Unconditionally-Secure Robust Secret Sharing with Compact Shares

Alfonso Cevallos[1], Serge Fehr[2⋆], Rafail Ostrovsky[3], and Yuval Rabani[4⋆]

[1] Mathematical Institute, Leiden University, The Netherlands
alfonsoc@gmail.com
[2] Centrum Wiskunde & Informatica (CWI), Amsterdam, The Netherlands
serge.fehr@cwi.nl
[3] Department of Computer Science, Department of Mathematics, UCLA
rafail@cs.ucla.edu
[4] The Rachel and Selim Benin School of Computer Science and Engineering, The Hebrew University of Jerusalem, Jerusalem 91904, Israel.
yrabani@cs.huji.ac.il

Abstract. We consider the problem of reconstructing a shared secret in the presence of faulty shares, with unconditional security. We require that any t shares give no information on the shared secret, and reconstruction is possible even if up to t out of the n shares are incorrect. The interesting setting is $n/3 \leq t < n/2$, where reconstruction of a shared secret in the presence of faulty shares is possible, but only with an increase in the share size, and only if one admits a small failure probability. The goal of this work is to minimize this overhead in the share size. Known schemes either have a $\Omega(\kappa n)$-overhead in share size, where κ is the security parameter, or they have a close-to-optimal overhead of order $O(\kappa + n)$ but have an exponential running time (in n).

In this paper, we propose a new scheme that has a close-to-optimal overhead in the share size of order $\tilde{O}(\kappa + n)$, *and* a polynomial running time. Interestingly, the shares in our new scheme are prepared in the very same way as in the well-known scheme by Rabin and Ben-Or, which relies on message authentication, but we use a message authentication code with *short* tags and keys and with correspondingly *weak* security. The short tags and keys give us the required saving in the share size. Surprisingly, we can compensate for the weakened security of the authentication and achieve an exponentially small (in κ) failure probability by means of a more sophisticated reconstruction procedure.

1 Introduction

Background. *Secret sharing*, invented independently by Shamir [18] and Blakley [2] in 1979, is a fundamental cryptographic primitive that has found numerous applications. In its basic form, it permits a dealer to share a secret s among a set of n players in such a way that: (1) up to t of the players learn no information

⋆ Part of this research was done while visiting UCLA.

on s by means of their shares, and (2) any $t+1$ of the players can (efficiently) recover s from their shares. The most famous example, Shamir's secret sharing scheme, works by choosing a random polynomial $f(X) \in \mathbb{F}[X]$ of degree at most t with s as constant coefficient (assuming that s comes from a finite field $\mathbb{F}$), and the n shares are computed as $s_1 = f(x_1), \ldots, s_n = f(x_n)$ for publicly known pairwise-distinct non-vanishing interpolation points $x_1, \ldots, x_n$. Properties (1) and (2) follow easily from Lagrange's interpolation theorem.

In its basic form, secret sharing assumes the players to be honest and to provide the correct shares when asked. However, in cryptographic scenarios we often want/need to protect against malicious behavior of the participants. Therefore, strengthened versions of secret sharing have been proposed and studied over the years. One natural strengthening is to require that the shared secret can be recovered even if some players hand in incorrect shares. This is sometimes referred to as *robust* secret sharing. Formally, it is required that if all the n players pool together their shares, but up to t of them are incorrect (and it is not known which ones), then the shared secret can still be reconstructed (except maybe with small probability). Robust secret sharing has direct applications to *secure storage* and *unconditionally secure message transmission*. The goal of secure storage is to outsource the storing of sensitive data to a group of servers, in such a way that any coalition of up to t dishonest servers does not compromise the privacy nor the retrievability of the data. In unconditionally secure message transmission, as introduced in [8], (for follow-up works, see [9, 10]) a sender wants to send some message to a receiver via a communication network that consists of n wires of which up to t may be under the control of an adversary, and privacy and receipt of the message should be guaranteed. It is immediate that "good" robust secret sharing schemes lead to "good" secure storage and "good" secure message transmission schemes. Furthermore, robust secret sharing schemes may act as stepping stone towards secret sharing schemes with yet stronger security guarantees. For instance, a *verifiable* secret sharing (VSS) scheme, as introduced in [4], additionally protects against a possibly malicious dealer who hands out inconsistent shares.

It follows immediately from the theory of Reed-Solomon error correcting codes that Shamir's secret sharing scheme is robust if (and only if) $t < n/3$. On the other hand, it is easy to see that robust secret sharing is impossible if $t \geq n/2$, and alternative definitions are needed [12]. Therefore, in this paper, we consider the range $n/3 \leq t < n/2$. In this range, robust secret sharing is possible, but only if one admits a small but positive failure probability. What makes robust secret sharing in the range $n/3 \leq t < n/2$ tricky is the fact that, say, Shamir shares alone do not carry enough redundancy to recover the correct secret in the presence of faulty shares, not even in principle. Indeed, if $n = 2t+1$, and $s_1, \ldots, s_n$ are Shamir shares of a secret s but t of the shares are incorrect, then *any* $t+1$ of the shares lie on a degree t polynomial and thus could actually be the $t+1$ correct shares. There is no way to reconstruct the correct secret s from the list of partly modified Shamir shares. Additional redundancy needs to be added to the shares to permit reconstruction in the presence of incorrect

shares. In the computational setting, this can be done by means of *commitments* (as e.g. in [16]); however, we aim for *unconditional security*, i.e., we do not put any computational restrictions on the adversary.

In this paper, we address the question of *how much* redundancy needs to be added to the shares in order to obtain robustness, for the maximal possible value of t, i.e., when $n = 2t+1$. In other words, how small can the shares be in robust secret sharing?

KNOWN SCHEMES. Interestingly, rather little is known about robust secret sharing with unconditional security for the range $n/3 \leq t < n/2$. To the best of our knowledge, up to small modifications, there exist two known (classes of) robust secret sharing schemes for this setting; we briefly discuss them here.

The first one is due to Rabin and BenOr [17] (called "*secret sharing when the dealer is a knight*" there). The Rabin-BenOr scheme consists of standard Shamir secret sharing, but enhanced my means of an (unconditionally secure) message authentication code. Specifically, for every pair of players P_i and P_j, P_i's Shamir share s_i is authenticated with an authentication tag τ_{ij}, where the corresponding authentication key key_{ji} is given to player P_j. During reconstruction, every player P_j can then verify the correctness of all the shares with the help of his authentication keys (and he detects every incorrect share except with small probability). If the reconstruction is performed by an outside reconstructor R that has no authentication keys, the reconstruction is slightly trickier. Every share s_i is then declared to be correct and used for reconstructing the secret by means of Lagrange interpolation if and only if it is accepted by the authentication keys of at least $t+1$ players.

In order for this scheme to have a failure probability (in reconstructing the correct secret) of $2^{-\kappa}$, the message authentication code must have a failure probability smaller than $2^{-\kappa}$, which means that keys and tags must be of bitsize at least κ. As a consequence, beyond the actual Shamir share, every player gets another $\Omega(n\kappa)$ bits of redundancy as part of his share.[5]

The other scheme was first pointed out by Cramer, Damgård and Fehr [5], based on an idea by [3]. This scheme works as follows. Using standard Shamir secret sharing, the dealer shares independently the actual secret $s \in \mathbb{F}$, a randomly chosen field element $r \in \mathbb{F}$, and its product $p = s \cdot r$. To reconstruct the secret, the reconstructor does the following *for every subset* of $t+1$ players. He reconstructs s', r' and p', supposed to be s, r and p, respectively, using the (possibly partly incorrect) shares of these $t+1$ players, checks if $s' \cdot r' = p'$, and halts and outputs s' if it is the case. One can show that for any subset of $t+1$ players: if $s' \neq s$ then $s' \cdot r' \neq p'$ except with probability $1/|\mathbb{F}|$. Thus, taking into account union bound over all subsets of size $t+1$, choosing $\mathbb{F}$ to be of cardinality $2^{\kappa+n}$ gives a robust secret sharing scheme with failure probability $2^{-\kappa}$ and shares of size $O(\kappa + n)$.

[5] There are some additional log terms that we ignore, for instance due to applying union bound over the players.

Hence, much less redundancy is added to the actual share than in the Rabin-BenOr scheme.[6] Furthermore, it is not too hard to see that an increase in share size of κ bits is *necessary* for robust reconstruction (with $t < n/2$); thus, this scheme has close-to-optimal share size (at least if n is of order κ). The obvious downside of the scheme is that the reconstruction has exponential (in n) running time, as it loops over all possible subsets of size $t + 1$. Up to now, it is not known if there is an *efficient* reconstruction procedure for this robust secret sharing scheme. Another drawback of this scheme is that it is insecure in case the dishonest players get to see the shares (of r) of the honest players *before* they have to submit their own shares. Thus, it cannot be used, say, if reconstruction is performed by the (partly corrupted) players, and the adversary is *rushing*, meaning that the corrupt players wait with announcing their shares and then rush to announce their (correspondingly modified) shares.[7]

The latter scheme can be understood as being obtained, in a generic way, from a secret sharing scheme that allows error *detection*, i.e., that detects if a set of $t + 1$ shares contains some incorrect ones (but can not necessarily tell which ones). Indeed, as rigorously analyzed in [14], any secret sharing scheme with error detection (as in [20, 3, 15]) can be transformed into a robust secret sharing scheme by looping over all sets of size $t + 1$; but of course, any such scheme will suffer from the same exponential running time. For dishonest majority, a notion of *identifiable* secret sharing is explored in [12].

Our Contribution. We propose a new robust secret sharing scheme that combines the advantages of both the above schemes. Our new scheme has a similar overhead in share size as the scheme by Cramer *et al.*, i.e., of the order $\tilde{O}(\kappa + n)$ rather than $\Omega(\kappa n)$, yet it is *computationally efficient*, meaning that sharing and reconstruction run in polynomial time in n, κ and the bitsize of the secret. Furthermore, security is preserved when reconstruction takes place among the partly corrupted players and the adversary is rushing.

Maybe somewhat surprisingly, the sharing procedure of our new robust secret sharing scheme is identical to that of the Rabin-BenOr scheme, except that we use a message authentication code with *short* keys and tags and with correspondingly *weak* security. In order to compensate for that, we need a more sophisticated reconstruction procedure, which inspects the *acceptance graph*, which describes which share is accepted by which player's key, more carefully. Essentially, the idea is to accept a share as being correct not as soon as it is correctly verified by $t+1$ players (as is the case in Rabin-BenOr), but only if it is correctly verified by $t + 1$ players *that hold accepted shares*. In other words, once a share is declared incorrect, then this player's vote is not counted anymore, making it harder for

[6] If the bitsize of the secret is much bigger than κ, then one can employ an adaptation of the scheme for which r and p can still live in a field of order $2^{\kappa+n}$, and thus the redundancy added to the actual share of s remains $O(\kappa + n)$ (see [6]).

[7] We stress that the Rabin-BenOr scheme does not suffer from this when the reconstruction is done in *two* rounds, where the players first announce their shares and tags, and only once everyone has revealed their shares and tags, then the keys are revealed. Looking ahead, the same will hold for our new scheme.

other incorrect shares to be accepted. In order to take care of a few incorrect shares that might survive, Reed-Solomon error correction is applied to the set of accepted shares. As will be seen later, although the basic idea of the new scheme is rather simple, its analysis is not. What makes the analysis tricky is that the probability of a bad share being detected now depends on how many other bad shares are detected. Thus, we cannot analyze the bad shares independently.

Interestingly, in [5] Cramer *et al.* prove a lower bound of $\Omega(\kappa n)$ on the necessary redundancy in the shares necessary to reconstruct a shared secret in the presence of up to a minority of incorrect shares. The discrepancy to our positive result stems from the fact that they consider a slightly stronger notion of robust secret sharing (which they called "*single-round honest-dealer VSS*" there): the reconstruction procedure must produce the correct secret except with probability $2^{-\kappa}$, but if it fails then it must output "`failure`". Thus, in their definition, reconstructing an incorrect secret is strictly prohibited, whereas we allow reconstruction of an incorrect secret with negligible probability. Also, they assume that reconstruction is done by the players with *one* round of communication and then each player deciding locally (possibly based on some part of his share he did not announce) on the reconstructed secret. Our new scheme does not seem to fit into this model since its security crucially relies on the fact that players release their shares in two rounds.

2 Preliminaries

2.1 Robust Secret Sharing

In order to define the robustness property of a secret sharing scheme, we formalize the latter by means of two interactive protocols, Share and Rec, where Share involves a *dealer* D and n players $P_1, \ldots, P_n$, and Rec involves the n players and a *reconstructor* R. More formally, an *n-player secret sharing scheme* for a message space $\mathcal{S}$ consists of two phases, the *sharing* and the *reconstruction* phase, specified by two protocols Share and Rec. During the sharing phase, the dealer D takes as input a secret $s \in \mathcal{S}$, locally computes shares $\sigma_1, \ldots, \sigma_n$, and sends the i-th share σ_i to player P_i for every $i \in [n]$. During reconstruction, player P_i (for every $i \in [n]$) communicates, possibly by means of several synchronous communication rounds, σ_i to the reconstructor R. Based on the received shares, R then produces an output s', which is supposed to be the original secret s.

Before we formalize the security requirements, we specify the capabilities (and limitations) of the *adversary* that tries to break the scheme. During the sharing phase, the adversary remains inactive, and he does not get to learn any information at all. In particular, he does not get to see the shares that D sends to the players. After the sharing phase, the adversary can adaptively corrupt up to t of the players P_i (but not D), where t is some parameter.[8] Once a player P_i is corrupted, the adversary learns P_i's share σ_i, and from now on,

[8] Since the sharing phase only involves one round of communication from D to the players, it does not help the adversary to corrupt players *during* the sharing phase.

the adversary has full control over P_i. The corruptions being adaptive means that after each corruption, the adversary can decide on who to corrupt next depending on the shares he has seen so far. During the reconstruction phase, the adversary gets to see the communication between *all* players P_i and the reconstructor R.[9] Furthermore, he controls the information that the dishonest players send to R. Namely, in every communication round, he can decide for every dishonest player on what this player should send to R, depending on what he has seen so far and depending on what the honest players have sent to R in the current round. The latter means that the adversary is *rushing*. Finally, if he has not yet corrupted t players, he can between each round of communication adaptively corrupt additional players, as long as the total number of corrupt players does not exceed t. We stress that the adversary cannot corrupt D or R.

Definition 2.1. *An n-player secret sharing scheme* $(\mathsf{Share}, \mathsf{Rec})$ *is* (t, δ)-robust *if the following properties hold for any distribution of $s \in \mathcal{S}$ and for any adversary as specified above.*

Privacy: *Before* Rec *is started, the adversary has no more information on the shared secret s than he had before the execution of* Share.

Reconstructability: *At the end of* Rec*, the reconstructor R outputs $s' = s$ except with probability at most δ.*

It is known that in any (not necessarily robust but perfectly private) secret sharing scheme, the bit-size of every share σ_i is at least the bit-size $\log |\mathcal{S}|$ of the secret. In this paper, we are interested in how much redundancy needs to be added to this minimal share size in order to achieve robustness, i.e., in the quantity $\max_i(\log |\Sigma_i|) - \log |\mathcal{S}|$, where Σ_i denotes the set of all possible shares σ_i for player i. We call this quantity the *overhead* of a scheme.

2.2 Message Authentication Codes

A message authentication code (MAC) is a tool that enables to verify the integrity of a message. Unconditionally secure MACs were initially invented by Carter and Wegman [21, 22]. We give here a definition that suits our needs.

Definition 2.2. *A* message authentication code *(or* MAC*) for a finite message space $\mathcal{M}$ consists of a function $MAC : \mathcal{M} \times \mathcal{K} \to \mathcal{T}$ for finite sets $\mathcal{K}$ and $\mathcal{T}$. Such a* MAC *is called ε-secure if for all $m, \hat{m} \in \mathcal{M}$ with $m \neq \hat{m}$ and for all $\tau, \hat{\tau} \in \mathcal{T}$:*

$$P[MAC(\hat{m}, K) = \hat{\tau} \mid MAC(m, K) = \tau] \leq \delta\,,$$

where the random variable K is uniformly distributed over $\mathcal{K}$.

[9] It may look unnatural at first glance that the adversary does not get to see the communication between D and the players, but he does get to see the communication between the players and R. The reason why we want to allow him to observe the communication with R is that in certain applications, it is actually the set of all players that wants/needs to reconstruct the secret. In this case, whenever the reconstruction procedure dictates player P_i to send some information to R, it has to send that information to all the players. But this of course then means that if at least one of the players is corrupt, then the adversary gets to see all the communication intended for R.

It is well known that if $\mathcal{M}$ is a finite field $\mathbb{F}$, then $MAC : \mathbb{F} \times \mathbb{F}^2 \to \mathbb{F}$ with $(m, (\alpha, \beta)) \mapsto \alpha \cdot m + \beta$ is a ε-secure MAC with $\varepsilon = 1/|\mathbb{F}|$. More generally, as first shown in [7, 13, 19],

$$MAC : \mathbb{F}^d \times \mathbb{F}^2 \to \mathbb{F},\ ((m_1 \ldots, m_d), (\alpha, \beta)) \mapsto \sum_{k=1}^{d} \alpha^i \cdot m_i + \beta$$

is a ε-secure MAC with $\varepsilon = d/|\mathbb{F}|$.

2.3 Reed-Solomon Error Correction

Let $\mathbb{F}$ be a finite field, let n' be a positive integer, and let $x_1, \ldots, x_{n'}$ be pairwise distinct interpolation points in $\mathbb{F}$. We consider the problem of recovering a polynomial $f(X) \in \mathbb{F}[X]$ of degree at most t, when given a perturbed version of its evaluations $(f(x_1), \ldots, f(x_{n'}))$, i.e., when given a vector $(y_1, \ldots, y_{n'})$ for which it is promised that $y_i = f(x_i)$ for all but e of the indexes $i \in \{1, \ldots, n'\}$, where e is some parameter, but it is not known for *which* indices. This is known as Reed-Solomon error correction. It is not hard to see, using Lagrange interpolation, that $f(X)$ is uniquely determined from $(y_1, \ldots, y_{n'})$ if (and only if) $n' \geq t + 1 + 2e$. Indeed, if there are two such polynomials, then they must coincide in at least $t + 1$ points, and hence are identical. Furthermore, there exist algorithms that permit to *efficiently compute* $f(X)$ from $(y_1, \ldots, y_{n'})$ in case $n' \geq t + 1 + 2e$, for instance the Berlekamp-Welch algorithm [1]. A simplified version of the original Berlekamp-Welch algorithm, provided by Gemmell and Sudan, can be found in [11].

3 The New Scheme and Its Analysis

Let t be an arbitrary positive integer, and $n = 2t + 1$. Consider Shamir's secret sharing scheme over a field $\mathbb{F}$ with $|\mathbb{F}| > n$, with pairwise-distinct non-vanishing interpolation points $x_1, \ldots, x_n \in \mathbb{F}$. Furthermore, let $MAC : \mathbb{F} \times \mathcal{K} \to \mathcal{T}$ be an ε-secure MAC with message space $\mathbb{F}$. The sharing procedure Share of our new scheme is presented in Figure 1.

Local computation: On input $s \in \mathbb{F}$, the dealer D chooses a random sharing polynomial $f(X) \in \mathbb{F}[X]$ with degree at most t and $f(0) = s$, and he computes the Shamir shares $s_1 = f(x_1), \ldots, s_n = f(x_n)$. Furthermore, for every pair $i, j \in [n]$, he chooses a random $key_{ij} \in \mathcal{K}$ and computes $\tau_{ij} = MAC(key_{ji}, s_i)$.

Share distribution: For every $i \in [n]$, the dealer D sends to player P_i the share $\sigma_i = (s_i, \tau_{i1}, \ldots, \tau_{in}, key_{i1}, \ldots, key_{in})$.

Fig. 1. Sharing procedure Share.

The new sharing procedure is identical to the sharing procedure of the Rabin-BenOr robust secret sharing scheme, except that we describe it by means of an arbitrary MAC. However, in the end we will use a MAC with *short* keys and tags and a correspondingly weak security, which would render the original Rabin-BenOr scheme insecure. The reader may think of ε being $1/n$; indeed, as we will see later, this will give us δ-robustness with δ approximately $2^{-n/4}$.

In order to deal with a non-negligible ε for the security of the MAC, we need a sophisticated reconstruction procedure. The idea is to inspect the *acceptance graph*, which describes which share s_i is consistent (together with the corresponding tag) with which authentication key key_{ji}, more carefully. Instead of accepting a share s_i as being correct as soon as it is consistent with (the keys of) at least $t+1$ players (which means that a dishonest player only needs to fool *one* honest player to get his share accepted), we will require, for a share to be accepted, that it is consistent with at least $t+1$ players *that hold accepted shares*. In other words, once a share is declared incorrect, then this player's vote is not counted anymore, making it harder for other incorrect shares to be accepted. Some might still survive, though; to take care of that, Reed-Solomon error correction is then applied to the set of accepted shares. The procedure is described in Figure 2. It is easy to see that the set $\mathcal{I}$ in step 2 is well defined and can efficiently be computed by starting with the set of all $i \in [n]$ and inductively eliminating "bad" players.

First round: Every player P_i sends s_i and $\tau_{i1}, \ldots, \tau_{in}$ to the reconstructor R.

Second round: Every player P_i sends $key_{i1}, \ldots, key_{in}$ to R.

Local computation:

1. For every $i, j \in [n]$, R sets v_{ij} to be 1 if the share s_i is accepted by (the key of) player P_j, i.e., if $\tau_{ij} = MAC(key_{ji}, s_i)$, and else to 0.
2. R computes the largest set $\mathcal{I} \subseteq [n]$ with the property that

$$\forall\, i \in \mathcal{I} : \; \big|\{j \in \mathcal{I} \,|\, v_{ij} = 1\}\big| = \sum_{j \in \mathcal{I}} v_{ij} \geq t+1\,;$$

   in other words, such that every share of a player in $\mathcal{I}$ is accepted by at least $t+1$ players in $\mathcal{I}$.
   Clearly, $\mathcal{I}$ contains all honest players. Let $c = |\mathcal{I}| - (t+1)$ be the maximum number of corrupt players in $\mathcal{I}$.
3. Using Berlekamp-Welch, R computes a polynomial $f(X) \in \mathbb{F}[X]$ of degree at most t such that $f(x_i) = s_i$ for at least $(t+1) + \frac{c}{2}$ players i in $\mathcal{I}$. If no such polynomial exists then R outputs $\perp$; otherwise, he outputs $s = f(0)$.

Fig. 2. Reconstruction procedure Rec.

The intuition behind the security is the following. If the corrupt players hand in only a few incorrect shares and many correct shares, then the incorrect shares have a good chance of surviving since they only need to be consistent with the keys of a few honest players. However, since there are only a few incorrect shares, the Reed-Solomon decoding will take care of them. On the other hand, if the corrupt players hand in many incorrect shares, then, because there are many of them, some will probably be detected as being incorrect, which will make it harder for the remaining incorrect shares because they now need to be consistent with more honest players, which means that some more incorrect shares will probably be detected, which will make it even harder for the remaining ones, etc., so that in the end, hopefully only a few survive so that again Reed-Solomon error correction takes care. The following theorem shows that the above intuition is indeed correct. However, the formal reasoning is quite involved, as we will see later.

Theorem 3.1. *For any positive integer t, any finite field $\mathbb{F}$ with $|\mathbb{F}| > n = 2t+1$, and any ε-secure MAC $: \mathbb{F} \times \mathcal{K} \to \mathcal{T}$ with $\varepsilon \leq 1/(t+1)$, the pair* (Share, Rec) *forms an n-player (t, δ)-robust secret sharing scheme for message space $\mathbb{F}$ with*

$$\delta \leq \mathrm{e} \cdot \big((t+1)\varepsilon\big)^{(t+1)/2},$$

where $\mathrm{e} = \exp(1)$.

The crucial property on δ is that it is not of order ε, as in the Rabin-BenOr scheme, but of order $\varepsilon^{\Omega(n)}$. This allows us to reduce the authentication key and tag sizes by a factor (linear in) n.

Specifically, we can get the following instantiation. Let λ be an arbitrary parameter, and let $GF(2^m)$ be the binary field with $2^m > n$ elements. By Section 2.2, there exists an ε-secure $MAC : GF(2^m) \times \mathcal{K} \to \mathcal{T}$ with $\mathcal{K} = GF(2^\lambda)^2$ and $\mathcal{T} = GF(2^\lambda)$ and $\varepsilon \leq m/2^\lambda$. By Theorem 3.1, the resulting secret sharing scheme is δ-robust for $\delta \leq \mathrm{e} \cdot ((t+1)m/2^\lambda)^{(t+1)/2}$. Therefore, for a given security parameter κ, setting $\lambda = \lceil \log(t+1) + \log(m) + \frac{2}{t+1}(\kappa + \log(\mathrm{e}))\rceil$, we obtain $\delta \leq 2^{-\kappa}$, and every share consists of the ordinary m-bit Shamir share plus an overhead of

$$3n\lambda \leq 12\kappa + 3n(\log(t+1) + \log(m) + 3)$$

bits.

Corollary 3.2. *For any positive integers t, m, κ, and for $n = 2t+1$, there exists an n-player (t, δ)-robust secret sharing scheme for message space $\mathcal{S} = \{0,1\}^m$, with $\delta = 2^{-\kappa}$ and an overhead of $O\big(\kappa + n(\log n + \log m)\big)$.*

We will now prove Theorem 3.1. Although the idea for the new scheme is rather simple and natural, the security analysis is non-trivial. One reason is that it is not clear what the optimal strategy for the adversary is. In comparison, in the Rabin-BenOr scheme, it is obvious that the best the adversary can do is to have every corrupt player hand in an incorrect share and hope that at least one gets accepted. In our new scheme, however, it might be advantages to have

some corrupt players hand in *correct* shares; the reason being that such players could support incorrect shares of other corrupt players, making it easier for them to survive the elimination round. On the other hand, having too many corrupt players handing in correct shares will facilitate Reed-Solomon decoding. Another reason is that there seems to be some circularity: in order to argue that many incorrect shares get eliminated, we want to argue that incorrect shares need to be accepted by many honest players in order to survive, but this is only true once many incorrect shares got eliminated.

Our proof below is pretty much "brute force". We work out a bound on the failure probability (for an arbitrary strategy) by essentially listing all possible scenarios of which incorrect shares might be accepted by which honest players, and then we simplify the resulting unhandy expression.

Proof (of Theorem 3.1). Privacy is obvious. It remains to prove the reconstructability property. Consider the state of the reconstruction phase right before the second round of communication, i.e., after R has received the shares and tags, but before the keys are communicated. We may assume that at this stage, the adversary has corrupted t players. We define the following sets. $\mathcal{A} \subset [n]$ is the set of corrupt players i that have handed in a *modified* Shamir share s_i, and $\mathcal{P} \subset [n]$ is the set of corrupt players i that have handed in the *correct* Shamir share s_i. It holds that $|\mathcal{A}| + |\mathcal{P}| = t$. The remaining set $\mathcal{H} = [n] \setminus (\mathcal{A} \cup \mathcal{P})$ is the set of uncorrupt players.[10]

We consider the probability space specified by the random choices of the authentication keys held by the uncorrupt players, conditioned on the shares and tags handed out by the dealer D during the sharing procedure, plus the choices of the (possibly modified) authentication keys claimed in the second round of the reconstruction procedure by the corrupt players. For every pair $i, j \in [n]$, we can define the binary random variable V_{ij} that specifies if player P_i's (possibly incorrect) share with the corresponding tag is accepted by player P_j's key. Since the authentication keys of uncorrupt players have been chosen independently, all the V_{ij} with $i \in [n]$ and $j \in \mathcal{H}$ are independent. Also, $V_{ij} = 1$ with probability 1 for every pair $i, j \in \mathcal{H}$, i.e., honest players accept each others shares. Furthermore, by the security of the MAC (Definition 2.2), $P[V_{ij} = 1] \leq \varepsilon$ for all $i \in \mathcal{A}$ and $j \in \mathcal{H}$. Finally, it is not too hard to see that it does not help the corrupt players to hand in correct Shamir shares but incorrect authentication tags: a player in $\mathcal{P}$ that is eliminated is of no use for the adversary; thus, we may assume that $V_{ij} = 1$ for every pair $i \in \mathcal{P}$, $j \in [n]$.

It follows that the set $\mathcal{I}$ computed during Rec (which depends on the V_{ij}'s and thus we treat it as a random variable here) contains $\mathcal{H}$ and $\mathcal{P}$ with certainty. Thus, the reconstruction procedure is guaranteed to output the correct secret if at most p players $i \in \mathcal{A}$ end up in $\mathcal{I}$, where $p = |\mathcal{P}|$. Indeed, if $|\mathcal{A} \cap \mathcal{I}| \leq p$, then the requirement for Reed-Solomon decoding is satisfied (see Section 2.3 with $n' = |\mathcal{I}| = t + 1 + c = t + 1 + p + e$ where $e = |\mathcal{A} \cap \mathcal{I}| \leq p$), and the polynomial

[10] The mnemonic is: $\mathcal{A}$ for *actively* corrupt, $\mathcal{P}$ for *passively* corrupt, and $\mathcal{H}$ for *honest*, but we stress that the players in $\mathcal{P}$ are merely passive with respect to their respective Shamir shares s_i; they may very well lie about their authentication keys and tags.

$f(X)$ computed during Rec is guaranteed to satisfy $f(x_i) = s_i$ for at least $t+1$ correct shares, and thus it is the correct sharing polynomial and $f(0)$ the correct secret.

It thus remains to analyze the probability $P[|\mathcal{A} \cap \mathcal{I}| > p]$. For this, it is sufficient to consider the case $p \leq (t-1)/2$; indeed, if $p > (t-1)/2$ and thus $p \geq t/2$ then obviously $|\mathcal{A}| \leq p$ and hence $P[|\mathcal{A} \cap \mathcal{I}| \leq p]$ with certainty. Actually, we will now show that $P[|\mathcal{A} \cap \mathcal{I}| > 0]$ is small if $p \leq (t-1)/2$.

We can write $P[|\mathcal{A} \cap \mathcal{I}| > 0] = \sum_\ell P[|\mathcal{A} \cap \mathcal{I}| = \ell]$ where the sum ranges from $\ell = 1$ to $t - p$. In order to bound the probability $P[|\mathcal{A} \cap \mathcal{I}| = \ell]$, it is convenient to introduce for every $i \in \mathcal{A}$ the random variable

$$N_i = \sum_{j \in \mathcal{H}} V_{ij} = \big|\{j \in \mathcal{H} \,|\, V_{ij} = 1\}\big|,$$

i.e., the number of honest players that accept P_i's incorrect share. Note that since the V_{ij}'s are independent for all $i \in [n]$ and $j \in \mathcal{H}$, so are all the N_i's. We can now bound $P[|\mathcal{A} \cap \mathcal{I}| = \ell]$ for an arbitrary ℓ in the range $1 \leq \ell \leq t - p$ as follows.

$$\begin{aligned}
P\big[|\mathcal{A} \cap \mathcal{I}| = \ell\big] &\leq P\big[\exists\, \mathcal{A}_\circ \subseteq \mathcal{A} : (|\mathcal{A}_\circ| = \ell) \wedge (\forall\, i \in \mathcal{A}_\circ : N_i \geq t + 1 - p - \ell)\big] \\
&\leq \sum_{\substack{\mathcal{A}_\circ \subseteq \mathcal{A} \\ |\mathcal{A}_\circ| = \ell}} P\big[\forall\, i \in \mathcal{A}_\circ : N_i \geq t + 1 - p - \ell\big] \\
&= \sum_{\substack{\mathcal{A}_\circ \subseteq \mathcal{A} \\ |\mathcal{A}_\circ| = \ell}} \prod_{i \in \mathcal{A}_\circ} P\big[N_i \geq t + 1 - p - \ell\big] \\
&\leq \sum_{\substack{\mathcal{A}_\circ \subseteq \mathcal{A} \\ |\mathcal{A}_\circ| = \ell}} \prod_{i \in \mathcal{A}_\circ} P\big[\exists\, \mathcal{H}_\circ \subseteq \mathcal{H} : (|\mathcal{H}_\circ| = t + 1 - p - \ell) \wedge (\forall\, j \in \mathcal{H}_\circ : V_{ij} = 1)\big] \\
&\leq \sum_{\substack{\mathcal{A}_\circ \subseteq \mathcal{A} \\ |\mathcal{A}_\circ| = \ell}} \prod_{i \in \mathcal{A}_\circ} \sum_{\substack{\mathcal{H}_\circ \subseteq \mathcal{H} \\ |\mathcal{H}_\circ| = t+1-p-\ell}} P\big[\forall\, j \in \mathcal{H}_\circ : V_{ij} = 1\big]
\end{aligned}$$

Now, since $P\big[\forall\, j \in \mathcal{H}_\circ : V_{ij} = 1\big] = \prod_{j \in \mathcal{H}_\circ} P[V_{ij} = 1]$ and $P[V_{ij} = 1] \leq \varepsilon$ for all $i \in \mathcal{A}$ and $j \in \mathcal{H}$, we can proceed as follows, where we write $a = |\mathcal{A}| = t - p \geq (t+1)/2$ and $\tilde{\varepsilon} = (t+1)\varepsilon$.

$$\begin{aligned}
P\big[|\mathcal{A} \cap \mathcal{I}| = \ell\big] &\leq \binom{a}{\ell} \cdot \left(\binom{t+1}{a-\ell+1} \cdot \varepsilon^{a-\ell+1}\right)^\ell \\
&\leq \binom{a}{\ell} \cdot \left(\frac{(t+1)^{a-\ell+1}}{(a-\ell+1)!} \cdot \varepsilon^{a-\ell+1}\right)^\ell = \frac{a!}{\ell!(a-\ell)!} \cdot \left(\frac{\tilde{\varepsilon}^{a-\ell+1}}{(a-\ell+1)!}\right)^\ell \\
&= \frac{\tilde{\varepsilon}^{\ell(a-\ell+1)}}{((a-\ell)!)^\ell} \cdot \frac{a!/(a-\ell)!}{\ell!(a-\ell+1)^\ell} = \frac{\tilde{\varepsilon}^{\ell(a-\ell+1)})}{((a-\ell)!)^\ell} \cdot \prod_{k=1}^{\ell} \underbrace{\frac{a-\ell+k}{k(a-\ell+1)}}_{\leq 1}
\end{aligned}$$

$$\leq \frac{\tilde{\varepsilon}^{\ell(a-\ell+1)}}{((a-\ell)!)^{\ell}} \leq \frac{\tilde{\varepsilon}^{\ell(a-\ell+1)}}{(a-\ell)!} \leq \frac{\tilde{\varepsilon}^{a}}{(a-\ell)!}$$

where the very last inequality follows from $\tilde{\varepsilon} = (t+1)\varepsilon \leq 1$ (by assumption on ε) and the fact that $\min\{\ell(a-\ell+1) \,|\, 1 \leq \ell \leq t-p = a\} = a$, which can easily be verified.[11] We can now conclude that

$$P[|\mathcal{A} \cap \mathcal{I}| > 0] = \sum_{\ell=1}^{t-p} P[|\mathcal{A} \cap \mathcal{I}| = \ell] \leq \sum_{\ell=1}^{t-p} \frac{\tilde{\varepsilon}^{a}}{(a-\ell)!}$$
$$\leq \tilde{\varepsilon}^{a} \sum_{k=0}^{a-1} \frac{1}{k!} \leq \tilde{\varepsilon}^{a} \sum_{k=0}^{\infty} \frac{1}{k!} \leq \tilde{\varepsilon}^{(t+1)/2}\,\mathrm{e}$$

which proves the claim. □

4 Conclusion and Open Questions

We have shown and analyzed a new robust secret sharing scheme, which combines the computational efficiency of the Rabin-BenOr scheme [17] with a close-to-optimal overhead of $\tilde{O}(\kappa + n)$ in the share size, as featured by the (computationally inefficient) scheme of Cramer *et al.* [5].

It is interesting to see that our new scheme is based on a completely different approach than the scheme of Cramer *et al.*, but displays the same order-n gap to the known lower bound of $\Omega(\kappa)$ for the share size in robust secret sharing. This raises the question of the true optimal share size in robust secret sharing: is the linear term in n inherent, or is it an artifact of current constructions?

Acknowledgments

We thank Brett Hemenway for multiple helpful discussions. R.O. was supported in part by NSF grants 0830803, 09165174, 1065276, 1118126 and 1136174, US-Israel BSF grant 2008411, B. John Garrick Foundation, Teradata award, Intel equipment grant, NSF Cybertrust grant, OKAWA Award, Xerox Innovation Group Award, IBM Faculty Award, Lockheed-Martin Corporation and the Defense Advanced Research Projects Agency through the U.S. Office of Naval Research under Contract N00014-11-1-0392. The views expressed are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government. Y.R was supported in part by the I-CORE program 4/11, by ISF grant 856-11, and by BSF grant 2008059. The research was initiated while S.F. and Y.R. were visiting UCLA, and S.F. and Y.R. would like to thank UCLA for the hospitality.

[11] Indeed, the function $\ell \mapsto \ell(a-\ell+1)$ is concave and thus reaches its minimum at one of the boundaries.

References

1. Elwyn R. Berlekamp and Lloyd R. Welch. Error correction of algebraic block codes. U.S. Patent Number 4.633.470, 1986.
2. George R. Blakley. Safeguarding cryptographic keys. In *National Computer Conference*, volume 48, pages 313–317. AFIPS Press, 1979.
3. Sergio Cabello, Carles Padró, and Germán Sáez. Secret sharing schemes with detection of cheaters for a general access structure. In *Fundamentals of Computation Theory*, volume 1684 of *Lecture Notes in Computer Science*, pages 185–194, 1999.
4. Benny Chor, Shafi Goldwasser, Silvio Micali, and Baruch Awerbuch. Verifiable secret sharing and achieving simultaneity in the presence of faults (extended abstract). In *26th Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 383–395, 1985.
5. Ronald Cramer, Ivan Damgård, and Serge Fehr. On the cost of reconstructing a secret, or VSS with optimal reconstruction phase. In *Advances in Cryptology—CRYPTO '01*, volume 2139 of *Lecture Notes in Computer Science*, pages 503–523. Springer, 2001.
6. Ronald Cramer, Yevgeniy Dodis, Serge Fehr, Carles Padró, and Daniel Wichs. Detection of algebraic manipulation with applications to robust secret sharing and fuzzy extractors. In *Advances in Cryptology—EUROCRYPT '08*, volume 4965 of *Lecture Notes in Computer Science*, pages 471–488. Springer, 2008.
7. Bert den Boer. A simple and key-economical unconditional authentication scheme. *Journal of Computer Security*, 2:65–72, 1993.
8. Danny Dolev, Cynthia Dwork, Orli Waarts, and Moti Yung. Perfectly secure message transmission. In *31st Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, volume I, pages 36–45, 1990.
9. Juan A. Garay, Clint Givens, and Rafail Ostrovsky. Secure message transmission with small public discussion. In *Advances in Cryptology—EUROCRYPT '10*, volume 6110 of *Lecture Notes in Computer Science*, pages 177–196. Springer, 2010.
10. Juan A. Garay, Clint Givens, and Rafail Ostrovsky. Secure message transmission by public discussion: A brief survey. In *Coding and Cryptology - Third International Workshop (IWCC)*, volume 6639 of *Lecture Notes in Computer Science*, pages 126–141. Springer, 2011.
11. Peter Gemmell and Madhu Sudan. Highly resilient correctors for polynomials. *Information Processing Letters*, 43(4):169–174, 1992.
12. Yuval Ishai, Rafail Ostrovsky, and Hakan Seyalioglu. Identifying cheaters without an honest majority. In *Theory of Cryptography Conference (TCC)*, volume 7194 of *Lecture Notes in Computer Science*. Springer, 2012.
13. Thomas Johansson, Gregory Kabatianskii, and Ben Smeets. On the relation between A-codes and codes correcting independent errors. In *Advances in Cryptology—EUROCRYPT '93*, volume 765 of *Lecture Notes in Computer Science*, pages 1–11. Springer, 1993.
14. Kaoru Kurosawa and Kazuhiro Suzuki. Almost secure (1-round, n -channel) message transmission scheme. *IEICE Transactions*, 92-A(1), 2009.
15. Wakaha Ogata, Kaoru Kurosawa, and Douglas R. Stinson. Optimum secret sharing scheme secure against cheating. *SIAM Journal on Discrete Mathematics*, 20(1):79–95, 2006.
16. Torben Pryds Pedersen. Non-interactive and information-theoretic secure verifiable secret sharing. In *Advances in Cryptology—CRYPTO '91*, volume 576 of *Lecture Notes in Computer Science*, pages 129–140. Springer, 1991.

17. Tal Rabin and Michael Ben-Or. Verifiable secret sharing and multiparty protocols with honest majority. In *21st Annual ACM Symposium on Theory of Computing (STOC)*, pages 73–85, 1989.
18. Adi Shamir. How to share a secret. *Communications of the ACM*, 22(11):612–613, November 1979.
19. Richard Taylor. An integrity check value algorithm for stream ciphers. In *Advances in Cryptology—CRYPTO '93*, volume 773 of *Lecture Notes in Computer Science*, pages 40–48. Springer, 1993.
20. Martin Tompa and Heather Woll. How to share a secret with cheaters. In *Advances in Cryptology—CRYPTO '86*, volume 263 of *Lecture Notes in Computer Science*, pages 261–265. Springer, 1986.
21. Mark N. Wegman and J. Lawrence Carter. New classes and applications of hash functions. In *20th Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 175–182, 1979.
22. Mark N. Wegman and J. Lawrence Carter. New hash functions and their use in authentication and set equality. *Journal of Computer and System Science*, 22(3):265–279, 1981.