Secret Sharing with Certified Deletion

James Bartusek[1] and Justin Raizes[2]

[1]UC Berkeley
[2]Carnegie Mellon University

Abstract

Secret sharing allows a user to split a secret into many shares so that the secret can be recovered if, and only if, an authorized set of shares is collected. Although secret sharing typically does not require any computational hardness assumptions, its security *does* require that an adversary cannot collect an authorized set of shares. Over long periods of time where an adversary can benefit from multiple data breaches, this may become an unrealistic assumption.

We initiate the systematic study of secret sharing *with certified deletion* in order to achieve security *even against an adversary that eventually collects an authorized set of shares*. In secret sharing with certified deletion, a (classical) secret s is split into quantum shares that can be destroyed in a manner verifiable by the dealer.

We put forth two natural definitions of security. **No-signaling security** roughly requires that if multiple non-communicating adversaries delete sufficiently many shares, then their combined view contains negligible information about s, even if the total set of corrupted parties forms an authorized set. **Adaptive security** requires privacy of s against an adversary that can continuously and adaptively corrupt new shares and delete previously-corrupted shares, as long as the total set of corrupted shares minus deleted shares remains unauthorized.

Next, we show that these security definitions are achievable: we show how to construct (i) a secret sharing scheme with no-signaling certified deletion for *any monotone access structure*, and (ii) a *threshold* secret sharing scheme with adaptive certified deletion. Our first construction uses Bartusek and Khurana's (CRYPTO 2023) 2-out-of-2 secret sharing scheme with certified deletion as a building block, while our second construction is built from scratch and requires several new technical ideas. For example, we significantly generalize the "XOR extractor" of Agarwal, Bartusek, Khurana, and Kumar (EUROCRYPT 2023) in order to obtain better seedless extraction from certain quantum sources of entropy, and show how polynomial interpolation can double as a high-rate randomness extractor in our context of threshold sharing with certified deletion.

1 Introduction

Secret sharing [Sha79, Bla79, ISN87] is a foundational cryptographic primitive that allows a dealer to distribute a secret s among n parties so that only certain "authorized" subsets of the parties may recover the secret. A particularly common scenario is (k, n) *threshold* secret sharing, where the dealer splits s into n shares such that any k of the shares can be combined to recover the secret s, but any $k-1$ or fewer shares leak no information about s. However, one can also consider a much more versatile setting, in which the authorized subsets of n are defined by any *monotone access structure* $\mathbb{S}$.[1] Secret sharing schemes are ubiqui-

[1]A set $\mathbb{S}$ of subsets of $[n]$ is monotone if for any subset $S \in \mathbb{S}$, it holds that $S' \in \mathbb{S}$ for all supersets $S' \supset S$.

tous in cryptography, and we refer the reader to Beimel's survey [Bei11] for a broader discussion, including several applications.

A particularly appealing aspect of secret sharing that sets it apart from most other cryptographic primitives is that it *doesn't require computational hardness assumptions*. That is, one can construct secret sharing schemes for arbitrary monotone access structures secure against any computationally unbounded adversary (e.g. [ISN87, BL90, LV18]).

However, the security of these schemes still rests on a stringent assumption: over the course of the (potentially unbounded) adversary's operation, it only ever sees an unauthorized set of shares. This may be unacceptable for users sharing particularly sensitive information. Even if an adversary initially may only access a limited number of shares, over time they may be able to corrupt more and more parties, or perhaps more and more shares become compromised independently and are leaked into the public domain. A user who becomes paranoid about this possibility generally has no recourse, and, worse yet, cannot even *detect* if an adversary has obtained access to enough shares to reconstruct their secret.

In this work, we ask whether it is possible to strengthen the standard notion of secret sharing security, and relax the assumption that the adversary only ever corrupts an unauthorized set of parties. In particular:

Is it possible to achieve meaningful secret sharing security against adversaries that may eventually corrupt authorized sets of parties?

Now, if the shares consist of only classical data, then there is no hope of answering the above question in the affirmative. Indeed, once an adversary obtains any share, they can make a copy and store it away. Once they've collected and stored an authorized set, they'll be able to recover the secret due to the correctness of the secret sharing scheme.

Certified deletion. On the other hand, the *uncertainty principle* of quantum mechanics offers some hope: if shares are encoded into quantum states, then the useful share information may be erased by applying certain destructive measurements. Thus, a user that is worried about an adversary eventually corrupting an authorized set of shares may request and verify that this "deletion" operation is performed on some set of their shares. Now, even if the adversary learns enough shares in the future to constitute an authorized set, the already-deleted shares will remain useless, and there is hope that the user's secret remains private.

Indeed, the basic idea of leveraging the uncertainty principle to perform "certified deletion" of private information was first put forth by Broadbent and Islam [BI20] in the context of one-time-pad encryption, and has since been applied in several contexts throughout cryptography, e.g. [HMNY21, HMNY22, Por23, BK23, BKM+23, HKM+24, BGK+24]. In fact, Bartusek and Khurana [BK23] previously constructed a very limited flavor of secret sharing with certified deletion, namely, 2-out-of-2 secret sharing where only one of the two shares admits the possibility the deletion. Their scheme allows a user to split a secret s into a quantum share $|\mathsf{sh}_1\rangle$ and a classical share sh_2. If an adversary first obtains and deletes $|\mathsf{sh}_1\rangle$, then obtains sh_2, it will still be unable to reconstruct the secret s. One can also view the original one-time-pad encryption with certified deletion scheme of [BI20] as exactly this flavor of 2-out-of-2 secret sharing with certified deletion, where the quantum share is the ciphertext and the classical share is the secret key. In this work, we show that it is possible to introduce certified deletion guarantees into more versatile and general flavors of secret sharing, addressing several definitional and technical issues along the way.

1.1 Our Results

We formulate two powerful but incomparable notions of certified deletion security for general-purpose secret sharing schemes, and show how to construct a scheme satisfying each definition. One of our key technical tools is a high-rate seedless extractor from certain quantum sources of entropy that significantly generalizes and improves upon the "XOR extractor" of [ABKK23].

No-signaling security. First, we address the shortcomings of [BK23]'s security definition for 2-out-of-2 secret sharing sketched above, and formulate a natural extension that (i) applies to schemes for *any* monotone access structure, and (ii) allows for the possibility that *any* of the shares may be deleted.

In particular, we model a scenario involving multiple non-communicating adversaries that each individually have access to some unauthorized set of shares. These adversaries may even share entanglement, but may not exchange messages. Now, the user may request that some of its shares are deleted. If the adversaries jointly delete enough shares so that the remaining undeleted shares form an *unauthorized* set, then we combine the views of all the adversaries together, and require that the user's secret remains private even given this joint view. That is, even if a single adversarial entity is eventually able to corrupt up to *all* of the parties, they will still not be able to recover the secret if enough shares have previously been deleted.[2]

We refer to this security notion for secret sharing schemes as *no-signaling security* (see Section 5 for a precise definition), emphasizing the fact that shares must be deleted by adversaries that cannot yet pool information about an authorized set of shares, as this would trivially allow for reconstruction of the secret. Then, in Section 6 we show how to combine [BK23]'s simple 2-out-of-2 secret sharing scheme with any standard secret sharing scheme for monotone access structure $\mathbb{S}$ (e.g. [ISN87, BL90, LV18]) in order to obtain a secret sharing scheme for $\mathbb{S}$ with no-signaling security.

Theorem 1 (Informal)**.** *There exists a secret sharing scheme with no-signaling certified deletion security for any monotone access structure* $\mathbb{S}$.

Adaptive security. Next, we consider a particularly cunning but natural class of adversaries that exhibit the following behavior. Suppose that initially the adversary only obtains access to some unauthorized set of shares. At some point, the user becomes paranoid and requests that some subset of these shares are deleted. The adversary obliges but then continues to corrupt new parties or locate other leaked shares. The adversary may continue to delete some of these shares to appease the user, while continuing to work behind the scenes to mount a long-term attack on the system. However, as long as the set of corrupted parties minus the set of certifiably deleted shares continues to be unauthorized, we can hope that the user's secret remains private from such an adversary.

Unfortunately, the notion of no-signaling security does not capture the behavior of such an *adaptive* adversary. That is, no-signaling security only models adversaries that delete once, and then receive some extra information after this single round of deletion. Thus, we formalize *adaptive security* as an alternative and quite strong notion of certified deletion security for secret sharing schemes (see Section 5 for a precise definition).

Protecting against such arbitrarily adaptive adversaries turns out to be a significant challenge. The main technical component of our work realizes a secret sharing scheme with adaptive certified deletion security for the specific case of threshold access structures (Section 7).

Theorem 2 (Informal)**.** *There exists a threshold secret sharing scheme with adaptive certified deletion security.*

[2]We remark that this definition also captures adversaries that don't end up corrupting *all* the shares, by imagining that there is a separate component of the adversary that honestly deletes the uncorrupted shares.

High-rate seedless extractors from quantum sources of entropy. One of our technical building blocks is an improved method for seedless extraction from certain quantum sources of entropy. Roughly, the source of entropy comes from performing a standard basis measurement on a register that is in superposition over a limited number of Fourier basis states.

While seedless extraction from such sources of entropy [ABKK23] has been a crucial component in previous realizations of cryptographic primitives with certified deletion [BK23],[3] the technique had been limited to (i) extracting from qubit registers (i.e. where data is natively represented as superpositions of bitstrings) and (ii) extracting only a single bit of entropy. Here, we generalize these techniques to extract from qu*dit* registers (i.e. where data is natively represented as superpositions of vectors over finite fields), and produce several field elements worth of entropy, vastly improving the rate of extraction. Beyond being interesting in its own right, it turns out that these improvements are crucial for showing security our construction of threshold sharing with adaptive certified deletion. Moreover, we show how these high-rate extraction techniques can be applied to extension fields, meaning that we can represent our quantum shares as string of qubits (as opposed to qudits), removing the need for entanglement in our construction. We refer the reader to Section 2.3 and Section 4 for more details.

2 Technical Overview

Intuitively, certified deletion for secret sharing aims to keep the secret private from an adversary if the total set of undeleted shares they have access to is unauthorized. One could formalize this by considering an adversary who initially receives an unauthorized set of shares and then deletes some of them. If the undeleted shares are still unauthorized when combined with the shares that the adversary did not receive, then we allow the adversary to access these remaining shares. This closely matches the definition of encryption with certified deletion, where the adversary initially receives and deletes a ciphertext $\mathsf{Enc}(k, m)$ encrypting message m using key k, and then later receives the key k.

However, this definition is not meaningful for all access structures. For example, in a k out of n access structure where $k < n/2$, the shares that the adversary does not start with *already* form an authorized set on their own, so it never makes sense to allow the adversary to access all of these shares at once. In this section, we give an overview of two different ways to address this definitional deficiency: no-signaling certified deletion and adaptive certified deletion.

2.1 No-Signaling Certified Deletion

In no-signaling certified deletion, we address this problem by allowing the adversary to delete from multiple sets of shares $P_1, \dots, P_\ell$. However, if $P_1 \cup \dots \cup P_\ell$ contains all shares, then the adversary as a whole gets to see every share before it generates any deletion certificates. Thus, to prevent trivial attacks, we do not allow the adversary to communicate across sets. However, the different parts of the adversary may still *share entanglement*. This modification yields the no-signaling certified deletion game $\mathsf{SS\text{-}NSCD}_{\mathbb{S}}(s)$ for secret s and access structure $\mathbb{S}$ over n parties, which we describe here.

1. The challenger secret-splits s into n shares with access structure $\mathbb{S}$.
2. Each adversary Adv_i is initialized with one register of a shared state $|\psi\rangle$, receives the shares in a set P_i, and produces some set of certificates $\{\mathsf{cert}_j\}_{j \in P_i}$. If Adv_i does not wish to delete share j, then it may set $\mathsf{cert}_j = \bot$.

[3]See discussion thereinfor why *seedless* as opposed to seeded extraction is crucial.

3. If the total set of shares that have not been deleted is unauthorized, then output the joint view of the adversaries. Otherwise, output $\perp$.

No-signaling certified deletion for secret sharing requires that for every secret pair (s_0, s_1) and every partition $P = (P_1, \ldots, P_\ell)$ of $[n]$, the outputs of $\mathsf{SS\text{-}NSCD}_{\mathbb{S}}(s_0)$ and $\mathsf{SS\text{-}NSCD}_{\mathbb{S}}(s_1)$ have negligible trace distance.

Tool: 2-of-2 Secret Sharing with Certified Deletion [BK23]. Recently, Bartusek and Khurana constructed a variety of primitives with certified deletion. One of these primitives is a secret sharing scheme which splits a secret s into a quantum share $|\mathsf{sh}_1\rangle$ and a classical share sh_2, along with a verification key vk that can be used to test the validity of deletion certificates. Given either one of the shares, the secret is hidden. Furthermore, if an adversary given $|\mathsf{sh}_1\rangle$ performs a destructive measurement that yields a valid deletion certificate, then they will never be able to recover s, even if they later obtain sh_2. Note that in this scheme, only one of the two shares can be deleted.

A Black-Box Compiler. We show how to compile Bartusek and Khurana's 2-of-2 certified deletion scheme together with any classical secret sharing scheme into a secret sharing scheme with no-signaling certified deletion. Notably, the compiled scheme inherits the same access structure as the classical secret sharing scheme. Thus, one can construct secret sharing with no-signaling certified deletion for general access structures by using any classical secret sharing scheme for general access structures, e.g. [ISN87, ABF+19].

As a starting point, let us first construct a scheme where only one of the shares can be deleted.

1. Secret split the secret s into a quantum share $|\mathsf{qsh}\rangle$ and a classical share csh using the 2-of-2 secret sharing scheme with certified deletion. This also produces a verification key vk.

2. Split the 2-of-2 classical share csh into classical shares $\mathsf{csh}_1, \ldots, \mathsf{csh}_n$ using the classical secret sharing scheme for $\mathbb{S}$.

3. The verification key is vk and the i'th share is csh_i. The deletable quantum share is $|\mathsf{qsh}\rangle$.

Given the quantum share and any authorized set of classical shares, s can be reconstructed by first recovering csh from the authorized set. On the other hand, any adversary which attempts to delete $|\mathsf{qsh}\rangle$ with only access to an unauthorized set of classical shares has no information about the 2-of-2 classical share csh. Thus if they produce a valid deletion certificate, they will have no information about s even after obtaining the rest of the classical shares, which only reveals csh.

Who Deletes? To finish the compiler, we need to enable certified deletion of *any share*. This can be achieved by adding a step at the beginning of the compiler to create n classical shares $\mathsf{sh}_1, \ldots, \mathsf{sh}_n$ of s with the same access structure $\mathbb{S}$. Then, the splitter can enable certified deletion for each share sh_i by using the prior compiler to produce a set of classical shares $\mathsf{csh}_{i,1}, \ldots, \mathsf{csh}_{i,n}$, a deletable quantum share $|\mathsf{qsh}_i\rangle$, and a verification key vk_i. The i'th share contains the deletable state $|\mathsf{qsh}_i\rangle$, as well as $\{\mathsf{csh}_{j,i}\}_{j\in[n]}$.

Note that anyone holding share i is able to delete sh_i by deleting $|\mathsf{qsh}_i\rangle$, as discussed previously. If sufficiently many shares are deleted, so that the only remaining sh_i form an unauthorized set, then no adversary can learn anything about the secret even after obtaining all of the remaining shares and residual states.

Proof of Security: Guessing Deletions. Although the intuition is straightforward, there is a nuance in the proof of security. When proving security, we wish to replace the deleted 2-of-2 secrets sh_i with empty secrets $\perp$. If we could do so, then security immediately reduces to the security of the classical $\mathbb{S}$-scheme, since only an unauthorized set of sh_i remains. However, it is difficult to determine which of these 2-of-2 secrets sh_i to replace with $\perp$ when preparing the shares.

Since non-local operations commute, we could consider generating the shares for each adversary Adv_i one at a time. For example, supposing Adv_1 operates on the set of shares $P_1 \subset [n]$, the experiment could initialize Adv_1 with uniformly random shares, and then for each $i \in P_1$, reverse-sample the shares $\{\mathsf{csh}_{i,j}\}_{j\in[n]\setminus P_1}$ for the rest of the adversaries to match either sh_i or $\perp$, depending on whether or not Adv_1 deleted share i.

Unfortunately, we cannot continue this strategy for all of the adversaries. It may be the case that the union of Adv_1 and Adv_2's shares $P_1 \cup P_2$ contains an authorized set. Thus, when initializing Adv_2, the challenger must already know whether, for each $i \in P_2$, the i'th share of s should be set to sh_i or $\perp$ (since this will be determined by $\{\mathsf{csh}_{i,j}\}_{j\in P_1\cup P_2}$). This view is constructed before the adversary decides whether or not to delete share i, so the only way for the challenger to do this is to guess whether the adversary will delete share i or not.

Now, guessing which shares the entire set of adversaries will delete incurs a multiplicative exponential (in n) loss in security. Fortunately, Bartusek and Khurana's 2-of-2 scheme actually satisfies an *inverse exponential* trace distance between the adversary's view of any two secrets, after deletion. Thus, by setting the parameters carefully, we can tolerate this exponential loss from guessing, and show that our scheme for general access structures satisfies negligible security.

2.2 Adaptive Certified Deletion

Intuitively, any definition of certified deletion should allow the adversary to eventually receive an authorized set of shares, as long as they have previously deleted enough shares so that their total set of undeleted shares remains unauthorized. In no-signaling certified deletion, we allowed multiple non-communicating adversaries to delete from different unauthorized sets of shares. That is, when Adv_i generates its set of certificates, it only has access to a single unauthorized set P_i. However, one could also imagine a more demanding setting where, after deleting some shares, the adversary can adaptively corrupt *new shares*, as long as their total set of undeleted shares remains unauthorized. Then, they can continue deleting shares and corrupting new shares as long as this invariant holds. This setting arises naturally when we consider an adversary which covertly compromises shares over a long period of time, while occasionally deleting shares to avoid revealing the extent of the infiltration. We call this notion *adaptive certified deletion*. It is defined using the following adaptive certified deletion game $\mathsf{SS\text{-}ACD}_{\mathbb{S}}(s)$.

1. The challenger splits the secret s into n shares with access structure $\mathbb{S}$. The adversary starts with an empty corruption set C and an empty deletion set D.

2. For as many rounds as the adversary likes, it gets to see the shares in C and choose whether to corrupt or delete a new share.

   Corrupt a new share. The adversary corrupts a new share i by adding i to C. If $C\setminus D$ is authorized, the experiment immediately outputs $\perp$.

   Delete a share. The adversary outputs a certificate cert for a share i. If cert is valid, add i to D. Otherwise, the experiment immediately outputs $\perp$.

3. When the adversary is done, the experiment outputs its view.

Adaptive certified deletion for secret sharing requires that for every secret pair (s_0, s_1), the outputs of $\mathsf{SS\text{-}ACD}_{\mathbb{S}}(s_0)$ and $\mathsf{SS\text{-}ACD}_{\mathbb{S}}(s_1)$ have negligible trace distance. In this work, we focus on the (k, n) threshold access structure, where any set of size $\geq k$ is authorized.

Incomparable Definitions. We have already seen that no-signaling certified deletion does not imply adaptive certified deletion. It is also the case that adaptive certified deletion does not imply no-signaling certified deletion. Consider a two-part no-signaling adversary Adv_1 and Adv_2 with views P_1 and P_2. To change $(\mathsf{Adv}_1, \mathsf{Adv}_2)$ to an adaptive adversary, one would need to come up with a transformation that deletes the same shares as $(\mathsf{Adv}_1, \mathsf{Adv}_2)$, in the same way. However, Adv_1 might not even decide which shares to delete until after they have seen every share in P_1. So, the new adaptive adversary would have to corrupt all of P_1 before it can delete a single share that Adv_1 would. Similarly, it would also have to corrupt all of P_2 before it knows which shares Adv_2 would delete. However, if $P_1 \cup P_2$ is authorized, then the experiment would abort before the new adaptive adversary gets the chance to delete shares for both Adv_1 and Adv_2.

An Attack on the No-Signaling Construction. Unfortunately, the previous construction actually does *not* in general satisfy adaptive certified deletion security. Indeed, observe that the classical parts of each share can never be deleted. Because of this, an adversary could, for any i, obtain k classical shares $\mathsf{csh}_{i,1}, \ldots, \mathsf{csh}_{i,k}$ that reveal csh_i, simply by corrupting and immediately deleting the first k shares one-by-one. Afterwards, the adversary will always have *both* the classical share csh_i and the quantum share $|\mathsf{qsh}_i\rangle$ when it corrupts a new share i, so it can recover the underlying classical share sh_i. Now it can "delete" the i'th share while keeping sh_i in its memory. Eventually, it can collect enough sh_i in order to obtain the secret s.

The core problem with the no-signaling construction is the fact that it encodes the 2-of-2 classical shares csh in a form which can never be deleted. If we were to take a closer look at Bartusek and Khurana's 2-of-2 scheme, we would observe that csh contains a mapping θ of which parts of $|\mathsf{qsh}\rangle$ encode the secret (the "data indices") and which parts contain only dummy information used to verify certificates (the "check indices"). Without θ, there is no way to decode the secret. Unfortunately, in order to encode θ in a deletable form, we seem to be back where we started - we need secret sharing with adaptive certified deletion!

A New Construction. To avoid this pitfall, we take a new approach that allows the parties to reconstruct *without knowledge of the check indices*. This removes the need to encode θ altogether. To achieve this, we begin with Shamir's secret sharing [Sha79], in which the shares are evaluation points of a degree $k-1$ polynomial f where $f(0) = s$. This polynomial is over some finite field $\mathbb{K}$ with at least $n+1$ elements. A useful property of Shamir's secret sharing is that it has good error-correcting properties - in fact, it also forms a Reed-Solomon code, which has the maximum possible error correction [RS60].

To split a secret s, we start by constructing a polynomial f where $f(0) = s$. Each share contains some number of evaluations of f encoded in the computational basis. These evaluations are mixed with a small number of random Fourier basis states that will aid in verifying deletion certificates. The positions of these checks, along with the value encoded, make up the verification key. An example share and key are illustrated here.

$$\begin{array}{lccccccccccc}
\mathsf{sh} = & |f(1)\rangle & \otimes & |f(2)\rangle & \otimes & \mathsf{QFT}\,|r_1\rangle & \otimes & |f(4)\rangle & \otimes & \mathsf{QFT}\,|r_2\rangle & \otimes & \ldots \\
\mathsf{vk} = & * & & * & & r_1 & & * & & r_2 & & \ldots
\end{array}$$

When reconstructing the secret, these checks are essentially random errors in the polynomial evaluation. By carefully tuning the degree of the polynomial together with the number of evaluations and checks in each share, we can ensure that any k shares contain enough evaluation points to correct the errors from the check positions, but that any $k-1$ shares do not contain enough evaluation points to determine the polynomial. This results in f being determined by slightly more than $k-1$ shares worth of evaluations. Additionally, we slightly increase the degree of the polynomial to account for the limited amount of information that an adversary can retain after deletion. See Section 7 for more details.

Share deletion and certificate verification follow the established formula. To delete a share, measure it in the Fourier basis and output the result as the certificate. To verify the certificate, check that it matches the verification key at the check positions.

Proving Adaptive Certified Deletion. Intuitively, we want to show that after the adversary deletes a share, the next share it corrupts gives it no additional information, no matter how many shares the adversary has seen so far. To formalize this, we will show that the adversary cannot distinguish between the real $\mathsf{SS\text{-}ACD}_{(k,n)}(s)$ experiment and an experiment in which each share is generated uniformly at random and independently of the others. Since the first $k-1$ shares to be corrupted do not yet uniquely determine the polynomial f, they already satisfy this. Thus, we can restrict our attention to modifying the last $n-k+1$ shares to be corrupted.

It will be useful to name the shares in the order they are corrupted or deleted. The a'th share to be corrupted is c_a, and the b'th share to be deleted is share d_b. In the (k,n) threshold case, if c_{k-1+b} is corrupted before d_b is deleted, then $C\backslash D$ has size k and is authorized, so the experiment will abort.

Techniques from BK23. We begin by recalling the techniques introduced in [BK23] to analyze 2-of-2 secret sharing with certified deletion, along with the construction. These techniques will form the starting point of our proof. To share a single-bit secret $s \in \{0,1\}$, sample random $x, \theta \leftarrow \{0,1\}^\lambda$ and output

$$\mathsf{sh}_1 = H^\theta \left|x\right\rangle, \quad \mathsf{sh}_2 = \left(\theta, s \oplus \bigoplus_{i:\theta_i=0} x_i\right), \quad \mathsf{vk} = (x, \theta),$$

where H^θ denotes applying the Hadamard gate H to the i'th register for each $i : \theta_i = 1$. Bartusek and Khurana showed that if an adversary given sh_1 produces a certificate cert such that $\mathsf{cert}_i = x_i$ for every check position $i : \theta_i = 1$, then they cannot distinguish whether $s = 0$ or $s = 1$ even if they later receive sh_2. Their approach has three main steps.

1. First, they delay the dependence of the experiment on s by initializing sh_1 to be the register $\mathcal{X}$ in $\sum_x \left|x\right\rangle^{\mathcal{X}} \otimes \left|x\right\rangle^{\mathcal{Y}}$. Later, the challenger can obtain x by measuring register $\mathcal{Y}$, and use it to derive s.

2. Second, they argue that if the adversary produces a valid deletion certificate, then sh_1 has been "almost entirely deleted", in the sense that the challenger's copy satisfies a checkable predicate with high probability. Intuitively, this predicate shows that the data positions ($\theta_i = 0$) of the challenger's copy have high joint entropy when measured in the computational basis. To show that the predicate holds, they use the fact that the adversary does not have sh_2 in their view, so sh_1 looks uniformly random. This allows a cut-and-choose argument where the locations of the check indices are determined *after* the adversary outputs its deletion certificate.

3. Finally, they show that the challenger derives a bit s that is uniformly random and independent of the adversary's view. This utilizes a result from [ABKK23] which shows that XOR is a good seedless extractor for entropy sources that satisfy the aforementioned predicate.

Adapting to Secret Sharing. As a starting point, let us try to adapt these techniques to undetectably change shares to uniformly random. For concreteness, consider the task of switching a share c_{k-1+b} to uniformly random. Although we have not yet outlined the general proof structure, we will eventually need to perform this task. We will use this starting point to gain insights that will help guide the eventual proof structure.

1. The first step is to delay the synthesis of the secret information until after the adversary outputs a deletion certificate. In our case, we will delay creating share c_{k-1+b} until after the adversary produces a valid certificate for share d_b.

   This can be achieved by sampling the first $k-1$ corrupted shares to be uniformly random, then using polynomial interpolation to prepare the rest of the shares. More concretely, consider the first corrupted $k-1$ shares $c_1, \dots, c_{k-1}$. The challenger will prepare each of these shares c_a using two registers $\mathcal{C}_{c_a}$ and $\mathcal{S}_{c_a}$, then send the share to the adversary in register $\mathcal{S}_{c_a}$. To prepare the j'th position of c_a, the experiment challenger prepares either a uniform superposition $\sum_{x\in\mathbb{K}} |x\rangle^{\mathcal{C}_{c_a,j}}$ or $\sum_{x\in\mathbb{K}} \mathsf{QFT}\,|x\rangle^{\mathcal{C}_{c_a,j}}$, depending on whether j is an evaluation position or a check position. If j is an evaluation position for share c_a, the experiment challenger copies $\mathcal{C}_{c_a,j}$ to $\mathcal{S}_{c_a,j}$ in the computational basis, yielding

$$\propto \sum_{x\in\mathbb{K}} |x\rangle^{\mathcal{S}_{c_a,j}} \otimes |x\rangle^{\mathcal{C}_{c_a,j}} ,$$

   and otherwise it copies $\mathcal{C}_{c_a,j}$ to $\mathcal{S}_{c_a,j}$ in the Fourier basis, yielding

$$\propto \sum_{x\in\mathbb{K}} \mathsf{QFT}\,|x\rangle^{\mathcal{S}_{c_a,j}} \otimes \mathsf{QFT}\,|x\rangle^{\mathcal{C}_{c_a,j}} .$$

   Note that the adversary cannot determine which positions are evaluation positions and which are check positions, since each $\mathcal{S}_{c_a,j}$ register looks maximally mixed. Also observe that $\mathcal{C}_{c_a}$ contains a copy of share c_a, and the evaluation positions in the initial $k-1$ $\mathcal{C}_{c_a}$ registers determine the polynomial f. Then, when the adversary requests to corrupt a later share, the challenger computes the evaluation points for that share by performing a polynomial interpolation using its copies of the prior shares. For reasons that will become apparent shortly, we require that share d_b is included when interpolating c_{k-1+b}. The other points may be arbitrary.

   The above procedure is actually slightly simplified; since the degree of f is slightly larger than the number of evaluation positions in $k-1$ shares, the first $k-1$ shares do not quite determine f. To remedy this, we will also initialize a small portion of *every* $\mathcal{S}_i$ to be uniformly random, before any interpolation takes place.

2. Next, we will need to show that $\mathcal{C}_{d_b}$, which contains the challenger's copy of share d_b, satisfies the deletion predicate if cert_{d_b} passes verification. This is not hard to show if d_b was generated uniformly at random, but it is not clear what happens if the adversary has some information about where the check positions are in d_b before deleting it. The first $k-1$ shares are uniformly random in the original experiment, so this is not a problem for any share which is deleted before c_k is corrupted. However, later shares depend on earlier shares, potentially leaking information about the check positions. This will be our first barrier to overcome.

3. Finally, we will need to show that interpolating c_{k-1+b} using $\mathcal{C}_{d_b}$ produces a uniformly random value whenever $\mathcal{C}_{d_b}$ satisfies the deletion predicate. In other words, **polynomial interpolation should double as a good randomness extractor from deleted shares**. Fortunately, polynomial interpolation is a matrix multiplication, and we have intuition from the classical setting that linear operations are good randomness extractors. Since a small amount of every share is uniformly random "for free", the extractor needs to produce only slightly less than a full share's worth of evaluations to produce c_{k-1+b}. This is our second technical contribution, which we will revisit in Section 2.3.

In step two, we seem to need the evaluation points in d_b to look like the check positions when d_b is deleted, i.e. they should be uniformly random and independent of the rest of the adversary's view. A natural approach to ensure this is to modify the shares to uniformly random round-by-round over a series of hybrid experiments. In hybrid i, the first $k-1+i$ shares to be corrupted are uniformly random. Since d_{i+1} must be deleted before c_{k+i} is corrupted (or else the experiment aborts), d_{i+1} must have been one of the uniformly random shares. Now we can apply the cut-and-choose argument to show that $\mathcal{C}_{d_i}$ satisfies the deletion predicate, thereby satisfying the extractor requirements to change c_{k+i} to be uniformly random and reach hybrid $i+1$. The first $k-1$ shares are already uniformly random, which gives us an opening to begin making modifications in round k.

Unfortunately, the strategy of modifying the shares one-by-one to be uniformly random has a major flaw. In particular, the challenger needs to produce additional polynomial evaluations whenever the adversary wishes to corrupt another share, which it does via interpolation. Recall that in order to claim that c_{k-1+i} is indistinguishable from random, we apply an extractor which uses register $\mathcal{C}_{d_i}$ as its source of entropy. But in order to invoke the security of the extractor, it seems that we cannot allow the challenger to re-use $\mathcal{C}_{d_i}$ when interpolating later points, as this might leak additional information about the source to the adversary.

To get around this issue, we might require that the challenger never uses $\mathcal{C}_{d_i}$ again to interpolate later points. However, the randomness extractor outputs *less* randomness than the size of a share. Intuitively, this occurs because the adversary can avoid fully deleting the source share d_i by guessing the location of a very small number of check positions.[4] Imperfect deletion limits the entropy of the source, which in turn limits the size of the extractor output. Now, since the challenger started with *exactly* enough evaluations to uniquely determine f, if we take away the points in $\mathcal{C}_{d_i}$ then there are no longer enough evaluation points remaining to create the rest of the shares, even given the newly interpolated points.

Predicates First, Replacement Later. Intuitively, the problem outlined above arises from the possibility that the adversary receives additional information about earlier shares from the later ones, since they are all correlated through the definition of the polynomial f. Our first idea to overcome this issue is to prove that the predicate holds for all rounds *before* switching any shares to uniformly random. In particular, we will consider a sequence of hybrid experiments where in the i'th hybrid, the challenger performs the predicate measurement on $\mathcal{C}_{d_i}$ after receiving and verifying the corresponding certificate. If the measurement rejects, the experiment immediately aborts and outputs $\perp$.

If we can undetectably reach the last hybrid experiment, then it is possible to undetectably replace every share with uniform randomness by working backwards. In the last hybrid experiment, either the predicate holds on the challenger's copy $\mathcal{C}_{d_{n-k+1}}$ of share d_{n-k+1} or the experiment aborts. In either case, the last share c_n to be corrupted can be undetectably replaced with uniform randomness. Since no further shares

[4]One may wonder whether it is possible to instead use coset states for the shares, which provide guarantees of *full* deletion [BGK+24]. Unfortunately, coset states induce errors which are the sum of a small number of uniformly random vectors. It is not clear how to correct these errors to reconstruct the secret without prior knowledge of the underlying subspace. However, encoding the subspace brings us back to the original problem of secret sharing with adaptive certified deletion.

are interpolated, we no longer run into the issue of re-using the randomness source $\mathcal{C}_{d_{n-k+1}}$, allowing the challenger to safely complete the experiment. Then, once c_n is uniformly random, the challenger no longer needs to interpolate shares after c_{n-1}, so c_{n-1} can also be replaced with uniform randomness. This argument can be continued until all shares are replaced.

To undetectably transition from hybrid i to hybrid $i+1$, we must show that the predicate measurement returns success with high probability on $\mathcal{C}_{d_{i+1}}$. This is not hard to show for shares which are deleted *before* the k'th share is corrupted, because the deleted shares must be one of the shares which were generated uniformly at random. However, it is not clear how to show for shares which are deleted *after* the k'th share is corrupted, since this seems to require replacing c_k with uniform randomness, which brings us back to our previous problem.

Chaining Deletions via Truncated Experiments. Our second insight is the observation that the result of a measurement made when d_i is deleted *is independent of later operations*. Thus, when arguing about the probability that the predicate measurement accepts on $\mathcal{C}_{d_i}$, it is sufficient to argue about the truncated experiment that ends immediately after the predicate measurement on $\mathcal{C}_{d_i}$. Crucially, the adversary cannot corrupt share c_{k+i} in the truncated experiment without causing the experiment to abort due to $|C \backslash D| \geq k$. Instead, c_{k-1+i} is the last share that can be corrupted. This prevents the catastrophic re-use of $\mathcal{C}_{d_i}$ after share c_{k-1+i} is constructed.

Let us assume that we have already shown that the deletion predicate measurement accepts on $\mathcal{C}_{d_i}$ with high probability; for example, this clearly holds for d_1, which must be corrupted before c_k is corrupted. How likely it is to accept on $\mathcal{C}_{d_{i+1}}$? Say the deletion predicate measurement accepts on $\mathcal{C}_{d_i}$. Then we can invoke the extractor to undetectably replace share c_{k-1+i} with uniform randomness in the truncated game, since no further shares are corrupted before the game ends. We can use similar logic to replace each of the first $k-1+i$ shares to be corrupted in the truncated game. At this point, the adversary has no choice but to delete a uniformly random share as d_{i+1}, so we can apply a cut-and-choose argument to show that the predicate holds with high probability on $\mathcal{C}_{d_{i+1}}$ This argument can be repeated inductively to show that the predicate holds in each of the polynomially many rounds.

Recap of the First Challenge. In summary, the first challenge to address in proving adaptive certified deletion is the possibility of later shares leaking information about prior shares through the re-use of $\mathcal{C}_{d_b}$ in interpolation. This prevents directly replacing each share with uniform randomness. To sidestep this issue, we first argue that every $\mathcal{C}_{d_b}$ is a good source of entropy using a series of games which end after d_b is deleted. Then even if $\mathcal{C}_{d_b}$ is used to interpolate both share c_{k-1+b} and c_{k+b}, we can rely on the entropy from $\mathcal{C}_{d_{b+1}}$ to mask its re-usage when interpolating c_{k+b}.

2.3 High Rate Seedless Extractors from Quantum Sources of Entropy

The final task to finish the proof of adaptive certified deletion security in the previous section is to show that polynomial interpolation is a good randomness extractor for entropy sources formed by deleted shares. Although polynomial interpolation arises quite naturally in our construction, there are additional technical reasons why it would be difficult to design a construction for adaptive certified deletion using existing extractors.

If we were to design a scheme using a *seeded* extractor, as done in [BI20], then every deletion would need to be independent of the seed to avoid the entropy source depending on the seed. However, as we saw with the no-signaling construction, safely encoding the seed seems to already require secret sharing with adaptive certified deletion. [BK23] makes use of the seedless XOR extractor developed by [ABKK23] to avoid a similar problem. Unfortunately, the XOR extractor produces only a single bit from a relatively large input. In the case of threshold secret sharing, the extractor must use the randomness produced by deleting a single share to extract an output which is only slightly smaller than a full share.

To address this need, we give a new family of seedless randomness extractors for quantum entropy sources with high rate. These constructions have connections to linear error-correcting codes and may be of independent interest.

A Family of Extractors. The input of the extractor is a vector of M elements of a finite field $\mathbb{F}$, and the output is a vector of m elements of $\mathbb{F}$. The source consists of a register $\mathcal{X}$ which may be arbitrarily entangled with a side-information register $\mathcal{A}$. If the register $\mathcal{X}$ is in superposition over Fourier basis vectors with Hamming weight $\leq (M-m)/2$ in $\mathbb{F}$, then we can argue that the output $\mathsf{Extract}(\mathcal{X})$ is uniformly random, even given $\mathcal{A}$.[5]

The extractor family consists of matrices $R \in \mathbb{F}^{m\times M}$ such that every set of m columns of R are linearly independent. In other words, R is a parity check matrix for a linear error-correcting code with distance at least m. An extractor R is applied by coherently multiplying R with the source register $\mathcal{X}$ in the computational basis and writing the result to the output register.

Application to Polynomial Interpolation. This family generalizes both the XOR extractor and polynomial interpolation. The XOR extractor can be represented as the all-ones matrix with one row. Each column is non-zero, so the extractor can produce a one-bit output. In the case of polynomial interpolation, we can write the linear interpolation operator for a polynomial f as a matrix R with $\mathsf{deg}(f)+1$ columns and a number of rows equal to the number of points being interpolated. R is a sub-matrix of a parity check matrix for a Reed-Solomon code, so it falls into the new extractor family. In fact, our result shows that any subset of columns in a polynomial interpolation matrix forms a good randomness extractor for an appropriate randomness source. When interpolating a share c_{k-1+b}, we can write the interpolation matrix as $R = [R_1|R_2]$, where R_2 is applied to d_b and R_1 is applied to the other points x on the polynomial. Then the new share is $c_{k-1+b} = R_1x + R_2d_b$. If d_b has satisfies the deletion predicate, then our extractor result shows that R_2d_b is uniformly random. Thus, the newly interpolated share c_b is also uniformly random.

Removing Entanglement. A downside of using polynomials for secret sharing is that each evaluation point exists in field $\mathbb{F}$ whose size scales with the total number of distinct points that must be defined on the polynomial. For example, $\mathbb{F}$ might be $\mathbb{Z}_p$ for some prime $p > nt$, where t is the number of evaluations per share. Using the approach outlined so far, each check position must be encoded in the Fourier basis over the same field $\mathbb{K}$. However, a logical $\mathbb{Z}_p$-qudit requires $\lceil \log_2(nt+1)\rceil$ qubits, which must all be entangled to produce a Fourier basis element of $\mathbb{Z}_p$.

We show how to remove the entanglement of the construction to only use *single-qubit* states, either in the Hadamard basis or in the computational basis. We modify the construction by setting the field $\mathbb{F}$ to be the binary extension field with $2^{\lceil \log_2(nt+1)\rceil}$ elements, so that each check position consists of $\lceil \log_2(nt+1)\rceil$ qubits. Then, we *individually* set each of these qubits to be a random Hadamard basis element. The other parts of the construction remain the same. Note that computational basis vectors over $\mathbb{F}$ can be encoded as a tuple of computational basis qubits.

Proving the security of this modification requires an expansion of the extractor theorem to allow general finite fields $\mathbb{F}$, which may have p^k elements for some prime p and $k \geq 1$. A Fourier basis element for such

[5]The Hamming weight over a (potentially non-binary) finite field is being the number of nonzero entries in the vector.

a field is obtained by applying the quantum Fourier transform over the additive group of $\mathbb{F}$, which is $\mathbb{Z}_p^k$. In particular, a Fourier basis element of $\mathbb{F}$ consists of k Fourier basis elements of $\mathbb{Z}_p$. In the case where $p = 2$, these are single-qubit Hadamard basis elements.

We emphasize that the *only* change is to modify how Fourier basis elements are defined by allowing general finite fields; both the extractor family and the Hamming weight requirement remain the same (i.e. they are still defined with respect to $\mathbb{F}$, *not* $\mathbb{Z}_p$). To gain intuition about the usefulness of this statement, let us consider its application in our secret-sharing construction. Ideally, an honest deleter would measure each qubit of its share in the Hadamard basis. However, since the dealer can only verify check positions, which each consist of $\lceil \log_2(nt+1) \rceil$ qubits, we can only prove a bound on the Hamming weight of the deleted state over $\lceil \log_2(nt+1) \rceil$-sized chunks, which corresponds to $\mathbb{F}$. This matches the entropy source requirements of the theorem. On the other hand, the polynomial that the secret is encoded in is also over $\mathbb{F}$, so polynomial interpolation must take place over $\mathbb{F}$. This matches the extractor family.

2.4 Open Problems

Although our results significantly strengthen secret sharing to resist new classes of attacks, we have only scratched the surface of an area with many fascinating open problems. We mention a few of them here.

- **Adaptive Certified Deletion for General Access Structures.** Against adaptive attacks, we construct a secret sharing scheme for the special case of threshold access structures. Is it possible to construct one for *general access structures*?

- **Stronger Definitions.** We prove the security of our schemes against *either* "distributed" attacks (i.e. no-signaling security) *or* adaptive attacks. Can we (i) formulate natural security definitions that capture both types of attacks, and (ii) prove the security of secret sharing schemes under such all-encompassing definitions?

- **Public Verification.** The question of publicly verifiable certificates for encryption with certified deletion has seen significant progress recently [HMNY21, Por23, BKP23, BGK+24, KNY23]. However, the techniques used seem to require the use of a classical secret to decode the plaintext. For secret sharing with certified deletion, this secret would need to also be encoded in a manner that can be certifiably deleted, as mentioned in Section 2.2. Is it possible to construct secret sharing with *publicly verifiable* certificates of deletion?

- **Other Threshold Primitives.** Aside from secret sharing, there are many other primitives which use thresholds or other access structures. For example, in threshold signatures, any k members may non-interactively sign messages under a secret key split between n parties [DF90]. Is it possible to construct *threshold signatures or other threshold primitives* with certified deletion?

- **High Rate Commitments with Certified Deletion.** A commitment with certified deletion allows the committed message to be certifiably and information-theoretically deleted [HMNY22, BK23]. However, current approaches either work in the random oracle model or require $\Theta(\lambda)$ qubits to commit to a single bit. Our new high-rate extractor (Theorem 3) provides a promising start to reduce the commitment overhead. Unfortunately, the proof technique pioneered by [BK23] for the plain model requires guessing the committed message, which incurs a security loss that is exponential in the size of the message. Is it possible to overcome this difficulty and construct commitments with certified deletion that have are *not much larger than the committed message*?

3 Preliminaries

3.1 Quantum Computation

For any set S, an S-qudit is a quantum state in the Hilbert space spanned by $\{|s\rangle : s \in S\}$. A quantum register $\mathcal{X}$ contains some number of qubits. $|x\rangle^{\mathcal{X}}$ denotes a quantum state $|x\rangle$ stored in register $\mathcal{X}$. A classical operation f can be applied to a quantum register $\mathcal{X}$ using the map

$$|x\rangle^{\mathcal{X}} \otimes |y\rangle^{\mathcal{Y}} \mapsto |x\rangle^{\mathcal{X}} \otimes |y + f(x)\rangle^{\mathcal{Y}}$$

We denote the application of this operation as $\mathcal{Y} \leftarrow f(\mathcal{X})$.

Lemma 1 (Gentle Measurement [Win99])**.** *Let ρ^X be a quantum state and let $(\Pi, \mathbb{I} - \Pi)$ be a projective measurement on X such that $\mathsf{Tr}(\Pi\rho) \geq 1 - \delta$. Let*

$$\rho' = \frac{\Pi\rho\Pi}{\mathsf{Tr}[\Pi\rho]}$$

be the state after applying $(\Pi, \mathbb{I}-\Pi)$ to ρ and post-selecting on obtaining the first outcome. Then $\mathsf{TD}(\rho, \rho') \leq 2\sqrt{\delta}$.

Quantum Fourier Transform over Finite Groups. Let G be a finite cyclic group and let $\omega_{|G|} := \exp(\frac{2\pi i}{|G|})$ be a primitive $|G|$'th root of unity. Let $\mathcal{X}$ be a register containing a G-qudits. The quantum Fourier transform (QFT) over G applied to $\mathcal{X}$ is the operation

$$|x\rangle^{\mathcal{X}} \mapsto \sum_{y \in G} \omega_{|G|}^{xy} |y\rangle^{\mathcal{X}}$$

Any Abelian group H may be decomposed as a product of cyclic groups $G_1 \times \cdots \times G_k$. The QFT over H is the tensor of the QFTs on each of these groups. For example, if $H = G^k$, then the QFT over H is given by the operation

$$|x\rangle^{\mathcal{X}} \mapsto \sum_{y \in G^k} \omega_{|G|}^{x \cdot y} |y\rangle^{\mathcal{X}}$$

When we consider taking a QFT over a finite field, we technically mean taking the QFT over its additive group. For example, if $\mathbb{F}$ has order p^k for some prime p, then its additive group is $\mathbb{Z}_p^k$ and the QFT is the mapping above, where $G = \mathbb{Z}_p$.

Fourier transforms are closely related to roots of unity. An n'th root of unity ω is an element of $\mathbb{C}$ such that $\omega^n = 1$. ω is a *primitive* n'th root of unity if $\omega^k \neq 1$ for every $k \in [n-1]$. The following claim about the summation of roots of unity will be useful.

Claim 1. *Let G be a cyclic group and let $\omega \neq 1$ be an $|G|$'th root of unity. Then*

$$\sum_{x \in G} \omega^x = 0$$

Proof.

$$\omega \sum_{x \in G} \omega^x = \sum_{x \in G} \omega^{1+x} \tag{1}$$

$$= \sum_{z \in G} \omega^z \tag{2}$$

where $z = 1 + x \in G$. Since $\omega \neq 1$ by definition, this can only be the case if $\sum_{x \in G} \omega^x = 0$. □

3.2 Statistics

Claim 2. *[Hoeffding's Inequality [Hoe94]] Let $X_1, \ldots, X_n$ be Boolean independent random variables. Let $\mu = \mathbb{E}[\sum_{i=1}^{n} X_i]$. Then for every $\delta > 0$,*

$$\Pr\left[\left|\sum_{i=1}^{n} X_i - \mu]\right| \geq \delta\right] \leq 2\exp\left(\frac{-2\delta^2}{n}\right)$$

3.3 Polynomials and Reed-Solomon Codes

We give some useful facts about polynomials over finite fields and the related Reed-Solomon error-correcting code.

Interpolation. Lagrange interpolation gives a method of finding every point on a degree d polynomial f over any field $\mathbb{K}$, given any $d+1$ points on f [LM01]. In particular, for every degree d, there is a linear operation $\mathsf{Interpolate}_d(Y, X)$ that takes in a set of $d+1$ pairs $(x, f(x))$ and outputs $f(y)$ for each $y \in Y$. The operation to find a set of s points is described by a matrix $R \in \mathbb{K}^{s \times (d+1)}$.

Fact 1. *Let $R \in \mathbb{K}^{s \times (d+1)}$ be an interpolation matrix for a degree d polynomial. Then any s columns of R are linearly independent.*

Proof. Consider the matrix $P = [R| - I_s]$. It suffices to show that any s columns of P are linearly independent. Observe that $Py = 0$ if and only if y_i lies on the unique degree d polynomial defined by $y_1, \ldots, y_{d+1}$ for every $i \in [d+2, d+1+s]$. Assume, for the sake of contradiction, that some set of s columns of P were linearly dependent. Then there exists a y with exactly s non-zero values such that $Py = 0$. Since y is in the kernel of P, it consists of $d+1+s$ evaluations of a degree d polynomial. However, this would imply the existence of a degree d polynomial with $d+1+s-s = d+1$ zeros, which does not exist. $\square$

Reed-Solomon Codes. A Reed-Solomon code is a error correcting code based on polynomials over a finite field $\mathbb{K}$ [RS60]. Given a message $m \in \mathbb{K}^d$, it encodes m as a polynomial f with degree $d+1$, then outputs s evaluations of f as the codeword. For finite field $\mathbb{K}$ and degree d, there exists an efficient correction procedure $\mathsf{Correct}_{\mathbb{K},d}(X)$ which attempts to recover a polynomial f using a noisy set of evaluation points X, e.g. [WB86, Gao03]. If X has size s and there are $< (s-d+1)/2$ pairs $(x, y) \in X$ such that $y \neq f(x)$, then $\mathsf{Correct}_{\mathbb{K},d}(X)$ outputs f.

3.4 Secret Sharing

Classical secret sharing. We introduce the standard notion of a secret sharing scheme, which allows one party to distribute a classical secret s among n parties, such that only certain subsets of parties have the ability to reconstruct the secret s. An access structure $\mathbb{S} \subseteq \mathcal{P}([n])$ for n parties is a monotonic set of sets, i.e. if $S \in \mathbb{S}$ and $S' \supset S$, then $S' \in \mathbb{S}$. Any set of parties $S \in \mathbb{S}$ is authorized to access the secret. Secret sharing for general monotone access structures was first introduced by [ISN87].

Definition 1 (Secret Sharing for Monotone Access Structures)**.** *A secret sharing scheme is specified by a monotone access structure $\mathbb{S}$ over n parties, and consists of two classical algorithms:*

- $\mathsf{Split}_{\mathbb{S}}(s)$ *is a randomized algorithm that takes in a secret s, and outputs n shares* $\mathsf{sh}_1, \ldots, \mathsf{sh}_n$.

- $\mathsf{Reconstruct}_{\mathbb{S}}(\{\mathsf{sh}_i\}_{i\in P})$ *is a deterministic algorithm that takes in a set of shares* $\{\mathsf{sh}_i\}_{i\in P}$ *for some* $P \subseteq [n]$*, and outputs either* s *or* $\bot$.

The scheme should satisfy the following notions of correctness and security.

- ***Correctness.*** *For all subsets* $P \subseteq [n]$ *such that there exists* $S \in \mathbb{S}$ *such that* $P \subseteq S$*,*

$$\Pr\left[\mathsf{Reconstruct}(\{\mathsf{sh}_i\}_{i\in P}) = s : (\mathsf{sh}_1,\ldots,\mathsf{sh}_n) \leftarrow \mathsf{Split}_{\mathbb{S}}(s)\right] = 1.$$

- ***Privacy.*** *There exists a randomized algorithm* Sim *such that for all subsets* $P \subseteq [n]$ *such that for all* $S \in \mathbb{S}$*,* $P \not\subseteq S$*, and any* s*,*

$$\left\{\{\mathsf{sh}_i\}_{i\in P} : (\mathsf{sh}_1,\ldots,\mathsf{sh}_n) \leftarrow \mathsf{Split}_{\mathbb{S}}(s)\right\} \equiv \left\{\{\mathsf{sh}_i\}_{i\in P} : \{\mathsf{sh}_i\}_{i\in P} \leftarrow \mathsf{Sim}(P)\right\}.$$

2-out-of-2 secret sharing with certified deletion Now, we recall the definition of 2-out-of-2 secret sharing *with certified deletion* as defined by [BK23]. A 2-out-of-2 secret sharing scheme is a very special case of secret sharing where the secret is split into two shares such that both shares together determine the secret, but either share individually cannot be used to recover the secret.

Definition 2 (2-out-of-2 Secret Sharing with Certified Deletion)**.** *We augment the standard notion of secret sharing to include a deletion algorithm* $\mathsf{Delete}_{2\text{-}2}$ *and a verification algorithm* $\mathsf{Verify}_{2\text{-}2}$*. We also specify that one share is quantum and the other share is classical. Finally, we introduce a security parameter* 1^λ*, since our deletion security guarantee will be statistical rather than perfect.*

- $\mathsf{Split}_{2\text{-}2}(1^\lambda, s)$ *is a quantum algorithm that takes in the security parameter* 1^λ*, a secret* s*, and outputs a quantum share* $|\mathsf{sh}_1\rangle$*, a classical share* sh_2*, and a classical verification key* vk.
- $\mathsf{Reconstruct}_{2\text{-}2}(|\mathsf{sh}_1\rangle, \mathsf{sh}_2)$ *is a quantum algorithm that takes in two shares and outputs the secret* s.
- $\mathsf{Delete}_{2\text{-}2}(|\mathsf{sh}_1\rangle)$ *is a quantum algorithm that takes in a quantum share* $|\mathsf{sh}_1\rangle$ *and outputs a deletion certificate* cert.
- $\mathsf{Verify}_{2\text{-}2}(\mathsf{vk}, \mathsf{cert})$ *is a classical algorithm that takes in a verification key* vk *and a deletion certificate* cert *and outputs either* $\top$ *or* $\bot$.

Beyond satisfying the standard secret sharing notions of correctness and privacy (Definition 1) for the 2-out-of-2 access structure, the scheme should satisfy the following properties.

- ***Deletion correctness.*** *For all* $\lambda \in \mathbb{N}$ *and* s*,*

$$\Pr\left[\mathsf{Verify}_{2\text{-}2}(\mathsf{vk}, \mathsf{cert}) = \top : \begin{array}{r}(|\mathsf{sh}_1\rangle, \mathsf{sh}_2) \leftarrow \mathsf{Split}_{2\text{-}2}(1^\lambda, s)\\ \mathsf{cert} \leftarrow \mathsf{Delete}_{2\text{-}2}(|\mathsf{sh}_1\rangle)\end{array}\right] = 1.$$

- ***Certified deletion security.*** *Let* Adv *be an adversary,* s *be a secret, and define the experiment* $\mathsf{2SS\text{-}NSCD}(1^\lambda, \mathsf{Adv}, s)$ *as follows:*
  - *Sample* $(|\mathsf{sh}_1\rangle, \mathsf{sh}_2, \mathsf{vk}) \leftarrow \mathsf{Split}_{2\text{-}2}(1^\lambda, s)$.
  - *Run* $(\mathsf{cert}, \mathcal{R}) \leftarrow \mathsf{Adv}(1^\lambda, |\mathsf{sh}_1\rangle)$*, where* $\mathcal{R}$ *is an arbitrary output register.*
  - *If* $\mathsf{Verify}_{2\text{-}2}(\mathsf{vk}, \mathsf{cert}) = \top$*, output* $(\mathcal{R}, \mathsf{sh}_2)$*, and otherwise output* $\bot$.

Then, for any unbounded adversary Adv *and any pair of secrets* s_0, s_1*, it holds that*

$$\mathsf{TD}\left[\mathsf{2SS\text{-}NSCD}(1^\lambda, \mathsf{Adv}, s_0), \mathsf{2SS\text{-}NSCD}(1^\lambda, \mathsf{Adv}, s_1)\right] = 2^{-\Omega(\lambda)}.$$

[BK23] showed the existence of a 2-out-of-2 secret sharing scheme with certified deletion satisfying the above definition. Notice that our certified deletion security definition requires a trace distance of $2^{-\Omega(\lambda)}$. While the theorem from [BK23] only states a bound of $\mathsf{negl}(\lambda)$, a quick inspection of their proof establishes that they in fact show a bound of $2^{-\Omega(\lambda)}$.

4 High-Rate Seedless Quantum Extractors

In this section, we study seedless extraction of large amounts of entropy from a quantum source. The source of entropy comes from applying a quantum Fourier transform to a state which is "almost" a computational basis state. In particular, the source register $\mathcal{X}$ is in superposition over vectors with low Hamming weight, and may be arbitrarily entangled with a register $\mathcal{A}$ that contains side-information about the source. Previously, [ABKK23] showed that the XOR function perfectly extracts a single bit of entropy in this setting. However, in order to extract multiple bits of entropy, they resorted to the use of a random oracle. We also remark that the case of *seeded* extraction has been well-studied by, e.g. [RK05, DFL+09, BF10].

We describe a general class of extractors that produces multiple truly random elements of any finite field $\mathbb{F}$, even conditioned on the side-information register $\mathcal{A}$. In the case where the finite field has order p^k for a prime p, we show that a large amount of entropy is generated even when the quantum Fourier transform is applied by interpreting each element $x \in \mathbb{F}_{p^k}$ as a vector $\mathbf{x} \in \mathbb{F}_p^k$ and applying the transform mod p to each index (as opposed to applying the transform mod p^k directly to the field element). This feature allows the source to be prepared using less entanglement in our eventual application to secret sharing.

Notation. The Hamming weight $h_\mathbb{F}(\mathbf{v})$ of a vector $\mathbf{v} \in \mathbb{F}^M$ over a finite field $\mathbb{F}$ is its number of non-zero positions. We denote vectors $\mathbf{v}$ and matrices $\mathbf{R}$ using bold font. Since we will be working with elements which can be interpreted as elements of two different fields, we use $(\cdot)_\mathbb{F}$ to denote that the contents of the parentheses should be interpreted as elements and operations over $\mathbb{F}$. For example, $(x+y)_\mathbb{F}$ denotes addition of x and y inside the field $\mathbb{F}$. If $\mathbf{x}, \mathbf{y} \in \mathbb{F}^k$, then $(\mathbf{x}+\mathbf{y})_\mathbb{F}$ denotes vector addition. For an extension field $\mathbb{K}$ of $\mathbb{F}$, a scalar $x \in \mathbb{K}$ can also be interpreted as a vector $\mathbf{x} \in \mathbb{F}^k$. In this case, for $x, y \in \mathbb{K}$, $(\mathbf{x}\cdot\mathbf{y})_\mathbb{F}$ produces a scalar in $\mathbb{F}$. If an element can be interpreted as either a vector or a scalar, we bold it depending on the context of the first operation applied; for example, $(x\mathbf{y})_\mathbb{K}$ or $(\mathbf{x}\cdot\mathbf{z})_\mathbb{F}$ for $x \in \mathbb{K}$, $\mathbf{y} \in \mathbb{K}^n$, and $\mathbf{z} \in \mathbb{F}^k$.

Theorem 3. *Let* $\mathbb{F}$ *be a finite field of order* p^k*. Let* $X = \mathcal{X}_1, \dots, \mathcal{X}_M$ *be a register containing* M $\mathbb{F}$*-qudits, and consider any quantum state*

$$|\gamma\rangle^{\mathcal{A},\mathcal{X}} = \sum_{\mathbf{u}\in\mathbb{F}^M : h_\mathbb{F}(\mathbf{u}) < \frac{M-m}{2}} |\psi_\mathbf{u}\rangle^\mathcal{A} \otimes |\mathbf{u}+\mathbf{w}\rangle^\mathcal{X}$$

for some integer $m \le M$ *and some fixed string* $\mathbf{w} \in \mathbb{F}^M$*. Let* $\mathbf{R} \in \mathbb{F}^{m\times M}$ *be a matrix such that every set of* m *columns of* $\mathbf{R}$ *are linearly independent.*

Let $\rho^{\mathcal{A},\mathcal{Y}}$ *be the mixed state that results from the following procedure:*

1. *Apply a quantum Fourier transform over* $\mathbb{F}$*'s additive group* $\mathbb{Z}_p^k$ *to each register* $\mathcal{X}_i$*. In other words, interpret* $\mathcal{X}_i$ *as a sequence of registers* $\mathcal{X}_{i,1}, \dots, \mathcal{X}_{i,k}$ *containing* $\mathbb{Z}_p$*-qudits, then apply a quantum Fourier transform mod* p *to each* $\mathcal{X}_{i,j}$*.*

2. *Initialize a fresh register $\mathcal{Y}$, and apply the isometry $|\mathbf{x}\rangle^{\mathcal{X}} \mapsto |\mathbf{x}\rangle^{\mathcal{X}} \otimes |\mathbf{Rx}\rangle^{\mathcal{Y}}$.*

3. *Trace out register $\mathcal{X}$.*

Then

$$\rho^{\mathcal{A},\mathcal{Y}} = \mathsf{Tr}^{X}[|\gamma\rangle\langle\gamma|] \otimes \left(\frac{1}{|\mathbb{F}|^m}\sum_{\mathbf{y}\in\mathbb{F}^m}|\mathbf{y}\rangle\langle\mathbf{y}|\right).$$

Remark 1. *As an example, consider $\mathbb{F}$ to be the field with 2^n elements. Note that for the source, the Hamming weight is taken over the larger field $\mathbb{F}$, but the quantum Fourier transform is done over the individual qubits, which in this case makes it just a Hadamard gate. The extractor R operates over $\mathbb{F}$ and produces an output in $\mathbb{F}^m$.*

Proof. First, we apply the Fourier transform to $|\gamma\rangle$ to obtain

$$\sqrt{|\mathbb{F}|^{-M}} \sum_{\mathbf{u}\in\mathbb{F}^M: h_{\mathbb{F}}(\mathbf{u})<\frac{M-m}{2}} |\psi_{\mathbf{u}}\rangle^{\mathcal{A}} \otimes \sum_{\mathbf{x}\in\mathbb{F}^M} \omega_p^{((\mathbf{u}+\mathbf{w})_{\mathbb{F}}\cdot\mathbf{x})_{\mathbb{Z}_p}} |\mathbf{x}\rangle^{\mathcal{X}},$$

where ω_p is a primitive p'th root of unity. Next, after applying the extractor, but before tracing out $\mathcal{X}$, the state becomes

$$\sqrt{|\mathbb{F}|^{-M}} \sum_{\mathbf{x}\in\mathbb{F}^M} \left(\sum_{\mathbf{u}\in\mathbb{F}^M: h_{\mathbb{F}}(\mathbf{u})<\frac{M-m}{2}} \omega_p^{((\mathbf{u}+\mathbf{w})_{\mathbb{F}}\cdot\mathbf{x})_{\mathbb{Z}_p}} |\psi_{\mathbf{u}}\rangle^{\mathcal{A}}\right) \otimes |\mathbf{x}\rangle^{\mathcal{X}} \otimes |\mathbf{Rx}\rangle^{\mathcal{Y}} \tag{3}$$

$$:= \sqrt{|\mathbb{F}|^{-M}} \sum_{\mathbf{x}\in\mathbb{F}^M} |\phi_{\mathbf{x}}\rangle^{\mathcal{A}} \otimes |\mathbf{x}\rangle^{\mathcal{X}} \otimes |\mathbf{Rx}\rangle^{\mathcal{Y}}. \tag{4}$$

Since the additive group of $\mathbb{F}$ is $\mathbb{Z}_p^k$, for every $\mathbf{u},\mathbf{w}\in\mathbb{F}^M$ and $\mathbf{x}\in\mathbb{Z}_p^{kM}$, we have

$$((\mathbf{u}+\mathbf{w})_{\mathbb{F}}\cdot\mathbf{x})_{\mathbb{Z}_p} = ((\mathbf{u}+\mathbf{w})_{\mathbb{Z}_p^k}\cdot\mathbf{x})_{\mathbb{Z}_p} = ((\mathbf{u}+\mathbf{w})\cdot\mathbf{x})_{\mathbb{Z}_p}.$$

Tracing out register $\mathcal{X}$ yields

$$\rho^{\mathcal{A},\mathcal{Y}} = |\mathbb{F}|^{-M}\sum_{\mathbf{x}\in\mathbb{F}^M} |\phi_{\mathbf{x}}\rangle\langle\phi_{\mathbf{x}}| \otimes |\mathbf{Rx}\rangle\langle\mathbf{Rx}| \tag{5}$$

$$= |\mathbb{F}|^{-M} \sum_{\substack{\mathbf{y}\in\mathbb{F}^m\\ \mathbf{x}\in\mathbb{F}^M:(\mathbf{Rx})_{\mathbb{F}}=\mathbf{y}}} |\phi_{\mathbf{x}}\rangle\langle\phi_{\mathbf{x}}| \otimes |\mathbf{y}\rangle\langle\mathbf{y}| \tag{6}$$

$$= |\mathbb{F}|^{-M} \sum_{\substack{\mathbf{y}\in\mathbb{F}^m\\ \mathbf{x}\in\mathbb{F}^M:(\mathbf{Rx})_{\mathbb{F}}=\mathbf{y}}} \left(\sum_{\substack{\mathbf{u}_1,\mathbf{u}_2\in\mathbb{F}^M:\\ h_{\mathbb{F}}(\mathbf{u}_1),h_{\mathbb{F}}(\mathbf{u}_2)\le\frac{M-m}{2}}} \omega_p^{((\mathbf{u}_1+\mathbf{w})\cdot\mathbf{x})_{\mathbb{Z}_p}} \overline{\omega_p^{((\mathbf{u}_2+\mathbf{w})\cdot\mathbf{x})_{\mathbb{Z}_p}}} |\phi_{\mathbf{u}_1}\rangle\langle\phi_{\mathbf{u}_2}|\right) \otimes |\mathbf{y}\rangle\langle\mathbf{y}| \tag{7}$$

$$= \sum_{\substack{\mathbf{u}_1,\mathbf{u}_2\in\mathbb{F}^M:\\ h_{\mathbb{F}}(\mathbf{u}_1),h_{\mathbb{F}}(\mathbf{u}_2)\le\frac{M-m}{2}}} |\phi_{\mathbf{u}_1}\rangle\langle\phi_{\mathbf{u}_2}| \otimes \left(|\mathbb{F}|^{-M}\sum_{\mathbf{y}\in\mathbb{F}^m}|\mathbf{y}\rangle\langle\mathbf{y}| \sum_{\mathbf{x}\in\mathbb{F}^M:(\mathbf{Rx})_{\mathbb{F}}=\mathbf{y}} \omega_p^{((\mathbf{u}_1+\mathbf{w})\cdot\mathbf{x}-(\mathbf{u}_2+\mathbf{w})\cdot\mathbf{x})_{\mathbb{Z}_p}}\right) \tag{8}$$

$$= \sum_{\substack{\mathbf{u}_1,\mathbf{u}_2\in\mathbb{F}^M: \\ h_{\mathbb{F}}(\mathbf{u}_1),h_{\mathbb{F}}(\mathbf{u}_2)\le\frac{M-m}{2}}} |\phi_{\mathbf{u}_1}\rangle\langle\phi_{\mathbf{u}_2}| \otimes \left(|\mathbb{F}|^{-M} \sum_{\mathbf{y}\in\mathbb{F}^m} |\mathbf{y}\rangle\langle\mathbf{y}| \sum_{\mathbf{x}\in\mathbb{F}^M:(\mathbf{R}\mathbf{x})_{\mathbb{F}}=\mathbf{y}} \omega_p^{((\mathbf{u}_1-\mathbf{u}_2)\cdot\mathbf{x})_{\mathbb{Z}_p}} \right). \tag{9}$$

Next, we apply Claim 3, proven below, to show that every $|\phi_{\mathbf{u}_1}\rangle\langle\phi_{\mathbf{u}_2}|$ term where $\mathbf{u}_1 \neq \mathbf{u}_2$ has coefficient 0. To see this, consider any such $\mathbf{u}_1, \mathbf{u}_2$ and the value $\mathbf{u} = (\mathbf{u}_1 - \mathbf{u}_2)_{\mathbb{Z}_p} = (\mathbf{u}_1 - \mathbf{u}_2)_{\mathbb{F}}$. Condition 1 is satisfied by $\mathbf{u}$ since $\mathbf{u}_1 \neq \mathbf{u}_2$. Condition 2 is satisfied since $h_{\mathbb{F}}(\mathbf{u}) \le h_{\mathbb{F}}(\mathbf{u}_1) + h_{\mathbb{F}}(\mathbf{u}_2) \le M - m$, so there are at least m indices of $\mathbf{u}$ which are zero. Finally, condition 3 is satisfied since any m columns of R are linearly independent.

Finally, noting that if $\mathbf{u}_1 = \mathbf{u}_2$, then the coefficient of $|\phi_{\mathbf{u}_1}\rangle\langle\phi_{\mathbf{u}_1}| \otimes |\mathbf{y}\rangle\langle\mathbf{y}|$ is the number of solutions to $(\mathbf{R}\mathbf{x})_{\mathbb{F}} = \mathbf{y}$, which is $|\mathbb{F}|^{M-m}$, we conclude that

$$\rho^{\mathcal{A},\mathcal{Y}} = \sum_{\mathbf{u}\in\mathbb{F}^M:h_{\mathbb{F}}(\mathbf{u})\le\frac{M-m}{2}} |\phi_{\mathbf{u}}\rangle\langle\phi_{\mathbf{u}}| \otimes \left(|\mathbb{F}|^{-m} \sum_{\mathbf{y}\in\mathbb{F}^m} |\mathbf{y}\rangle\langle\mathbf{y}| \right) \tag{10}$$

$$= \mathsf{Tr}^X[|\gamma\rangle\langle\gamma|] \otimes \left(|\mathbb{F}|^{-m} \sum_{\mathbf{y}\in\mathbb{F}^m} |\mathbf{y}\rangle\langle\mathbf{y}| \right). \tag{11}$$

□

Claim 3. *Let $\mathbf{u} \in \mathbb{F}^M$ and $\mathbf{y} \in \mathbb{F}^m$, and suppose that*

1. $\mathbf{u}_i \neq 0$ *for some index i.*
2. *There exists a set $J \subseteq [0, \ldots, M-1]$ of size m such that for every $j \in J$, $\mathbf{u}_j = 0$.*
3. *The submatrix $\mathbf{R}_J$ consisting of the columns of $\mathbf{R}$ corresponding to J has full rank.*

Then

$$\sum_{\mathbf{x}\in\mathbb{F}^M:(\mathbf{R}\mathbf{x})_{\mathbb{F}}=\mathbf{y}} \omega_p^{(\mathbf{u}\cdot\mathbf{x})_{\mathbb{Z}_p}} = 0.$$

Remark 2. *We note that in the case that $\mathbb{F} = \mathbb{F}_p$, then the above expression actually holds for any $\mathbf{u} \notin$ $\mathsf{rowspan}(\mathbf{R})$, which follows from a standard argument. The three conditions above do imply that $\mathbf{u} \notin$ $\mathsf{rowspan}(\mathbf{R})$, but are more restrictive. We take advantage of the extra restrictions in order to prove that the expression holds even in the case where $\mathbb{F}$ is an extension field of $\mathbb{F}_p$.*

Proof. Our strategy will be to partition the affine subspace $S_{\mathbf{y}} = \{\mathbf{x} \in \mathbb{F}^M : (\mathbf{R}\mathbf{x})_{\mathbb{F}} = \mathbf{y}\}$ into parallel lines, and then claim that the sum over each line is 0.

To begin, define a vector $\mathbf{z} \in \mathbb{F}^M$ so that

- $\mathbf{z}_i = 1$,
- $\mathbf{z}_j = 0$ for all $j \notin J \cup \{i\}$, and
- $(\mathbf{R}\mathbf{z})_{\mathbb{F}} = 0^m$,

which is possible because the $m \times m$ submatrix $\mathbf{R}_J$ has full rank. By construction, we have that for any $c \in \mathbb{F}$,

$$(\mathbf{u} \cdot (c\mathbf{z})_{\mathbb{F}})_{\mathbb{Z}_p} = \left(\mathbf{u}_i \cdot (c\mathbf{z}_i)_{\mathbb{F}} + \sum_{j \in J} \mathbf{u}_j \cdot (c\mathbf{z}_j)_{\mathbb{F}} + \sum_{j \notin J \cup \{i\}} \mathbf{u}_j \cdot (0)_{\mathbb{F}} \right)_{\mathbb{Z}_p} = (\mathbf{u}_i \cdot \mathbf{c})_{\mathbb{Z}_p}, \tag{12}$$

where note that in the final expression, $\mathbf{u}_i$ and $\mathbf{c}$ are interpreted as vectors in $\mathbb{Z}_p^k$.

Now, fix any $\mathbf{x} \in S_{\mathbf{y}}$ and $c \in \mathbb{F}$. Then we have that $(\mathbf{x} + c\mathbf{z})_{\mathbb{F}} \in S_{\mathbf{y}}$, since $(\mathbf{R}(\mathbf{x} + c\mathbf{z}))_{\mathbb{F}} = \mathbf{y} + 0$. Therefore, we can partition $S_{\mathbf{y}}$ into one-dimensional cosets (lines) of the form $\{\mathbf{x} + c\mathbf{z}\}_{c \in \mathbb{F}}$.

We now show that the sum over any particular coset is 0, i.e. that for any $\mathbf{x} \in S_{\mathbf{y}}$,

$$\sum_{c \in \mathbb{F}} \omega_p^{(\mathbf{u} \cdot (\mathbf{x} + c\mathbf{z})_{\mathbb{F}})_{\mathbb{Z}_p}} = 0.$$

Since the additive group of $\mathbb{F}$ is $\mathbb{Z}_p^k$, by Eq. (12) we have that

$$\begin{aligned}(\mathbf{u} \cdot (\mathbf{x} + c\mathbf{z})_{\mathbb{F}})_{\mathbb{Z}_p} &= (\mathbf{u} \cdot \mathbf{x} + \mathbf{u} \cdot (c\mathbf{z})_{\mathbb{F}})_{\mathbb{Z}_p} \\ &= (\mathbf{u} \cdot \mathbf{x} + \mathbf{u}_i \cdot \mathbf{c})_{\mathbb{Z}_p}\end{aligned}$$

We now view $\mathbf{u}_i \in \mathbb{F}$ as a vector $\mathbf{u}_i = u_{i,0}, \ \ldots, \ u_{i,k-1} \in \mathbb{Z}_p^k$. In particular, since $\mathbf{u}_i \neq 0$, there exists an index t such that $u_{k,t} \neq 0 \in \mathbb{Z}_p$. By also interpreting $\mathbf{c} \in \mathbb{F}$ as an element of $\mathbb{Z}_p^k$, we can decompose it as $\mathbf{c} = (\mathbf{c}' + c_t \mathbf{e}_t)_{\mathbb{Z}_p}$, where $\mathbf{c}' \in \mathbb{Z}_p^k$ is such that $\mathbf{c}'_t = 0$, where $c_t \in \mathbb{Z}_p$, and where $\mathbf{e_t} \in \mathbb{Z}_p^k$ is the t'th standard basis vector. Therefore

$$(\mathbf{u} \cdot (\mathbf{x} + c\mathbf{z})_{\mathbb{F}})_{\mathbb{Z}_p} = (\mathbf{u} \cdot \mathbf{x} + \mathbf{u}_i \cdot \mathbf{c}' + u_{i,t} c_t)_{\mathbb{Z}_p}.$$

Since $u_{i,t} \neq 0$ and ω_p is a primitive p'th root of unity, we know that $\omega_p^{u_{i,t}} \neq 1$ is a p'th root of unity. Therefore by Claim 1,

$$\begin{aligned}\sum_{c \in \mathbb{F}} \omega_p^{(\mathbf{u} \cdot (\mathbf{x} + c\mathbf{z})_{\mathbb{F}})_{\mathbb{Z}_p}} &= \sum_{\mathbf{c}' \in \mathbb{Z}_p^k : \mathbf{c}'_t = 0} \omega_p^{(\mathbf{u} \cdot \mathbf{x} + \mathbf{u}_i \cdot \mathbf{c}')_{\mathbb{Z}_p}} \cdot \sum_{c_t \in \mathbb{Z}_p} \left(\omega_p^{u_{i,t}}\right)^{c_t} &(13)\\ &= \sum_{\mathbf{c}' \in \mathbb{Z}_p^k : \mathbf{c}'_t = 0} \omega_p^{(\mathbf{u} \cdot \mathbf{x} + \mathbf{u}_i \cdot \mathbf{c}')_{\mathbb{Z}_p}} \cdot 0 &(14)\\ &= 0. &(15)\end{aligned}$$

□

5 Definitions of Secret Sharing with Certified Deletion

A secret sharing scheme with certified deletion augments the syntax of a secret sharing scheme with additional algorithms to delete shares and verify deletion certificates. We define it for general access structures. As described in Section 3.4, an access structure $\mathbb{S} \subseteq \mathcal{P}([n])$ for n parties is a monotonic set of sets, i.e. if $S \in \mathbb{S}$ and $S' \supset S$, then $S' \in \mathbb{S}$. Any set of parties $S \in \mathbb{S}$ is authorized to access the secret. A simple example of an access structure is the threshold structure, where any set of at least k parties is authorized to access the secret. We denote this access structure as (k, n).

Definition 3 (Secret Sharing with Certified Deletion)**.** *A secret sharing scheme with certified deletion is specified by a monotone access structure $\mathbb{S}$ over n parties, and consists of four algorithms:*

- $\mathsf{Split}_{\mathbb{S}}(1^\lambda, s)$ *takes in a secret s, and outputs n share registers $\mathcal{S}_1, \ldots, \mathcal{S}_n$ and a verification key* vk.
- $\mathsf{Reconstruct}_{\mathbb{S}}(\{S_i\}_{i \in P})$ *takes in set of share registers for some $P \subseteq [n]$, and outputs either s or $\bot$.*
- $\mathsf{Delete}_{\mathbb{S}}(\mathcal{S}_i)$ *takes in a share register and outputs a certificate of deletion* cert.
- $\mathsf{Verify}_{\mathbb{S}}(\mathsf{vk}, i, \mathsf{cert})$ *takes in the verification key* vk*, an index i, and a certificate of deletion* cert*, and outputs either $\top$ (indicating accept) or $\bot$ (indicating reject).*

Definition 4 (Correctness of Secret Sharing with Certified Deletion)**.** *A secret sharing scheme with certified deletion must satisfy two correctness properties:*

- ***Reconstruction Correctness.*** *For all $\lambda \in \mathbb{N}$ and all sets $S \in \mathbb{S}$,*

$$\Pr\left[\mathsf{Reconstruct}_{\mathbb{S}}(\{\mathcal{S}_i\}_{i \in S}) : (\mathcal{S}_1, \ldots, \mathcal{S}_n, \mathsf{vk}) \leftarrow \mathsf{Split}_{\mathbb{S}}(1^\lambda, s)\right] = 1.$$

- ***Deletion Correctness.*** *For all $\lambda \in \mathbb{N}$ and all $i \in [n]$,*

$$\Pr\left[\mathsf{Verify}_{\mathbb{S}}(\mathsf{vk}, i, \mathsf{cert}) = \top : \begin{array}{r} (\mathcal{S}_1, \ldots, \mathcal{S}_n, \mathsf{vk}) \leftarrow \mathsf{Split}_{\mathbb{S}}(1^\lambda, s) \\ \mathsf{cert} \leftarrow \mathsf{Delete}_{\mathbb{S}}(\mathcal{S}_i) \end{array}\right] = 1.$$

The standard notion of security for secret sharing requires that no set of unauthorized shares $S \notin \mathbb{S}$ reveals any information about the secret (see Section 3.4). We next present our notion of *no-signaling certified deletion security*. Here, the shares are partitioned into unauthorized sets, and different parts of the adversary operate on each partition, potentially deleting some number of shares from each. The different parts of the adversary are allowed to share entanglement, but are not allowed to signal. If the adversaries jointly produce a valid certificate for at least one share from every authorized set, then we require that the joint residual state of *all* of the adversaries contains no (or negligible) information about the secret. Observe that this notion of security is at least as strong as the standard notion of security for secret sharing (if we relax to statistical rather than perfect security). Indeed, if the standard notion does not hold, and thus there is some unauthorized set S that leaks information about the secret, then the adversary would be able to win the certified deletion game by honestly deleting every share except for those in S.

Definition 5 (No-Signaling Certified Deletion Security for Secret Sharing)**.** *Let $P = (P_1, \ldots, P_\ell)$ be a partition of $[n]$, let $|\psi\rangle$ be an ℓ-part state on registers $\mathcal{R}_1, \ldots, \mathcal{R}_\ell$, and let $\mathsf{Adv} = (\mathsf{Adv}_1, \ldots, \mathsf{Adv}_\ell)$ be an ℓ-part adversary. Define the experiment $\mathsf{SS\text{-}NSCD}_{\mathbb{S}}(1^\lambda, P, |\psi\rangle, \mathsf{Adv}, s)$ as follows:*

1. *Sample $(\mathcal{S}_1, \ldots, \mathcal{S}_n, \mathsf{vk}) \leftarrow \mathsf{Split}_{\mathbb{S}}(1^\lambda, s)$.*
2. *For each $t \in [\ell]$, run $(\{\mathsf{cert}_i\}_{i \in P_t}, \mathcal{R}'_t) \leftarrow \mathsf{Adv}_t(\{\mathcal{S}_i\}_{i \in P_t}, \mathcal{R}_t)$, where $\mathcal{R}'_t$ is an arbitrary output register.*
3. *If for all $S \in \mathbb{S}$, there exists $i \in S$ such that $\mathsf{Verify}_{\mathbb{S}}(\mathsf{vk}, i, \mathsf{cert}_i) = \top$, then output $(\mathcal{R}'_1, \ldots, \mathcal{R}'_\ell)$, and otherwise output $\bot$.*

A secret sharing scheme for access structure $\mathbb{S}$ has no-signaling certified deletion security if for any "admissible" partition $P = (P_1, \ldots, P_\ell)$ (i.e. for all $P_t \in P$ and $S \in \mathbb{S}$, $P_t \not\subseteq S$), any ℓ-part state $|\psi\rangle$, any (unbounded) ℓ-part adversary Adv*, and any pair of secrets s_0, s_1,*

$$\mathsf{TD}[\mathsf{SS\text{-}NSCD}_{\mathbb{S}}(1^\lambda, P, |\psi\rangle, \mathsf{Adv}, s_0),\ \mathsf{SS\text{-}NSCD}_{\mathbb{S}}(1^\lambda, P, |\psi\rangle, \mathsf{Adv}, s_1)] = \mathsf{negl}(\lambda).$$

Next, we present an alternative definition which allows the adversary to start by corrupting some unauthorized set, and then continue to adaptively delete some shares and corrupt new parties, as long as the total set of parties corrupted minus the set of shares deleted is unauthorized. Similarly to the previous definition, adaptive certified deletion for secret sharing subsumes the standard notion of security for secret sharing.

Definition 6 (Adaptive Certified Deletion for Secret Sharing)**.** *Let* Adv *be an adversary with internal register $\mathcal{R}$ which is initialized to a state $|\psi\rangle$, let $\mathbb{S}$ be an access structure, and let s be a secret. Define the experiment* $\mathsf{SS\text{-}ACD}_{\mathbb{S}}(1^\lambda, |\psi\rangle, \mathsf{Adv}, s)$ *as follows:*

1. *Sample $(\mathcal{S}_1, \ldots, \mathcal{S}_n, \mathsf{vk}) \leftarrow \mathsf{Split}_{\mathbb{S}}(1^\lambda, s)$. Initialize the corruption set $C = \emptyset$ and the deleted set $D = \emptyset$.*
2. *In each round i, the adversary may do one of three things:*
   (a) *End the experiment by outputting a register $\mathcal{R} \leftarrow \mathsf{Adv}(\{\mathcal{S}_j\}_{j\in C}, \mathcal{R})$.*
   (b) *Delete a share by outputting a certificate cert_i, an index $j_i \in [n]$, and register $(\mathsf{cert}_i, j_i, \mathcal{R}) \leftarrow \mathsf{Adv}(\{\mathcal{S}_j\}_{j\in C}, \mathcal{R})$. When the adversary chooses this option, if $\mathsf{Verify}_{\mathbb{S}}(\mathsf{vk}, j_i, \mathsf{cert}_i)$ outputs $\top$, then add j_i to D. Otherwise, immediately abort the experiment and output $\bot$.*
   (c) *Corrupt a new share by outputting an index $j_i \in [n]$ and register $(j_i, \mathcal{R}) \leftarrow \mathsf{Adv}(\{\mathcal{S}_j\}_{j\in C}, \mathcal{R})$. When the adversary chooses this option, add j_i to C. If $C \backslash D \in \mathbb{S}$, immediately abort the experiment and output $\bot$.*
3. *Output $\mathcal{R}$, unless the experiment has already aborted.*

A secret sharing scheme for access structure $\mathbb{S}$ has adaptive certified deletion security if for any (unbounded) adversary Adv*, any state $|\psi\rangle$, and any pair of secrets (s_0, s_1),*

$$\mathsf{TD}[\mathsf{SS\text{-}ACD}_{\mathbb{S}}(1^\lambda, |\psi\rangle, \mathsf{Adv}, s_0),\ \mathsf{SS\text{-}ACD}_{\mathbb{S}}(1^\lambda, |\psi\rangle, \mathsf{Adv}, s_1)] = \mathsf{negl}(\lambda)$$

It will also be convenient to establish some notation for the order of the corrupted and deleted shares. Let c_a be the a'th share to be corrupted (i.e. added to C) and let d_b be the b'th share to be deleted (i.e. added to D).

6 Secret Sharing with No-Signaling Certified Deletion

In this section, we'll show how to combine any classical secret sharing scheme $(\mathsf{CSplit}_{\mathbb{S}}, \mathsf{CReconstruct}_{\mathbb{S}})$ (Definition 1) for access structure $\mathbb{S} \in \mathcal{P}([n])$ with a 2-out-of-2 secret sharing scheme with certified deletion $(\mathsf{Split}_{2\text{-}2}, \mathsf{Reconstruct}_{2\text{-}2}, \mathsf{Delete}_{2\text{-}2}, \mathsf{Verify}_{2\text{-}2})$ (Definition 2) in order to obtain a secret sharing scheme for $\mathbb{S}$ that satisfies no-signaling certified deletion security (Definition 5).

Theorem 4. *The construction given in Fig. 1 satisfies no-signaling certified deletion security (Definition 5).*

$\underline{\mathsf{Split}_{\mathbb{S}}(1^\lambda, s)}$

- Sample $(\mathsf{sh}_1, \ldots, \mathsf{sh}_n) \leftarrow \mathsf{CSplit}_{\mathbb{S}}(s)$.
- Set $\kappa = \max\{\lambda, n\}^2$, and for each $i \in [n]$, sample $(|\mathsf{qsh}_i\rangle, \mathsf{csh}_i, \mathsf{vk}_i) \leftarrow \mathsf{Split}_{2\text{-}2}(1^\kappa, \mathsf{sh}_i)$.
- For each $i \in [n]$, sample $(\mathsf{csh}_{i,1}, \ldots, \mathsf{csh}_{i,n}) \leftarrow \mathsf{CSplit}_{\mathbb{S}}(\mathsf{csh}_i)$.
- Set $\mathsf{vk} = (\mathsf{vk}_1, \ldots, \mathsf{vk}_n)$, and initialize register $\mathcal{S}_i$ to $|\mathsf{qsh}_i\rangle, \{\mathsf{csh}_{j,i}\}_{j\in[n]}$.

$\underline{\mathsf{Reconstruct}_{\mathbb{S}}(\{\mathcal{S}_i\}_{i\in P})}$

- Parse each register $\mathcal{S}_i$ as $|\mathsf{qsh}_i\rangle, \{\mathsf{csh}_{j,i}\}_{j\in[n]}$.
- For each $i \in P$, compute $\mathsf{csh}_i \leftarrow \mathsf{CReconstruct}_{\mathbb{S}}(\{\mathsf{csh}_{i,j}\}_{j\in P})$, and output $\bot$ if the result is $\bot$.
- For each $i \in P$, compute $\mathsf{sh}_i \leftarrow \mathsf{Reconstruct}_{2\text{-}2}(|\mathsf{qsh}_i\rangle, \mathsf{csh}_i)$.
- Output $s \leftarrow \mathsf{CReconstruct}_{\mathbb{S}}(\{\mathsf{sh}_i\}_{i\in P})$.

$\underline{\mathsf{Delete}_{\mathbb{S}}(\mathcal{S}_i)}$

- Parse $\mathcal{S}_i$ as $|\mathsf{qsh}_i\rangle, \{\mathsf{csh}_{j,i}\}_{j\in[n]}$ and output $\mathsf{cert} \leftarrow \mathsf{Delete}_{2\text{-}2}(|\mathsf{qsh}_i\rangle)$.

$\underline{\mathsf{Verify}_{\mathbb{S}}(\mathsf{vk}, i, \mathsf{cert})}$

- Parse $\mathsf{vk} = (\mathsf{vk}_1, \ldots, \mathsf{vk}_n)$ and output $\mathsf{Verify}_{2\text{-}2}(\mathsf{vk}_i, \mathsf{cert})$.

Figure 1: Secret sharing with no-signaling certified deletion security for any access structure $\mathbb{S}$.

Proof. Let $\mathsf{Adv} = (\mathsf{Adv}_1, \ldots, \mathsf{Adv}_\ell)$ be any ℓ-part adversary that partitions the shares using an admissible partition $P = (P_1, \ldots, P_\ell)$ and is initialized with the ℓ-part state $|\psi\rangle$ on registers $\mathcal{R}_1, \ldots, \mathcal{R}_\ell$. Let s_0, s_1 be any two secrets, and assume for contradiction that

$$\mathsf{TD}[\mathsf{SS\text{-}NSCD}_{\mathbb{S}}(1^\lambda, P, |\psi\rangle, \mathsf{Adv}, s_0),\ \mathsf{SS\text{-}NSCD}_{\mathbb{S}}(1^\lambda, P, |\psi\rangle, \mathsf{Adv}, s_1)] = \mathsf{nonnegl}(\lambda).$$

Now, for $s \in \{s_0, s_1\}$, define a hybrid $\mathcal{H}_1(s)$ as follows.

$\underline{\mathcal{H}_1(s)}$

1. Sample $C \leftarrow \mathcal{P}([n])$.
2. Sample $(\mathsf{sh}_1, \ldots, \mathsf{sh}_n) \leftarrow \mathsf{CSplit}_{\mathbb{S}}(s)$.
3. Set $\kappa = \max\{\lambda, n\}^2$, and for each $i \in [n]$, sample $(|\mathsf{qsh}_i\rangle, \mathsf{csh}_i, \mathsf{vk}_i) \leftarrow \mathsf{Split}_{2\text{-}2}(1^\kappa, \mathsf{sh}_i)$.
4. For each $i \in [n]$, sample $(\mathsf{csh}_{i,1}, \ldots, \mathsf{csh}_{i,n}) \leftarrow \mathsf{CSplit}_{\mathbb{S}}(\mathsf{csh}_i)$.
5. Set $\mathsf{vk} = (\mathsf{vk}_1, \ldots, \mathsf{vk}_n)$, and initialize register $\mathcal{S}_i$ to $|\mathsf{qsh}_i\rangle, \{\mathsf{csh}_{j,i}\}_{j\in[n]}$.

6. For each $t \in [\ell]$, run $(\{\mathsf{cert}_i\}_{i \in P_t}, \mathcal{R}'_t) \leftarrow \mathsf{Adv}_t(\{\mathcal{S}_i\}_{i \in P_t}, \mathcal{R}_t)$.

7. Let $C^* := \{i : \mathsf{Verify}_{\text{2-2}}(\mathsf{vk}_i, i, \mathsf{cert}_i) = \top\}$. Output $(\mathcal{R}'_1, \ldots, \mathcal{R}'_\ell)$ if $C = C^*$ and $[n] \setminus C^* \notin \mathbb{S}$, and otherwise output $\bot$.

That is, $\mathcal{H}_1(s)$ is the same as $\mathsf{SS\text{-}NSCD}_{\mathbb{S}}(1^\lambda, P, |\psi\rangle, \mathsf{Adv}, s)$ except that $\mathcal{H}_1(s)$ makes a uniformly random guess C for the subset of shares for which the adversary produces a valid deletion certificate, and aborts if this guess is incorrect.

Claim 4. $\mathsf{TD}\left[\mathcal{H}_1(s_0), \mathcal{H}_1(s_1)\right] = \mathsf{nonnegl}(\lambda) \cdot 2^{-n}$.

Proof. This follows directly from the fact that $\mathcal{H}_1(s)$'s guess for C is correct with probability $1/2^n$, and, conditioned on the guess being correct, $\mathcal{H}_1(s)$ is identical to $\mathsf{SS\text{-}NSCD}_{\mathbb{S}}(1^\lambda, P, |\psi\rangle, \mathsf{Adv}, s)$. □

Now, for $s \in \{s_0, s_1\}$ and $k \in [0, \ldots, n]$, define a sequence of hybrids $\mathcal{H}_{2,k}(s)$ as follows.

$\underline{\mathcal{H}_{2,k}(s)}$

1. Sample $C \leftarrow \mathcal{P}([n])$.

2. Sample $(\mathsf{sh}_1, \ldots, \mathsf{sh}_n) \leftarrow \mathsf{CSplit}_{\mathbb{S}}(s)$.

3. Set $\kappa = \max\{\lambda, n\}^2$ and for each $i \in [n]$, if $i \leq k$ and $i \in C$, sample $(|\mathsf{qsh}_i\rangle, \mathsf{csh}_i, \mathsf{vk}_i) \leftarrow \mathsf{Split}_{\text{2-2}}(1^\kappa, \bot)$, and otherwise sample $(|\mathsf{qsh}_i\rangle, \mathsf{csh}_i, \mathsf{vk}_i) \leftarrow \mathsf{Split}_{\text{2-2}}(1^\kappa, \mathsf{sh}_i)$.

4. For each $i \in [n]$, sample $(\mathsf{csh}_{i,1}, \ldots, \mathsf{csh}_{i,n}) \leftarrow \mathsf{CSplit}_{\mathbb{S}}(\mathsf{csh}_i)$.

5. Set $\mathsf{vk} = (\mathsf{vk}_1, \ldots, \mathsf{vk}_n)$, and initialize register $\mathcal{S}_i$ to $|\mathsf{qsh}_i\rangle, \{\mathsf{csh}_{j,i}\}_{j \in [n]}$.

6. For each $t \in [\ell]$, run $(\{\mathsf{cert}_i\}_{i \in P_t}, \mathcal{R}'_t) \leftarrow \mathsf{Adv}_t(\{\mathcal{S}_i\}_{i \in P_t}, \mathcal{R}_t)$.

7. Let $C^* := \{i : \mathsf{Verify}_{\text{2-2}}(\mathsf{vk}_i, i, \mathsf{cert}_i) = \top\}$. Output $(\mathcal{R}'_1, \ldots, \mathcal{R}'_\ell)$ if $C = C^*$ and $[n] \setminus C^* \notin \mathbb{S}$, and otherwise output $\bot$.

First, note that $\mathcal{H}_1(s_0) = \mathcal{H}_{2,0}(s_0)$ and $\mathcal{H}_1(s_1) = \mathcal{H}_{2,0}(s_1)$. Next, we show the following claim.

Claim 5. $\mathcal{H}_{2,n}(s_0) \equiv \mathcal{H}_{2,n}(s_1)$.

Proof. In each experiment, if the output is not $\bot$, then we know that $[n] \backslash C$ is an unauthorized set. Moreover, the experiments do not depend on the information $\{\mathsf{sh}_i\}_{i \in C}$. Thus, the claim follows by the perfect privacy of $(\mathsf{CSplit}_{\mathbb{S}}, \mathsf{CReconstruct}_{\mathbb{S}})$, which implies that

$$\begin{aligned} &\{\{\mathsf{sh}_i\}_{i \in C \backslash [n]} : (\mathsf{sh}_1, \ldots, \mathsf{sh}_n) \leftarrow \mathsf{CSplit}_{\mathbb{S}}(s_0)\} \\ &\quad \equiv \{\{\mathsf{sh}_i\}_{i \in C \backslash [n]} : (\mathsf{sh}_1, \ldots, \mathsf{sh}_n) \leftarrow \mathsf{CSplit}_{\mathbb{S}}(s_1)\}. \end{aligned}$$

□

Finally, we show the following claim.

Claim 6. *For $s \in \{s_0, s_1\}$ and $k \in [n]$, it holds that* $\mathsf{TD}\left[\mathcal{H}_{2,k-1}(s), \mathcal{H}_{2,k}(s)\right] = 2^{-\Omega(\kappa)}$.

Proof. The only difference between these hybrids is that if $k \in C$, we switch sh_k to $\bot$ in the third step. So, suppose that $k \in C$, and consider the following reduction to the certified deletion security (Definition 2) of $(\mathsf{Split}_{2\text{-}2}, \mathsf{Reconstruct}_{2\text{-}2}, \mathsf{Delete}_{2\text{-}2}, \mathsf{Verify}_{2\text{-}2})$. This experiment is parameterized by a bit b which determines which one of two secrets the certified deletion challenger will share.

- The reduction samples $C \leftarrow \mathcal{P}([n])$ and $(\mathsf{sh}_1, \dots, \mathsf{sh}_n) \leftarrow \mathsf{CSplit}_{\mathbb{S}}(s)$, and sends $\{\mathsf{sh}_k, \bot\}$ to the challenger.
- The challenger samples $(|\mathsf{qsh}_k\rangle, \mathsf{csh}_k, \mathsf{vk}_k) \leftarrow \mathsf{Split}_{2\text{-}2}(1^\kappa, \mathsf{sh}_k)$ if $b = 0$ or $(|\mathsf{qsh}_k\rangle, \mathsf{csh}_k, \mathsf{vk}_k) \leftarrow \mathsf{Split}_{2\text{-}2}(1^\kappa, \bot)$ if $b = 1$, and sends $|\mathsf{qsh}_k\rangle$ to the reduction.
- For each $i \in [n] \backslash \{k\}$, if $i < k$ and $i \in C$, the reduction samples $(|\mathsf{qsh}_i\rangle, \mathsf{csh}_i, \mathsf{vk}_i) \leftarrow \mathsf{Split}_{2\text{-}2}(1^\kappa, \bot)$, and otherwise samples $(|\mathsf{qsh}_i\rangle, \mathsf{csh}_i, \mathsf{vk}_i) \leftarrow \mathsf{Split}_{2\text{-}2}(1^\kappa, \mathsf{sh}_i)$.
- Let $t^* \in [\ell]$ be such that $k \in P_{t^*}$. For each $i \in [n] \setminus \{k\}$, the reduction samples $(\mathsf{csh}_{i,1}, \dots, \mathsf{csh}_{i,n}) \leftarrow \mathsf{CSplit}_{\mathbb{S}}(\mathsf{csh}_i)$. Next, the reduction samples $\{\mathsf{csh}_{k,i}\}_{i \in P_{t^*}} \leftarrow \mathsf{Sim}(P_{t^*})$, where Sim is the simulator guaranteed by the privacy of $(\mathsf{CSplit}_{\mathbb{S}}, \mathsf{CReconstruct}_{\mathbb{S}})$.
- For each $i \in P_{t^*}$, initialize register $\mathcal{S}_i$ to $|\mathsf{qsh}_i\rangle, \{\mathsf{csh}_{j,i}\}_{j \in [n]}$.
- The reduction runs $(\{\mathsf{cert}_i\}_{i \in P_{t^*}}, \mathcal{R}'_{t^*}) \leftarrow \mathsf{Adv}_{t^*}(\{\mathcal{S}_i\}_{i \in P_{t^*}}, \mathcal{R}_{t^*})$, and sends cert_k to the challenger.
- The challenger checks whether $\mathsf{Verify}_{2\text{-}2}(\mathsf{vk}_k, \mathsf{cert}_k) = \top$. If so, the challenger returns csh_k, and otherwise the experiment aborts and outputs $\bot$.
- The reduction samples $\{\mathsf{csh}_{k,i}\}_{i \in [n] \backslash P_{t^*}}$ conditioned on the joint distribution of $(\mathsf{csh}_{k,1}, \dots, \mathsf{csh}_{k,n})$ being identical to $\mathsf{CSplit}_{\mathbb{S}}(\mathsf{csh}_k)$. This is possible due to the guarantee of $\mathsf{Sim}(P_{t^*})$, that is,
$$\begin{aligned}\left\{\{\mathsf{csh}_{k,i}\}_{i \in P_{t^*}} : (\mathsf{csh}_{k,1}, \dots, \mathsf{csh}_{k,n}) \leftarrow \mathsf{CSplit}_{\mathbb{S}}(\mathsf{csh}_k)\right\} \\ \equiv \left\{\{\mathsf{csh}_{k,i}\}_{i \in P_{t^*}} : \{\mathsf{csh}_{k,i}\}_{i \in P_{t^*}} \leftarrow \mathsf{Sim}(P_{t^*})\right\}.\end{aligned}$$
- For each $i \in [n] \setminus P_{t^*}$, the reduction initializes register $\mathcal{S}_i$ to $|\mathsf{qsh}_i\rangle, \{\mathsf{csh}_{j,i}\}_{j \in [n]}$.
- For each $t \in [\ell] \setminus \{t^*\}$, run $(\{\mathsf{cert}_i\}_{i \in P_t}, \mathcal{R}'_t) \leftarrow \mathsf{Adv}_t(\{\mathcal{S}_i\}_{i \in P_t}, \mathcal{R}_t)$.
- Let $C^* := \{i : \mathsf{Verify}_{2\text{-}2}(\mathsf{vk}_i, i, \mathsf{cert}_i) = \top\}$. The reduction outputs $(\mathcal{R}'_1, \dots, \mathcal{R}'_\ell)$ if $C = C^*$ and $[n] \setminus C^* \notin \mathbb{S}$, and otherwise outputs $\bot$.

Observe that in the case $b = 0$, the output of this experiment is identical to $\mathcal{H}_{2,k-1}(s)$ while if $b = 1$, the output of this experiment is identical to $\mathcal{H}_{2,k}(s)$. Thus, the claim follows from the certified deletion security of $(\mathsf{Split}_{2\text{-}2}, \mathsf{Reconstruct}_{2\text{-}2}, \mathsf{Delete}_{2\text{-}2}, \mathsf{Verify}_{2\text{-}2})$.

□

Thus, by combining Claim 5 and Claim 6, we have that
$$\mathsf{TD}\left[\mathcal{H}_1(s_0), \mathcal{H}_1(s_1)\right] = 2n \cdot 2^{-\Omega(\kappa)} \leq 2^{-\Omega(\{\max\{\lambda,n\}^2\})}.$$
However, this violates Claim 4, since
$$2^{-\Omega(\{\max\{\lambda,n\}^2\})} < \mathsf{nonnegl}(\lambda) \cdot 2^{-n},$$
which completes the proof.

□

7 Threshold Secret Sharing with Adaptive Certified Deletion

In this section, we show how to construct a secret sharing scheme for threshold access structures that satisfies adaptive certified deletion (Definition 6).

7.1 Construction

Our construction is given in Figure 3, which uses a set of parameters described in Figure 2. We provide some intuition about the parameter settings here.

The secret is encoded in a polynomial f of degree p. For security, we need p to be at least as large as the number of points of f that the adversary can learn. At most, the adversary can hold up to $k-1$ intact shares and the residual states of $n-k+1$ deleted shares. Each of the $k-1$ intact shares contains t' evaluations of f. Additionally, the adversary may retain some small amount of information about each of the deleted shares. We upper bound the retained information by a parameter ℓ for each share. This gives the adversary a maximum of

$$(k-1)t' + (n-k+1)\ell$$

evaluations of f, which becomes the minimum safe setting for p.

Each share will also include a number of "check positions", which contain Fourier basis states that are used for verification of deletion. The number of check positions r and upper bound ℓ are set roughly so that with overwhelming probability, the adversary can retain no more than ℓ evaluations of f when it deletes a share (more precisely, the adversary may retain a *superposition* over potentially different sets of ℓ evaluations). The reader may find it useful to think of ℓ as being the maximum number of unexamined positions in a classical string x when an adversary successfully creates a string y that matches x on r random verification indices. Finally, the total size t of each share is set so that k shares contain less than $(kt-p)/2$ check positions, which is the maximum number of errors that can be corrected in kt evaluations of a polynomial of degree p (see Section 3.3).

Theorem 5. *There exists secret sharing for threshold access structures which satisfies adaptive certified deletion.*

Proof. The construction is given in Figure 3. Deletion correctness is apparent from inspection of the construction. We prove reconstruction correctness in Lemma 2 and adaptive certified deletion security in Lemma 3. □

Lemma 2. *The construction in Figure 3 using parameters from Figure 2 has reconstruction correctness.*

Proof. The set $\{(it+j, y_{i,j})\}_{i\in P', j\in[t]}$ contains kt pairs which were obtained by measuring k shares. As mentioned in Section 3.3, if all but $e < (kt-p)/2$ of these pairs $(it+j, y_{i,j})$ satisfy $y_{i,j} = f(it+j)$, then $\mathsf{Correct}_{\mathbb{K},p}$ recovers the original polynomial f, where $f(0) = s$. The only points which do not satisfy this are the check positions, of which there are r per share, for a total of kr. Therefore for correctness, we require that

$$\begin{aligned} 2kr &< kt - p && (16)\\ &= kt - (k-1)(t-r) - (n-k+1)\ell && (17)\\ &= t + (k-1)r - (n-k+1)\ell && (18)\end{aligned}$$

The construction in Figure 3 uses the following parameters.[a]

- Each share consists of t total $\mathbb{K}$-registers, where

$$t = (k+1)r\left(1 + \frac{(n-k+1)\log(\lambda)}{\lambda}\right) + 1$$

- A share is divided into r check indices and $t' = t - r$ data indices, where

$$r = (\lambda + (n-k+1)\log(\lambda))^2$$

- ℓ intuitively represents an upper bound on the amount of information which is not destroyed when an adversary generates a valid deletion certificate for a share.

$$\ell = t\frac{\log(\lambda)}{\sqrt{r}}$$

  See the proof of Lemma 3 for a more precise usage of ℓ.

- The secret will be encoded in a polynomial of degree

$$p = (k-1)t' + (n-k+1)\ell$$

[a]The parameters provided here are slightly looser than necessary, to facilitate easier inspection. We present a tighter set of parameters in Figure 4.

Figure 2: Parameters for Secret Sharing with Adaptive Certified Deletion

Therefore $t - (n-k+1)\ell > (k+1)r$. Substituting $\ell = t\frac{\log(\lambda)}{\sqrt{r}}$ yields

$$t\left(1 - (n-k+1)\frac{\log(\lambda)}{\sqrt{r}}\right) > (k+1)r \tag{19}$$

$$\begin{aligned}
t &> (k+1)r\frac{1}{1-(n-k+1)\frac{\log(\lambda)}{\sqrt{r}}} &(20)\\
&= (k+1)r\frac{\sqrt{r}}{\sqrt{r}-(n-k+1)\log(\lambda)} &(21)\\
&= (k+1)r\left(1 + \frac{(n-k+1)\log(\lambda)}{\sqrt{r}-(n-k+1)\log(\lambda)}\right) &(22)\\
&= (k+1)r\left(1 + \frac{(n-k+1)\log(\lambda)}{\lambda + (n-k+1)\log(\lambda) - (n-k+1)\log(\lambda)}\right) &(23)\\
&= (k+1)r\left(1 + \frac{(n-k+1)\log(\lambda)}{\lambda}\right) &(24)
\end{aligned}$$

Parameters: Let $\mathbb{F}_2$ be the binary field and let $\mathbb{K}$ be the field with $2^{\lceil \log_2(nt+1) \rceil}$ elements. See Figure 2 for descriptions and settings of the parameters t, t', r, ℓ, and p.

$\underline{\mathsf{Split}_{(k,n)}(1^\lambda, s)}$

- Sample a random polynomial f with coefficients in $\mathbb{K}$ and degree p such that $f(0) = s$.
- For each $i \in [n]$:
  1. Sample a random set of indices $J_i \subset [t]$ of size $t' = t - r$.
  2. For each $j \in J_i$, set $|\psi_{i,j}\rangle = |f(it + j)\rangle$. These are the t' data positions.
  3. For each $j \in [t] \backslash J_i$, sample a uniform element $y_{i,j} \leftarrow \mathbb{K}$ and set $|\psi_{i,j}\rangle = H^{\otimes \lceil \log_2(n+1) \rceil} |y_{i,j}\rangle$. These are the r check positions.
  4. Initialize register $\mathcal{S}_i$ to $\bigotimes_{j=1}^{t} |\psi_{i,j}\rangle$.
- Set $\mathsf{vk} = \{J_i, \{y_{i,j}\}_{j \in [t] \backslash J_i}\}_{i \in [n]}$.

$\underline{\mathsf{Reconstruct}_{(k,n)}(1^\lambda, \{\mathcal{S}_i\}_{i \in P})}$

- If $|P| < k$, output $\bot$. Otherwise, set P' to be any k shares in P.
- For each $i \in P'$, measure $\mathcal{S}_i$ in the computational basis to obtain $y_i = (y_{i,1}, \ldots, y_{i,t}) \in \mathbb{K}^t$.
- Compute $f \leftarrow \mathsf{Correct}_{\mathbb{K},p}(\{(it + j, y_{i,j})\}_{i \in P', j \in [t]})$, as defined in Section 3.3.
- Output $f(0)$.

$\underline{\mathsf{Delete}_{(k,n)}(1^\lambda, \mathcal{S}_i)}$

- Parse $\mathcal{S}_i$ as a sequence of $t\lceil \log_2(n+1) \rceil$ single qubit registers, measure each qubit register in Hadamard basis and output the result cert.

$\underline{\mathsf{Verify}_{(k,n)}(1^\lambda, \mathsf{vk}, i, \mathsf{cert})}$

- Parse $\mathsf{vk} = \{J_i, \{y_{i,j}\}_{j \in [t] \backslash J_i}\}_{i \in [n]}$, and parse $\mathsf{cert} \in \mathbb{K}^t$ as a sequence of t elements of $\mathbb{K}$. Output $\top$ if $\mathsf{cert}_j = y_{i,j}$ for every $j \in J_i$, and $\bot$ otherwise.

Figure 3: Construction for Secret Sharing with Adaptive Certified Deletion

Note that Equation (20) requires that $\left(1 - (n - k + 1)t\frac{\log(\lambda)}{\sqrt{r}}\right) > 0$. Since the number of check positions is $r = (\lambda + (n - k + 1)\log(\lambda))^2$, we have

$$1 - (n - k + 1)\frac{\log(\lambda)}{\lambda + (n - k + 1)\log(\lambda)} > 1 - \frac{(n - k + 1)\log(\lambda)}{(n - k + 1)\log(\lambda)} = 0 \qquad (25)$$

Finally, observe that the choice of parameters in the construction satisfies these constraints. □

7.2 Proof of Security

Recall that c_a is the a'th share to be corrupted (i.e. added to C) and d_b is the b'th share to be deleted (i.e. added to D). Observe that if c_{k-1+b} is corrupted before d_b is deleted, then $C \backslash D$ has size $\geq k$ and is authorized, so $\mathsf{SS\text{-}ACD}_{(k,n)}$ would abort.

Lemma 3. *The construction in Figure 3 using parameters from Figure 2 satisfies adaptive certified deletion for threshold secret sharing.*

We begin by defining a projector which will be useful for reasoning about how many data indices were *not* destroyed when an adversary produces a valid certificate for a share i. A certificate cert_i for share i can be parsed as t elements $\mathsf{cert}_{i,1}, \dots, \mathsf{cert}_{t,i}$ of $\mathbb{K}$. Denote $\mathsf{cert}'_i = (\mathsf{cert}_{i,j})_{j \in J_i}$ to be the subtuple of elements belonging to data indices. For any certificate cert, we define the projector[6]

$$\Pi_{\mathsf{cert}} = \sum_{\mathbf{u} \in \mathbb{K}^{t'} : h_{\mathbb{K}}(\mathbf{u}) < \ell/2} H^{\otimes t' \lceil \log_2(n+1) \rceil} \left| \mathbf{u} + \mathsf{cert}' \right\rangle \left\langle \mathbf{u} + \mathsf{cert}' \right| H^{\otimes t' \lceil \log_2(n+1) \rceil}$$

Note that H is the Hadamard gate, i.e. it implements a quantum Fourier transform over the binary field $\mathbb{F}_2$, and that the Hamming weight is taken over $\mathbb{K}$.

Let Adv be any adversary which is initialized with a state $|\psi\rangle$ on register $\mathcal{R}$. For $s \in \{s_0, s_1\}$, define the following $n - k + 3$ hybrid experiments, where $\mathcal{H}_0(s)$ is the original $\mathsf{SS\text{-}ACD}(1^\lambda, |\psi\rangle, \mathsf{Adv}, s)$ experiment.

$\underline{\mathcal{H}_1(s)}$

In $\mathcal{H}_1(s)$, we sample the shares lazily using polynomial interpolation.

1. For each share i, sample the set of data indices $J_i \subset [t]$. Then for every share i and every check position $j \in [t] \backslash J_i$, sample the check position $|\psi_{i,j}\rangle$ as in $\mathcal{H}_0(s)$.

2. For each share i, divide the data indices J_i into a left set J_i^L of size ℓ and a right set J_i^R of size $t' - \ell$. For each $j \in J_i^L$, sample $f(it + j) \leftarrow \mathbb{K}$ uniformly at random.

3. Until $k - 1$ shares are corrupted, i.e. $|C| = k - 1$, run $\mathsf{Adv}(\{S_j\}_{j \in C}, \mathcal{R})$ as in SS-ACD, with the following exception. Whenever Adv corrupts a new share by outputting $(c_a, \mathcal{R})$, finish preparing share c_a by sampling $f(c_a t + j) \leftarrow \mathbb{K}$ uniformly at random for every $j \in J_{c_i}^R$.

   At the end of this step, exactly $p = (k-1)t' + (n-k+1)\ell$ points of f have been determined, in addition to $f(0) = s$. This uniquely determines f.

4. Continue to run $\mathsf{Adv}(\{S_j\}_{j \in C}, \mathcal{R})$ as in SS-ACD, with the following exception. Whenever Adv corrupts a new share by outputting $(c_{k-1+b}, \mathcal{R})$, finish preparing c_{k-1+b} by interpolating the points in $J_{c_{k-1+b}}^R$ using share d_b and any other set of $p - t'$ points that have already been determined on f. Specifically, let

$$\mathsf{Int}_{k-1+b} \subset \{0\} \cup \bigcup_{m \in C} \{mt + j : j \in J_m\} \cup \bigcup_{m \notin C} \{mt + j : j \in J_m^L\}$$

[6]This projector defines the "deletion predicate" mentioned in the technical overview (Section 2.2).

be any set of $p+1$ indices to be used in the interpolation, such that

$$\{d_b t + j : j \in J_{d_b}\} \subset \mathsf{Int}_{k-1+b}$$

For each $j \in J^R_{c_{k-1+b}}$, compute

$$f(c_{k-1+b}t + j) \leftarrow \mathsf{Interpolate}_p\left(c_{k-1+b}t + j, \{(m, f(m)) : m \in \mathsf{Int}_{k-1+b}\}\right)$$

See Section 3.3 for the definition of $\mathsf{Interpolate}$.

Note that if SS-ACD does not abort in a round, $|C \backslash D| \leq k-1$. In the round where Adv corrupts c_{k-1+i}, $|C| = k-1+i$, so d_i has already been determined.

$\underline{\mathcal{H}_2(s)}$

In $\mathcal{H}_2(s)$, we purify the share sampling using a register $\mathcal{C} = (\mathcal{C}_1, \ldots, \mathcal{C}_n)$ which is held by the challenger. The challenger will maintain a copy of share i in register $\mathcal{C}_i = (\mathcal{C}_{i,1}, \ldots, \mathcal{C}_{i,t})$. Both $\mathcal{S}$ and $\mathcal{C}$ are initialized to $|0\rangle$ at the beginning of the experiment.

1. For each share i, sample the set of data indices $J_i \subset [t]$. Then for every share i and every check position $j \in [t] \backslash J_i$, prepare the state

$$\propto \sum_{y \in \mathbb{K}} |y\rangle^{\mathcal{S}_{i,j}} \otimes |y\rangle^{\mathcal{C}_{i,j}}$$

   Measure $\mathcal{C}_{i,j}$ in the Hadamard basis to obtain $y_{i,j}$ for the verification key.

2. Divide each J_i into J_i^L and J_i^R as in $\mathcal{H}_1(s)$. For each $j \in J_i^L$, prepare the state

$$\propto \sum_{y \in \mathbb{K}} |y\rangle^{\mathcal{S}_{i,j}} \otimes |y\rangle^{\mathcal{C}_{i,j}}$$

3. Until $k-1$ shares are corrupted, i.e. $|C| = k-1$, run $\mathsf{Adv}(\{S_j\}_{j \in C}, \mathcal{R})$ as in SS-ACD, with the following exception. Whenever Adv corrupts a new share by outputting $(c_a, \mathcal{R})$, for every $j \in J^R_{c_a}$ prepare the state

$$\propto \sum_{y \in \mathbb{K}} |y\rangle^{\mathcal{S}_{c_a,j}} \otimes |y\rangle^{\mathcal{C}_{c_a,j}}$$

4. Continue to run $\mathsf{Adv}(\{S_j\}_{j \in C}, \mathcal{R})$ as in SS-ACD, with the following exception whenever Adv corrupts a new share by outputting $(c_{k-1+b}, \mathcal{R})$. Let Int_{k-1+b} be the set of indices to be used in interpolation for share c_{k-1+b}, as in $\mathcal{H}_1(s)$. For each $j \in J_{c_{k-1+b}}$, compute

$$\mathcal{C}_{c_{k-1+b},j} \leftarrow \mathsf{Interpolate}_p\left(c_{k-1+b}t + j, (mt + j, \mathcal{S}_{m,j})_{mt+j \in \mathsf{Int}_{k-1+b}}\right)$$

   Finally, copy $\mathcal{C}_{c_{k-1+b},j}$ into $\mathcal{S}_{c_{k-1+b},j}$ in the computational basis, i.e. perform a controlled NOT operation with source register $\mathcal{C}_{c_{k-1+b},j}$ and target register $\mathcal{S}_{c_{k-1+b},j}$.

We emphasize that the timing of initializing each $\mathcal{S}_{i,j}$ is the same as in $\mathcal{H}_1(s)$. Note that since $\mathcal{H}_2(s)$ outputs either $\bot$ or Adv's view, register $\mathcal{C}$ never appears in the output of the experiment.

$\mathcal{H}_{2+i}(s)$ for $i \in [n-k+1]$

The only difference between $\mathcal{H}_{2+i}$ and $\mathcal{H}_{3+i}$ is that when the i'th share d_i is deleted in $\mathcal{H}_{3+i}$ (i.e. D reaches size i), the challenger performs a "deletion predicate" measurement on register $\mathcal{C}_{d_i}$. Specifically, let cert_{d_i} be the certificate output by Adv for share d_i. Immediately after verifying cert_{d_i} and adding d_i to D, the challenger measures the data positions in register $\mathcal{C}_{d_i}$ (i.e. register $(\mathcal{C}_{d_i,j})_{j \in J_{d_i}}$) with respect to the binary projective measurement $\{\Pi_{\mathsf{cert}_{k+i}}, I - \Pi_{\mathsf{cert}_{k+i}}\}$. If the measurement result is "reject" (i.e. $I - \Pi_{\mathsf{cert}_{k+i}}$), immediately output $\bot$ in the experiment. The difference between $\mathcal{H}_2$ and $\mathcal{H}_3$ is the same, for $i = 1$.

In addition to hybrids $\mathcal{H}_0$ through $\mathcal{H}_{3+n-k}$, we define a set of simulated experiments. Each Sim_i will be useful for reasoning about hybrid $\mathcal{H}_{2+i}$. Sim_i is similar to $\mathcal{H}_{2+i}$ except that all of the shares are randomized, whereas in $\mathcal{H}_{2+i}$, shares corrupted after c_{k-1+i} are interpolated.

Sim_i for $i \in [n-k+1]$

Run the $\mathsf{SS\text{-}ACD}(1^\lambda, |\psi\rangle, \mathsf{Adv}, s)$ experiment, with the following exceptions.

- Do *not* initialize $(\mathcal{S}_1, \ldots, \mathcal{S}_n, \mathsf{vk}) \leftarrow \mathsf{Split}_{\mathbb{S}}(1^\lambda, s)$ in step 1.
- Whenever Adv corrupts a new share by outputting $(c_a, \mathcal{R})$, prepare the state
$$\propto \sum_{\mathbf{y} \in \mathbb{K}^t} |\mathbf{y}\rangle^{\mathcal{S}_{c_a}} \otimes |\mathbf{y}\rangle^{\mathcal{C}_{c_a}}$$
Then, sample the set of data indices $J_{c_a} \subset [t]$ of size t' and for each check index $j \in [t] \backslash J_{c_a}$ measure $\mathcal{C}_{c_a,j}$ in the Hadamard basis to obtain $y_{a,j}$ for the verification key.
- For the first i deletions d_b where $b \leq i$, immediately after the challenger verifies cert_{d_b} and adds d_b to D, it measures the data positions in register $\mathcal{C}_{d_b}$ with respect to the binary projective measurement $\{\Pi_{\mathsf{cert}_{d_b}}, I - \Pi_{\mathsf{cert}_{d_b}}\}$. If the measurement result is "reject", immediately output $\bot$ in the experiment.

Claim 7. *For every secret s,*
$$\mathsf{TD}[\mathcal{H}_0(s), \mathcal{H}_2(s)] = 0$$

Proof. It is sufficient to show that $\mathsf{TD}[\mathcal{H}_0(s), \mathcal{H}_1(s)] = 0$ and $\mathsf{TD}[\mathcal{H}_1(s), \mathcal{H}_2(s)] = 0$. The former is true by the correctness of polynomial interpolation (see Section 3.3). To see the latter, observe that steps 1, 2, and 3 in $\mathcal{H}_2(s)$ are equivalent to sampling a uniformly random state (in *any* basis) in register $\mathcal{S}_{i,j}$ by preparing a uniform superposition over the basis elements in $\mathcal{S}_{i,j}$, then performing a delayed measurement from $\mathcal{S}_{i,j}$ to $\mathcal{C}_{i,j}$ in that basis. Observe that steps 1, 2, and 3 in $\mathcal{H}_1(s)$ also sample uniformly random states in $\mathcal{S}_{i,j}$. Now consider step 4. In $\mathcal{H}_2(s)$, step 4 performs a (classical) polynomial interpolation using copies of points $(it + j, f(it + j))$ that are obtained by measuring $\mathcal{S}_{i,j}$. This is equivalent to directly interpolating using $\mathcal{S}_{i,j}$ if $\mathcal{S}_{i,j}$ contained a computational basis state, which is the case in $\mathcal{H}_1(s)$. $\square$

We show that $\mathcal{H}_2$ has negligible trace distance from $\mathcal{H}_{3+n-k}$ in Claim 9. To prove Claim 9, we will need an additional fact which we show in Claim 8. Claim 8 will also show that the final hybrid $\mathcal{H}_{3+n-k}$ has zero trace distance from Sim_{n-k+1}, which is independent of the secret s.

Let $\mathcal{H}_i[c_a](s)$ denote the truncated game where $\mathcal{H}_i(s)$ is run until the end of the round where the a'th corruption occurs, i.e. when $|C|$ reaches a. At this point, $\mathcal{H}_i[c_a](s)$ outputs the adversary's register $\mathcal{R}$ and the set of corrupted registers $\{\mathcal{S}_j\}_{j\in C}$, unless the game has ended earlier (e.g. from an abort).[7] Let $\mathcal{H}_i[d_b](s)$ similarly represent the truncated game where $\mathcal{H}_i(s)$ is run until the end of the round where the b'th deletion occurs, i.e. when $|D|$ reaches b. Define $\mathsf{Sim}_i[c_a]$ and $\mathsf{Sim}_i[d_b]$ similarly.

Observe that after the n'th corruption in any hybrid experiment, the rest of the challenger's actions in the experiment is independent of the secret s. Therefore for every hybrid $\mathcal{H}_i$ and every pair of secrets (s_0, s_1),

$$\mathsf{TD}[\mathcal{H}_i[c_n](s_0), \mathcal{H}_i[c_n](s_1)] = \mathsf{TD}[\mathcal{H}_i(s_0), \mathcal{H}_i(s_1)]$$

Claim 8. *For every $i \in [0, n-k+1]$ and every secret s,*

$$\mathsf{TD}[\mathcal{H}_{2+i}[c_{k-1+i}](s), \mathsf{Sim}_i[c_{k-1+i}]] = 0$$

Combining this claim with the previous observation about the relation of a truncated experiment to its full version, it is clear that

$$\begin{aligned}\mathsf{TD}[\mathcal{H}_{3+n-k}(s_0), \mathcal{H}_{3+n-k}(s_1)] &= \mathsf{TD}[\mathcal{H}_{3+n-k}[c_n](s_0), \mathcal{H}_{3+n-k}[c_n](s_1)]\\ &= \mathsf{TD}[\mathsf{Sim}_{n-k+1}[c_n], \mathsf{Sim}_{n-k+1}[c_n]]\\ &= 0\end{aligned}$$

By Claim 9, we have

$$\mathsf{TD}[\mathcal{H}_2(s_0), \mathcal{H}_2(s_1)] \le \mathsf{TD}[\mathcal{H}_{3+n-k}(s_0), \mathcal{H}_{3+n-k}(s_1)] + \mathsf{negl}(\lambda)$$

Therefore, combining claims Claim 7, Claim 8, and Claim 9, we have

$$\begin{aligned}\mathsf{TD}[\mathcal{H}_0(s_0), \mathcal{H}_0(s_1)] &\le \mathsf{TD}[\mathcal{H}_{3+n-k}(s_0), \mathcal{H}_{3+n-k}(s_1)] + \mathsf{negl}(\lambda)\\ &\le 0 + \mathsf{negl}(\lambda)\end{aligned}$$

which completes the proof. All that remains is to prove Claim 8, and Claim 9.

Proof of Claim 8. We proceed via induction. This is clearly true for $i = 0$, since the first $k-1$ shares to be corrupted are prepared as maximally mixed states in both $\mathcal{H}_2(s)$ and Sim.

Before addressing the case of $i > 0$, we define some notation for our specific application of interpolation. When preparing a share c_{k-1+i} after it is corrupted, the challenger interpolates evaluations of f into a register

$$\mathcal{C}^R_{c_{k-1+i}} := (\mathcal{C}_{c_{k-1+i},j})_{j\in J^R_{c_{k-1+i}}}$$

$\mathcal{C}^R_{c_{k-1+i}}$ consists of the right data positions of share c_{k-1+i} and contains $t' - \ell$ $\mathbb{K}$-qudits. To do the interpolation, the challenger uses evaluations of f contained in registers

$$\mathcal{C}'_{d_i} := (\mathcal{C}_{d_i,j})_{j\in J_{d_i}}$$

[7]The truncated version of the game outputs both the set of corrupted registers and $\mathcal{R}$, while the full version only outputs $\mathcal{R}$. In the full version, the adversary can move whatever information it wants into $\mathcal{R}$. However, the truncated game ends early, so the adversary may not have done this when the game ends. Outputting the corrupted registers directly ensures that they appear in the output in some form if the game does not abort.

and some other registers which we group as $\mathcal{I}$. $\mathcal{C}'_{d_i}$ consists of the data positions in share d_i and contains t' $\mathbb{K}$-qudits. Since polynomial interpolation is a linear operation over $\mathbb{K}$, the system immediately after c_{k-1+i} is prepared can be described as a state

$$\sum_{\substack{\mathbf{x_1}\in\mathbb{K}^{t'}\\ \mathbf{x_2}\in\mathbb{K}^{d-t'}}} \alpha_{\mathbf{x_1},\mathbf{x_2}} \left|\mathbf{x_1}\right\rangle^{\mathcal{C}'_{d_i}} \otimes \left|\mathbf{x_2}\right\rangle^{\mathcal{I}} \otimes \left|\mathbf{R_1}\mathbf{x_1}+\mathbf{R_2}\mathbf{x_2}\right\rangle^{\mathcal{C}^R_{c_{k-1+i}}} \otimes \left|\mathbf{R_1}\mathbf{x_1}+\mathbf{R_2}\mathbf{x_2}\right\rangle^{\mathcal{S}^R_{c_{k-1+i}}} \otimes \left|\phi_{\mathbf{x_1},\mathbf{x_2}}\right\rangle^{\mathcal{C}',\mathcal{S}',\mathcal{R}}$$

where $\mathbf{R_1} \in \mathbb{K}^{(t'-\ell)\times t'}$ and $\mathbf{R_2} \in \mathbb{K}^{(t'-\ell)\times(d+1-t')}$ are submatrices of the interpolation transformation, where $\mathcal{S}^R_{c_{k-1+i}}$ contains the copy of the evaluations in $\mathcal{C}^R_{c_{k-1+i}}$, where $\mathcal{C}'$ and $\mathcal{S}'$ respectively consist of the unmentioned registers of $\mathcal{C}$ and $\mathcal{S}$, and where $\mathcal{R}$ is the adversary's internal register.

Now we will show that the claim holds for $i+1$ if it holds for i. Define the following hybrid experiments.

- $\mathcal{H}_{3+i}[c_{k+i}]$: Recall that the only difference between $\mathcal{H}_{2+i}$ and $\mathcal{H}_{3+i}$ is an additional measurement made in the same round that the $(i+1)$'th share is deleted, i.e. when $|D|$ reaches $i+1$.
- $\mathcal{H}'_{3+i}[c_{k+i}]$: The only difference from $\mathcal{H}_{3+i}[c_{k+i}]$ occurs when the adversary requests to corrupt share c_{k+i}. When this occurs, the challenger prepares the right data positions of share c_{k+i} as the state

$$\propto \sum_{\mathbf{y}\in\mathbb{K}^{t'}} \left|\mathbf{y}\right\rangle^{\mathcal{C}^R_{c_{k+i}}} \otimes \left|\mathbf{y}\right\rangle^{\mathcal{S}^R_{c_{k+i}}}$$

- $\mathsf{Sim}_i[c_{k+i}]$: The only difference from $\mathcal{H}'_{3+i}[c_{k+i}]$ is that $\mathsf{Sim}_i[c_{k+i}]$ is run until c_{k+i} is corrupted, then the experiment is finished according to $\mathcal{H}'_{3+i}[c_{k+i}]$.

We first show that $\mathcal{H}'_{3+i}[c_{k+i}]$ and $\mathsf{Sim}_i[c_{k+i}]$ are close. By the inductive hypothesis,

$$\mathsf{TD}[\mathcal{H}_{2+i}[c_{k-1+i}](s), \mathsf{Sim}_{i-1}[c_{k-1+i}]] = 0$$

Note that $\mathcal{H}_{2+i}(s)$ and $\mathcal{H}'_{3+i}(s)$ are identical until the $(i+1)$'th deletion d_{i+1}. Similarly, Sim_{i-1} and Sim_i are identical until the $(i+1)$'th deletion. Therefore

$$\mathsf{TD}[\mathcal{H}'_{3+i}[d_i](s), \mathsf{Sim}_i[d_i]] = 0$$

Finally, observe that if the experiment does not abort, then d_i is deleted before c_{k+i} is corrupted (otherwise $|C\backslash D| = k$ during some round). Because of this, $\mathcal{H}'_{3+i}[c_{k+i}]$ and $\mathsf{Sim}[c_{k+i}]$ behave identically after the round where d_i is corrupted. Therefore

$$\mathsf{TD}[\mathcal{H}'_{3+i}[c_{k+i}], \mathsf{Sim}[c_{k+i}]] = 0$$

It remains to be shown that

$$\mathsf{TD}[\mathcal{H}_{3+i}[c_{k+i}], \mathcal{H}'_{3+i}[c_{k+i}]] = 0$$

The only difference between $\mathcal{H}_{3+i}[c_{k+i}]$ and $\mathcal{H}'_{3+i}[c_{k+i}]$ is at the end of the last round, where c_{k+i} is corrupted.[8] If the experiment reaches the end of this round without aborting, then d_i has already been corrupted, since otherwise at some point $|C\backslash D| = k$. Furthermore, the deletion predicate measurement on

[8]In the definition of the SS-ACD experiment, the corruption set C is updated in between adversarial access to the corrupted registers, which occur once at the beginning of each round. The truncated game outputs according to the updated C, including $\mathcal{S}_{c_{k+i}}$.

$\mathcal{C}_{d_i}$ must have accepted or else the experiment also would have aborted. It is sufficient to prove that the two experiments have 0 trace distance conditioned on not aborting.

In $\mathcal{H}'_{3+i}[c_{k+i}]$, if the experiment does not abort then its end state (after tracing out the challenger's $\mathcal{C}$ register) is

$$\sum_{\substack{\mathbf{x_1}\in\mathbb{K}^{t'}\\ \mathbf{x_2}\in\mathbb{K}^{d-t'}\\ x_3\in\mathbb{K}^{t'-\ell}}} |\alpha_{\mathbf{x_1},\mathbf{x_2}}|^2 \left|x_3\right\rangle\left\langle x_3\right|^{\mathcal{S}^R_{c_{k-1+i}}} \otimes \mathsf{Tr}^{\mathcal{C}'}\left[\left|\phi_{\mathbf{x_1},\mathbf{x_2}}\right\rangle\left\langle\phi_{\mathbf{x_1},\mathbf{x_2}}\right|^{\mathcal{C}',\mathcal{S}',\mathcal{R}}\right]$$

We now calculate the end state of $\mathcal{H}'_{3+i}[c_{k+i}]$, conditioned on it not aborting. In this case, the challenger for $\mathcal{H}'_{3+i}[c_{k+i}]$ prepares the next corrupted register $\mathcal{S}_{c_{k+i}}$ by a polynomial interpolation which uses registers $\mathcal{C}_{d_i}$ and $\mathcal{I}$. The deletion predicate measurement on $\mathcal{C}_{d_i}$ must have accepted to avoid an abort, so before performing the interpolation, the state of the system is of the form

$$\left|\gamma\right\rangle^{A,C',\mathsf{Int}_2,C_{d_{k+i}}} = \sum_{\mathbf{u}\in\mathbb{K}^{t'}:h_{\mathbb{K}}(u)<\ell/2} \alpha_{\mathbf{u}}\left|\psi_{\mathbf{u}}\right\rangle^{\mathcal{S},\mathcal{C}',\mathcal{I}} \otimes H^{\otimes t'\lceil\log_2(n+1)\rceil}\left|\mathbf{u}+\mathsf{cert}_{k+i}\right\rangle^{\mathcal{C}_{d_i}}$$

We will apply Theorem 3 with input size t' and output size $t'-\ell$ to show that share d_i contributes uniform randomness to the preparation of c_{k+i}. Observe that $\mathcal{C}_{d_i}$ contains t' $\mathbb{K}$-qudits and the interpolation target $\mathcal{C}^R_{c_{k+i}}$ contains $t'-\ell$ $\mathbb{K}$-qudits. Furthermore, $[R_1|R_2] \in \mathbb{K}^{(t'-\ell)\times(d+1)}$ is an interpolation matrix for a polynomial of degree p. By Fact 1, any $t'-\ell$ columns of $[R_1|R_2]$ are linearly independent. In particular, any $t'-\ell$ columns of R_1 are linearly independent. Finally, note that $(t'-(t'-\ell))/2 = \ell/2$. Therefore by Theorem 3, the state of the system after preparing register $\mathcal{C}^R_{c_{k+i}}$ and tracing out $\mathcal{C}_{d_i}$ when the experiment ends at the end of this round is

$$\sum_{\mathbf{x_3}\in\mathbb{K}^{t'-\ell}} \mathsf{Tr}^{\mathcal{C}_{d_i}}\left[\left|\gamma_{\mathbf{x_3}}\right\rangle\left\langle\gamma_{\mathbf{x_3}}\right|^{\mathcal{C},\mathcal{S},\mathcal{R}}\right]$$

where

$$\left|\gamma_{\mathbf{x_3}}\right\rangle = \sum_{\substack{\mathbf{x_1}\in\mathbb{K}^{t'}\\ \mathbf{x_2}\in\mathbb{K}^{d-t'}}} \alpha_{\mathbf{x_1},\mathbf{x_2}}\left|\mathbf{x_1}\right\rangle^{\mathcal{C}_{d_i}} \otimes \left|\mathbf{x_2}\right\rangle^{\mathcal{I}} \otimes \left|\mathbf{x_3}+\mathbf{R_2}\mathbf{x_2}\right\rangle^{\mathcal{C}_{c_{k+i}}} \otimes \left|\mathbf{x_3}+\mathbf{R_2}\mathbf{x_2}\right\rangle^{\mathcal{S}_{c_{k+i}}} \otimes \left|\varphi_{\mathbf{x_1},\mathbf{x_2}}\right\rangle^{\mathcal{C}',\mathcal{S}',\mathcal{R}}$$

After this round, $\mathcal{H}_{3+i}[c_{k+i}]$ ends and register $\mathcal{C}$ is traced out. This yields the state

$$\begin{aligned}
&\sum_{\mathbf{x_3}\in\mathbb{K}^{t'-\ell}} \mathsf{Tr}^{\mathcal{C}}\left[\left|\gamma_{\mathbf{x_3}}\right\rangle\left\langle\gamma_{\mathbf{x_3}}\right|^{\mathcal{C},\mathcal{S},\mathcal{R}}\right]\\
&\quad= \sum_{\substack{\mathbf{x_1}\in\mathbb{K}^{t'}\\ \mathbf{x_2}\in\mathbb{K}^{d-t'}\\ \mathbf{x_3}\in\mathbb{K}^{t'-\ell}}} |\alpha_{\mathbf{x_1},\mathbf{x_2}}|^2\left|\mathbf{x_3}+\mathbf{R_2}\mathbf{x_2}\right\rangle\left\langle\mathbf{x_3}+\mathbf{R_2}\mathbf{x_2}\right|^{\mathcal{S}^R_{c_{k+i}}} \otimes \mathsf{Tr}^{\mathcal{C}'}\left[\left|\phi_{\mathbf{x_1},\mathbf{x_2}}\right\rangle\left\langle\phi_{\mathbf{x_1},\mathbf{x_2}}\right|^{\mathcal{C}',\mathcal{S}',\mathcal{R}}\right]\\
&\quad= \sum_{\substack{\mathbf{x_1}\in\mathbb{K}^{t'}\\ \mathbf{x_2}\in\mathbb{K}^{d-t'}\\ \mathbf{x_4}\in\mathbb{K}^{t'-\ell}}} |\alpha_{\mathbf{x_1},\mathbf{x_2}}|^2\left|\mathbf{x_4}\right\rangle\left\langle\mathbf{x_4}\right|^{\mathcal{S}^R_{c_{k+i}}} \otimes \mathsf{Tr}^{\mathcal{C}'}\left[\left|\phi_{\mathbf{x_1},\mathbf{x_2}}\right\rangle\left\langle\phi_{\mathbf{x_1},\mathbf{x_2}}\right|^{\mathcal{C}',\mathcal{S}',\mathcal{R}}\right]
\end{aligned}$$

where $\mathbf{x_4} = \mathbf{x_3} + \mathbf{R_2}\mathbf{x_2}$. This state is identical to the state at the end of $\mathcal{H}'_{3+i}[c_{k+i}]$ conditioned on the experiments not aborting.

□

Claim 9. *For every $i \in [0, n-k]$ and every secret s,*

$$\mathsf{TD}[\mathcal{H}_{2+i}(s), \mathcal{H}_{3+i}(s)] = \mathsf{negl}(\lambda)$$

Proof. The only difference between $\mathcal{H}_{2+i}(s)$ and $\mathcal{H}_{3+i}(s)$ is an additional deletion predicate measurement $\Pi_{\mathsf{cert}_{d_{i+1}}}$ during the round where d_{i+1} is corrupted. Say the deletion predicate accepts with probability $1-\epsilon$. Then the Gentle Measurement Lemma (Lemma 1) implies that, conditioned on the deletion predicate accepting, the distance between $\mathcal{H}_{2+i}(s)$ and $\mathcal{H}_{3+i}(s)$ is at most $2\sqrt{\epsilon}$. We upper bound the case where the deletion predicate rejects by 1 to obtain

$$\mathsf{TD}[\mathcal{H}_{2+i}(s), \mathcal{H}_{3+i}(s)] \leq (1-\epsilon)2\sqrt{\epsilon} + \epsilon$$

Thus, it is sufficient to show that $\epsilon = \mathsf{negl}(\lambda)$, i.e. the deletion predicate accepts with high probability on $\mathcal{C}_{d_{i+1}}$ in $\mathcal{H}_{3+i}$. To show this, we consider the following hybrids, and claim that the probability that the deletion predicate accepts on $\mathcal{C}_{d_{i+1}}$ is *identical* in each of the hybrids.

- $\mathcal{H}_{3+i}$
- $\mathcal{H}_{3+i}[d_{i+1}]$: The only difference is that the game ends after the round where d_{i+1} is deleted.
- $\mathsf{Sim}_{i+1}[d_{i+1}]$: Recall that the only difference between $\mathcal{H}_{3+i}$ and Sim_{i+1} is that every share j is prepared as the maximally entangled state

$$\sum_{\mathbf{x} \in \mathbb{K}^t} |\mathbf{x}\rangle^{C_j} \otimes |\mathbf{x}\rangle^{A_j}$$

- $\mathsf{Sim}'_{i+1}[d_{i+1}]$: The same as $\mathsf{Sim}_{i+1}[d_{i+1}]$, except that after preparing the maximally entangled state for each share j, we delay choosing J_j and measuring the check indices $\mathcal{C}_{i,j}$ for $j \in [t] \backslash J_j$. These are now done immediately after Adv deletes j by outputting $(\mathsf{cert}_j, j, \mathcal{R})$, and before the challenger verifies cert_j.

Observe that $\mathcal{H}_{3+i}$ and $\mathcal{H}_{3+i}[d_{i+1}]$ are identical until the deletion predicate measurement in the round where d_{i+1} is deleted, so the probability of acceptance is identical. By Claim 8,

$$\mathsf{TD}[\mathcal{H}_{3+i}[c_{k+i}](s), \mathsf{Sim}_{i+1}[c_{k+i}]] = 0$$

Share d_{i+1} is deleted before share c_{k+i} is corrupted in both $\mathcal{H}_{3+i}(s)$ and Sim_{i+1}, unless they abort. Therefore

$$\mathsf{TD}[\mathcal{H}_{3+i}[d_{i+1}](s), \mathsf{Sim}_{i+1}[d_{i+1}]] = 0$$

and the probability of acceptance is identical in $\mathcal{H}_{3+i}[d_{i+1}](s)$ and $\mathsf{Sim}_{i+1}[d_{i+1}]$. Finally, the probability of acceptance is identical in $\mathsf{Sim}'_{i+1}[d_{i+1}]$ because the register C is disjoint from the adversary's registers.

Thus, it suffices to show that $\epsilon = \mathsf{negl}(\lambda)$ in $\mathsf{Sim}'_{i+1}[d_{i+1}]$. Since $\mathbb{K}$ forms a vector space over $\mathbb{F}_2$, the certificate verification measurement and $\Pi_{\mathsf{cert}_{d_{i+1}}}$ are diagonal in the binary Fourier basis (i.e. the Hadamard basis) for every cert. Therefore the probability that Verify accepts $\mathsf{cert}_{d_{i+1}}$ but the deletion predicate measurement *rejects* $\mathcal{C}_{d_{i+1}}$ is

$$\epsilon = \Pr_{\substack{\mathsf{cert}, \mathbf{y} \in \mathbb{K}^t \\ J \subset [t] : |J| = t'}} \left[\mathsf{cert}_{\overline{J}} = \mathbf{y}_{\overline{J}} \wedge \Delta_{\mathbb{K}}(\mathsf{cert}_J, \mathbf{y}_J) \geq \frac{\ell}{2} \right]$$

where J is the set of data indices for share d_{i+1}, where $\overline{J}$ is the set complement of J (i.e. the set of check indices for share d_{i+1}), and where $\Delta_{\mathbb{K}}(\mathsf{cert}_J, \mathbf{y}_J) = h_{\mathbb{K}}(\mathsf{cert}_J - \mathbf{y}_J)$ is the Hamming distance of cert_J from y_J. Here, the probability is taken over the adversary outputting a certificate cert for d_{k+i}, the challenger sampling a set of check indices $\overline{J}$, and the challenger measuring all of register $\mathcal{C}_{d_{i+1}}$ in the Hadamard basis to obtain $\mathbf{y} \in \mathbb{K}^t$.

This value can be upper bounded using Hoeffding's inequality, for any fixed cert and $\mathbf{y}$ with $\Delta_{\mathbb{K}}(\mathsf{cert}_J, \mathbf{y}_J) \geq \ell/2$. Note that the probability of acceptance is no greater than if the r check indices $\overline{J}$ are sampled *with* replacement. In this case, the expected number of check indices which do *not* match is

$$\geq \frac{\ell r}{2t} = \frac{t\log(\lambda)}{\lambda + (n-k+1)\log(\lambda)} \frac{(\lambda + (n-k+1)\log(\lambda))^2}{2t} \tag{26}$$

$$= \frac{\log(\lambda)}{2}(\lambda + (n-k+1)\log(\lambda)) \tag{27}$$

Therefore Hoeffding's inequality (Claim 2) implies that

$$\epsilon \leq 2\exp\left(\frac{-2\left(\frac{\log(\lambda)}{2}(\lambda + (n-k+1)\log(\lambda))\right)^2}{(\lambda + (n-k+1)\log(\lambda))^2}\right) \tag{28}$$

$$= 2\exp\left(-\frac{\log^2(\lambda)}{2}\right) \tag{29}$$

$$= \mathsf{negl}(\lambda) \tag{30}$$

□

8 Acknowledgements

We thank Orestis Chardouvelis and Dakshita Khurana for collaboration and insightful discussions during the early stages of this research.

References

[ABF+19] Benny Applebaum, Amos Beimel, Oriol Farràs, Oded Nir, and Naty Peter. Secret-sharing schemes for general and uniform access structures. In Yuval Ishai and Vincent Rijmen, editors, *EUROCRYPT 2019, Part III*, volume 11478 of *LNCS*, pages 441–471. Springer, Heidelberg, May 2019.

[ABKK23] Amit Agarwal, James Bartusek, Dakshita Khurana, and Nishant Kumar. A new framework for quantum oblivious transfer. In Carmit Hazay and Martijn Stam, editors, *EUROCRYPT 2023, Part I*, volume 14004 of *LNCS*, pages 363–394. Springer, Heidelberg, April 2023.

[Bei11] Amos Beimel. Secret-sharing schemes: A survey. In Yeow Meng Chee, Zhenbo Guo, San Ling, Fengjing Shao, Yuansheng Tang, Huaxiong Wang, and Chaoping Xing, editors, *Coding and Cryptology*, pages 11–46, Berlin, Heidelberg, 2011. Springer Berlin Heidelberg.

[BF10] Niek J. Bouman and Serge Fehr. Sampling in a quantum population, and applications. In Tal Rabin, editor, *CRYPTO 2010*, volume 6223 of *LNCS*, pages 724–741. Springer, Heidelberg, August 2010.

[BGK$^+$24] James Bartusek, Vipul Goyal, Dakshita Khurana, Giulio Malavolta, Justin Raizes, and Bhaskar Roberts. Software with certified deletion. In *Eurocrypt 2024 (to appear)*, 2024.

[BI20] Anne Broadbent and Rabib Islam. Quantum encryption with certified deletion. In Rafael Pass and Krzysztof Pietrzak, editors, *TCC 2020, Part III*, volume 12552 of *LNCS*, pages 92–122. Springer, Heidelberg, November 2020.

[BK23] James Bartusek and Dakshita Khurana. Cryptography with certified deletion. In Helena Handschuh and Anna Lysyanskaya, editors, *CRYPTO 2023, Part V*, volume 14085 of *LNCS*, pages 192–223. Springer, Heidelberg, August 2023.

[BKM$^+$23] James Bartusek, Dakshita Khurana, Giulio Malavolta, Alexander Poremba, and Michael Walter. Weakening assumptions for publicly-verifiable deletion. In Guy Rothblum and Hoeteck Wee, editors, *Theory of Cryptography*, pages 183–197, Cham, 2023. Springer Nature Switzerland.

[BKP23] James Bartusek, Dakshita Khurana, and Alexander Poremba. Publicly-verifiable deletion via target-collapsing functions. In Helena Handschuh and Anna Lysyanskaya, editors, *CRYPTO 2023, Part V*, volume 14085 of *LNCS*, pages 99–128. Springer, Heidelberg, August 2023.

[BL90] Josh Cohen Benaloh and Jerry Leichter. Generalized secret sharing and monotone functions. In Shafi Goldwasser, editor, *CRYPTO'88*, volume 403 of *LNCS*, pages 27–35. Springer, Heidelberg, August 1990.

[Bla79] G. R. Blakley. Safeguarding cryptographic keys. *1979 International Workshop on Managing Requirements Knowledge (MARK)*, pages 313–318, 1979.

[DF90] Yvo Desmedt and Yair Frankel. Threshold cryptosystems. In Gilles Brassard, editor, *CRYPTO'89*, volume 435 of *LNCS*, pages 307–315. Springer, Heidelberg, August 1990.

[DFL$^+$09] Ivan Damgård, Serge Fehr, Carolin Lunemann, Louis Salvail, and Christian Schaffner. Improving the security of quantum protocols via commit-and-open. In Shai Halevi, editor, *CRYPTO 2009*, volume 5677 of *LNCS*, pages 408–427. Springer, Heidelberg, August 2009.

[Gao03] Shuhong Gao. *A New Algorithm for Decoding Reed-Solomon Codes*, pages 55–68. Springer US, Boston, MA, 2003.

[HKM$^+$24] Taiga Hiroka, Fuyuki Kitagawa, Tomoyuki Morimae, Ryo Nishimaki, Tapas Pal, and Takashi Yamakawa. Certified everlasting secure collusion-resistant functional encryption, and more. In *Eurocrypt 2024 (to appear)*, 2024.

[HMNY21] Taiga Hiroka, Tomoyuki Morimae, Ryo Nishimaki, and Takashi Yamakawa. Quantum encryption with certified deletion, revisited: Public key, attribute-based, and classical communication. In Mehdi Tibouchi and Huaxiong Wang, editors, *ASIACRYPT 2021, Part I*, volume 13090 of *LNCS*, pages 606–636. Springer, Heidelberg, December 2021.

[HMNY22] Taiga Hiroka, Tomoyuki Morimae, Ryo Nishimaki, and Takashi Yamakawa. Certified everlasting zero-knowledge proof for QMA. In Yevgeniy Dodis and Thomas Shrimpton, editors, *CRYPTO 2022, Part I*, volume 13507 of *LNCS*, pages 239–268. Springer, Heidelberg, August 2022.

[Hoe94] Wassily Hoeffding. *Probability Inequalities for sums of Bounded Random Variables*, pages 409–426. Springer New York, New York, NY, 1994.

[ISN87] M. Ito, A. Saito, and Takao Nishizeki. Secret sharing schemes realizing general access structure. In *Proc. IEEE Global Telecommunication Conf. (Globecom'87)*, pages 99–102, 1987.

[KNY23] Fuyuki Kitagawa, Ryo Nishimaki, and Takashi Yamakawa. Publicly verifiable deletion from minimal assumptions. In Guy N. Rothblum and Hoeteck Wee, editors, *Theory of Cryptography - 21st International Conference, TCC 2023, Taipei, Taiwan, November 29 - December 2, 2023, Proceedings, Part IV*, volume 14372 of *Lecture Notes in Computer Science*, pages 228–245. Springer, 2023.

[LM01] J.L. Lagrange and T.J. McCormack. *Lectures on Elementary Mathematics*. Open Court Publishing Company, 1901.

[LV18] Tianren Liu and Vinod Vaikuntanathan. Breaking the circuit-size barrier in secret sharing. In *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing*, STOC 2018, page 699–708, New York, NY, USA, 2018. Association for Computing Machinery.

[Por23] Alexander Poremba. Quantum Proofs of Deletion for Learning with Errors. In Yael Tauman Kalai, editor, *14th Innovations in Theoretical Computer Science Conference (ITCS 2023)*, volume 251 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 90:1–90:14, Dagstuhl, Germany, 2023. Schloss Dagstuhl – Leibniz-Zentrum für Informatik.

[RK05] Renato Renner and Robert König. Universally composable privacy amplification against quantum adversaries. In Joe Kilian, editor, *TCC 2005*, volume 3378 of *LNCS*, pages 407–425. Springer, Heidelberg, February 2005.

[RS60] I. S. Reed and G. Solomon. Polynomial codes over certain finite fields. *Journal of the Society for Industrial and Applied Mathematics*, 8(2):300–304, 1960.

[Sha79] Adi Shamir. How to share a secret. *Communications of the Association for Computing Machinery*, 22(11):612–613, November 1979.

[WB86] Lloyd R Welch and Elwyn R Berlekamp. Error correction for algebraic block codes, December 30 1986. US Patent 4,633,470.

[Win99] Andreas J. Winter. Coding theorem and strong converse for quantum channels. *IEEE Trans. Inf. Theory*, 45(7):2481–2485, 1999.

A Tighter Parameters for the Threshold Construction

In this section, we give alternate parameter settings for the construction in Figure 3 that result in slightly smaller share sizes. The parameters are described in Figure 4. The main difference from Figure 2 is that r is slightly smaller, which also impacts t.

The construction in Figure 3 uses the following parameters.

- Each share consists of t total $\mathbb{K}$-registers, where

$$t = (k+1)r\left(1 + \frac{(n-k+1)\log(\lambda)}{\sqrt{r} - (n-k+1)\log(\lambda)}\right) + 1$$

- A share is divided into r check indices and $t' = t - r$ data indices, where

$$r = \lambda + (n-k+1)^2 \log^2(\lambda)$$

- ℓ intuitively represents an upper bound on the amount of information which is not destroyed when an adversary generates a valid deletion certificate for a share.

$$\ell = t\frac{\log(\lambda)}{\sqrt{r}}$$

  See the proof of Lemma 3 for a more precise usage of ℓ.

- The secret will be encoded in a polynomial of degree

$$d = (k-1)t' + (n-k+1)\ell$$

Figure 4: Alternate Parameters for Secret Sharing with Adaptive Certified Deletion

Lemma 4. *The construction in Figure 3 has reconstruction correctness with the parameters in Figure 4.*

Proof. The set $\{(it+j, y_{i,j})\}_{i\in P', j\in[t]}$ contains kt pairs which were obtained by measuring k shares. As mentioned in Section 3.3, if all but $e < (kt-d)/2$ of these pairs $(it+j, y_{i,j})$ satisfy $y_{i,j} = f(it+j)$, then $\mathsf{Correct}_{\mathbb{K},d}$ recovers the original polynomial f, where $f(0) = s$. The only points which do not satisfy this are the check positions, of which there are r per share, for a total of kr. Therefore for correctness, we require that

$$\begin{aligned} 2kr &< kt - d & (31)\\ &= kt - (k-1)(t-r) - (n-k+1)\ell & (32)\\ &= t + (k-1)r - (n-k+1)\ell & (33) \end{aligned}$$

Therefore $t - (n-k+1)\ell > (k+1)r$. Substituting $\ell = t\frac{\log(\lambda)}{\sqrt{r}}$ yields

$$t\left(1 - (n-k+1)\frac{\log(\lambda)}{\sqrt{r}}\right) > (k+1)r \quad (34)$$

$$t > (k+1)r\frac{1}{1 - (n-k+1)\frac{\log(\lambda)}{\sqrt{r}}} \quad (35)$$

$$= (k+1)r\frac{\sqrt{r}}{\sqrt{r}-(n-k+1)\log(\lambda)} \tag{36}$$

$$= (k+1)r\left(1+\frac{(n-k+1)\log(\lambda)}{\sqrt{r}-(n-k+1)\log(\lambda)}\right) \tag{37}$$

Note that Equation (35) requires that $\left(1-(n-k+1)t\frac{\log(\lambda)}{\sqrt{r}}\right) > 0$. Since the number of check positions is $r = \lambda + (n-k+1)^2\log^2(\lambda)$, we have

$$1-(n-k+1)\frac{\log(\lambda)}{\sqrt{\lambda+(n-k+1)^2\log^2(\lambda)}} > 1-\frac{(n-k+1)\log(\lambda)}{(n-k+1)\log(\lambda)} = 0 \tag{38}$$

Finally, observe that the choice of parameters in the construction satisfies these constraints. □

Lemma 5. *The construction in Figure 3 has adaptive certified deletion security with the parameters in Figure 4.*

Proof Sketch. The proof is almost the same as that of Lemma 3, except for the application of Hoeffding's inequality in Claim 9. The expected number of check indices which do not match becomes

$$\begin{aligned}
\geq \frac{\ell r}{2t} &= \frac{t\log(\lambda)}{\sqrt{\lambda+(n-k+1)^2\log^2(\lambda)}}\frac{\lambda+(n-k+1)^2\log^2(\lambda)}{2t}\\
&= \frac{\log(\lambda)}{2}\sqrt{\lambda+(n-k+1)^2\log^2(\lambda)}
\end{aligned}$$

Then Hoeffding's inequality implies

$$\begin{aligned}
\epsilon &\leq 2\exp\left(\frac{-2\left(\frac{\log(\lambda)}{2}\sqrt{\lambda+(n-k+1)^2\log^2(\lambda)}\right)^2}{\lambda+(n-k+1)^2\log^2(\lambda)}\right)\\
&= 2\exp\left(-\frac{\log^2(\lambda)}{2}\right)\\
&= \mathsf{negl}(\lambda)
\end{aligned}$$

□