan, P.N.: Nearly optimal robust secret sharing against rushing adversaries. In: Micciancio, D., Ristenpart, T. (eds.) CRYPTO 2020. LNCS, vol. 12172, pp. 156–185. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-56877-1_6
10. Rabin, T., Ben-Or, M.: Verifiable secret sharing and multiparty protocols with honest majority (extended abstract). In: Proceedings of the 21st Annual ACM Symposium on Theory of Computing, 14–17 May 1989, Seattle, Washigton, USA, pp. 73–85 (1989)