Traceable Secret Sharing Based on the Chinese Remainder Theorem

Charlotte Hoffmann

Institute of Science and Technology Austria
charlotte.hoffmann@ista.ac.at

Abstract

Traceable threshold secret sharing schemes, introduced by Goyal, Song and Srinivasan (CRYPTO'21), allow to provably trace leaked shares to the parties that leaked them. The authors give the first definition and construction of traceable secret sharing schemes. However, the size of the shares in their construction are quadratic in the size of the secret. Boneh, Partap and Rotem (CRYPTO'24) recently proposed a new definition of traceable secret sharing and the first practical constructions. In their definition, one considers a reconstruction box R that contains f leaked shares and, on input $t-f$ additional shares, outputs the secret s. A scheme is traceable if one can find out the leaked shares inside the box R by only getting black-box access to R and a private tracing key. Boneh, Partap and Rotem give constructions from Shamir's secret sharing and Blakley's secret sharing. The constructions are efficient as the size of the secret shares is only twice the size of the secret.

In this work we present the first traceable secret sharing scheme based on the Chinese remainder theorem. This was stated as an open problem by Boneh, Partap and Rotem, as it gives rise to traceable secret sharing with weighted threshold access structures. The scheme is based on Mignotte's secret sharing and increases the size of the shares of the standard Mignotte secret sharing scheme only by a factor of 2. We also introduce a new definition of semi-public secret sharing that does not require a private tracing key and give a construction based on the Chinese Remainder Theorem.

1 Introduction

Threshold secret sharing, introduced by Shamir [30] and Blakley [3], allows a dealer to split a secret s into n shares $\mathsf{sh}_1, \ldots, \mathsf{sh}_n$ such that s can be reconstructed from any t shares, while any $t-1$ shares reveal basically no information about s.

Traceable secret sharing. Goyal, Song and Srinivasan [19] recently introduced the notion of *traceable secret sharing* which allows one to trace back leaked shares to the parties that leaked them. They consider the following scenario: Alice has shared a secret s, e.g., a secret key, among n servers with a threshold secret sharing scheme. Suppose f servers collude and sell their shares, possibly in an obfuscated way such that they can not be trivially traced back to the owners. In a traceable secret sharing scheme it should be possible to trace at least one of the corrupted servers given the leaked information. Further, the tracer should be able to produce a proof that implicates the corrupted servers.

Goyal, Song and Srinivasan [19] gave the first definition and construction of a traceable secret sharing scheme. Their construction, however, is not practical as the size of the secret shares is quadratic in the size of the secret.

Boneh, Partap and Rotem [6] propose a new definition of traceable secret sharing, which allows them to give the first practical constructions from Shamir's secret sharing and Blakley's secret sharing. In their definition a tracer is given black-box access to a reconstruction box R that has $f < t$ shares hardcoded in it. On input $t-f$ additional shares, R outputs the secret that can be reconstructed from the t shares it now holds. In a traceable secret sharing scheme the dealer not only shares the secret, but also constructs a tracing key and a verification key. The scheme is called *traceable* if, given the tracing key, the tracer can find all f parties that own one of the shares hardcoded in R and produce a proof that implicates these parties. The proof should be verifiable given the verification key. The scheme is called *non-imputable* if the tracer cannot falsely accuse a party by forging a proof of their corruptness. The authors present two schemes that satisfy traceability and non-imputability – one based on Shamir's secret sharing and one based on Blakley's secret sharing scheme. The schemes are practical in the sense that the share size is only twice as large as the size of the secret.

Secret sharing based on the Chinese remainder theorem. in Shamir's and Blakley's secret sharing schemes the secret is randomly embedded into a higher dimensional space and encoded via polynomials or hyperplanes. Different examples of classic secret sharing schemes are based on the Chinese remainder theorem (CRT). The main idea underlying these type of schemes is the following: The secret s can be seen as a group element of $\mathbb{Z}_N$ and the shares are of the form $\mathsf{sh}_i = (s_i, p_i)$, where p_i is a divisor of N and $s_i := s \mod p_i$. Given shares $\mathsf{sh}_{i_1}, \ldots, \mathsf{sh}_{i_t}$ with $p_{i_1} \cdot \ldots \cdot p_{i_t} = N$ one can reconstruct the secret using the Chinese remainder theorem. Two classic examples of such schemes are Mignotte's secret sharing scheme [25] and the Asmuth-Bloom secret sharing scheme [1]. In Mignotte's scheme the shares are smaller than the secret and in the Asmuth-Bloom scheme shares are larger than the secret. While the Asmuth-Bloom does not satisfy *perfect* privacy, i.e., all secrets are equally likely even given $t-1$ shares, it does hold that given $t-1$ shares, all elements in the secret space could be the shared secret. In Mignotte's scheme $t-1$ shares can already rule out some of the secrets in the secret space. However, the parameters of the scheme can be set such that given $t-1$ shares, the number of possible secrets is still large enough. This is sufficient for the application described above, where the secret is a random secret key.

Secret sharing with more general access structures. While most CRT based secret sharing schemes do not satisfy perfect privacy, they have a very useful property: they can be extended to allow for more general access structures, for example *weighted* threshold access structures, where each share has a weight associated with it and the secret can be reconstructed whenever the sum of the weights of the shares exceeds the threshold [20]. Shamir's and Blakley's scheme only have this property to a certain degree: One can give certain parties more shares than others. However, they can not account for more complicated access structures like the following example from [2]: The secret is shared between parties $1, 2, 3$ and 4 and the secret should only be reconstructable if either the pair $(1, 2)$ is involved or the pair $(3, 4)$ is involved. Both Mignotte's secret sharing scheme and the Asmuth-Bloom secret sharing scheme can support a variety of access structures [20]. To realize the access structure above, for example, one could choose integers $p_1 < p_2 < p_3 < p_4$ of which only the pairs (p_1, p_2) and (p_3, p_4) are coprime, choose the secret $p_4 < S < p_1 \cdot p_2$ and then give share $\mathsf{sh}_i = (S \mod p_i, p_i)$ to party i for all $i \in \{1, 2, 3, 4\}$.

Traceability of CRT based secret sharing schemes. All known traceable secret sharing schemes are based on either Shamir's or Blakley's schemes, which encode the secret via polynomials or hyperplanes. However, due to the advantages discussed above, for some applications CRT based schemes are preferable. Boneh, Partap and Rotem [6] therefore pose the following open question:

Is it possible to build traceable secret sharing schemes based on the Chinese Remainder Theorem?

If this is possible, the next question that arises is if we can hope for the scheme to be practical like the schemes in [6].

1.1 Our Contribution

In this work we answer the above questions in the affirmative. We present the first traceable secret sharing schemes based on the Chinese remainder theorem. Our schemes are practical as they only increase the size of the secret by a factor of 2. Hence, they are a practical alternative to polynomial and hyperplane based schemes for applications that require more general access structures or shares that are smaller than the secret. More precisely, our contributions are the following:

1. We construct a practical traceable versions of Mignotte's secret sharing scheme that satisfies the private traceability notion of Boneh, Partap and Rotem [6].
2. We give a definition of a semi-publicly traceable secret sharing scheme, where the tracing of the reconstruction box can be performed by anyone, but the verification key is still private.
3. We construct a practical semi-publicly traceable version of Mignotte's secret sharing scheme. The scheme additionally improves the complexity of the tracing algorithm of our first scheme and achieves non-imputability against unbounded adversaries.

We now present the main ideas and contributions of the paper in detail.

Mignotte's secret sharing scheme. In the original t-out-of-n Mignotte scheme, the dealer has access to a public sequence $p_1 < \ldots < p_n$ of coprime integers that satisfies some special properties. The shares of the secret $s < p_1 \cdot \ldots \cdot p_t$ are of the form $\mathsf{sh}_i := (p_i, s_i := s \mod p_i)$. We sometimes call p_i the *identifier* of the share since it is a public value. To reconstruct s with t shares $\mathsf{sh}_{i_1}, \ldots, \mathsf{sh}_{i_t}$, one only needs to solve the system

$$\begin{cases} X = s_{i_1} \mod p_{i_1} \\ \vdots \\ X = s_{i_t} \mod p_{i_t} \end{cases}$$

using the Chinese remainder theorem.

Our first scheme: Traceable Mignotte secret sharing. The sharing algorithm of the first traceable secret sharing scheme is similar to the one of the original Mignotte scheme, except that the p_i are chosen at random from a large sequence $\mathcal{P}$ and are kept secret from everyone except the party that holds the share. As was already observed by Boneh, Partap and Rotem [6], for a traceable secret sharing scheme it is necessary to choose the identifiers of the shares from a large set, since otherwise it is very likely that the tracer chooses a share identifier as input that is already contained in R. In this case the box R cannot reconstruct the secret and we have no guarantees on its behavior.

The key idea behind the tracing algorithm of our scheme is the following: Assume that the box has the shares $(p_1, s_1), \ldots, (p_{t-1}, s_{t-1})$ hardcoded in it. To trace the shares inside R, the tracer queries R on (p_t, s_t^*) and (p_t, s_t^{**}) for some uniform s_t^*, s_t^{**}. For simplicity we assume that both queries yield a set of t distinct consistent shares and the box R always behaves perfectly and outputs s^* and s^{**}, which correspond to the outputs of the reconstruction algorithm on those two set of shares. In this case, the constructive Chinese remainder theorem and Bézout's identity give us a relation between the values $p_1, \ldots, p_t$, the inputs s_t^*, s_t^{**}, the outputs s^*, s^{**} and the Bézout coefficients of $p_1 \cdot \ldots \cdot p_{t-1}$ and p_t. Note that even though we know that those Bézout coefficient exists, we cannot compute them since we don't know the values $p_1 \cdot \ldots \cdot p_{t-1}$ inside the box R. Fortunately, it turns out that we do not need to know the explicit values to leverage their properties in the tracing procedure. A careful analysis yields that the following system of equations over $\mathbb{Z}$ with indeterminates X and Y is solvable with at least constant probability:

$$\begin{cases} s^* &= p_t X + s_t^* Y \\ s^{**} &= p_t X + s_t^{**} Y. \end{cases} \tag{1}$$

Denote the solution of the system by (x, y). We show that the corrupted $p_1, \dots, p_{t-1}$ always divide y but any other p_i from the public sequence does not divide it with good probability. Hence, if only $t-1$ elements from the sequence $\mathcal{P}$ divide y, the tracing algorithm can terminate and output those elements.

We note that the size of the sequence $\mathcal{P}$ has to be chosen carefully since the tracing algorithm basically has to iterate through the entire sequence, when determining which elements divide y. On the other hand, we need it to be big enough to avoid collisions of the p_i hardcoded in R and the ones queried by the tracing algorithm.

The size of $\mathcal{P}$ is also important for the non-imputability property. Let's assume that it is of size 2^κ for some positive integer κ. To make the scheme non-imputable we give the sharing algorithm a hash function H and let it construct the tracing key and verification key as follows: Whenever it samples an element p_i from $\mathcal{P}$ it also samples a random string r_i from $\{0,1\}^\kappa$. It then computes $s_i = s \mod p_i$ and $h_i = H(s_i, p_i, r_i)$ and adds (i, h_i) to the tracing key and the verification key. This can be seen as a commitment to the share of party i. Now, if the tracing algorithm finds one corrupted element p_i it can also compute the corresponding s_i and then link it to party i by taking s_i, p_i as the first two inputs to H and then iterating through all possible $r_i \in \{0,1\}^\kappa$. This takes at most 2^κ operations. It then sends (i, s_i, p_i, r_i) to the verifier. The verifier checks if $h_i = H(s_i, p_i, r_i)$ and accepts or rejects accordingly. In order to frame an innocent party, an adversary would need to guess the correct p_i and r_i for party i, which amounts to finding a preimage of H. If we model H as a random oracle, this corresponds to finding the preimage of a random oracle with min-entropy at least $2^{2\kappa}$, even if the adversary knows the secret and therefore knows the correct s_i for each p_i. We therefore have a quadratic gap between the tracing complexity and the security of non-imputability.

Tracing more general access structures. The original Mignotte secret sharing scheme can be extended to allow for more general access structures by changing how to choose the elements $p_1, \dots, p_n$. To obtain a weighted secret sharing scheme one can make some p_i significantly larger than others, such that those parties need less than $t-1$ additional shares to recover the secret. To obtain an access structure in which some t parties should not be able to recover the secret, one can give those parties integers p_i that are not all pairwise coprime.

To trace Mignotte's scheme with more general access structures one only needs to carefully adapt how the sharing algorithm of our scheme chooses the p_i, since it can not choose them uniformly from one sequence $\mathcal{P}$ anymore. However, each p_i should still be chosen from a large enough set such that it can not be guessed by any adversary. We note that for very complicated access structures, it might take more tries for the tracing algorithm to find a set of input queries that, together with the shares hardcoded in R, yields a set of coprime p_i with which R can reconstruct a secret.

On the complexity of the tracing algorithm. As described above, our tracing algorithm basically proceeds in two phases. In the first phase it queries the reconstruction box R until it obtains "good" outputs. This phase takes time polynomial in the inverse of the success probability of R. In the second phase the algorithm computes a value y and then checks for all p_i from a superpolynomial sized set P, if they divide the value y. This phase takes up to time 2^κ. We note that this is not an issue because the tracing algorithm does not need to be run regularly. It only needs to be run in emergency situations, where one suspects that some parties have leaked their shares, which we don't expect to happen on a regular basis. As long as its running time is feasible, the mere existence of a tracing algorithm already prevents parties from leaking their shares. Jumping ahead, if there exist applications in which one needs a more efficient tracing algorithm, one can use the semi-publicly traceable version of our scheme which we describe below.

On public traceability. The traceability notion of Boneh, Partap and Rotem [6] does not assume that the parties have access to the tracing key or the verification key, which means that the tracing key is not public, i.e., the scheme is not *publicly* traceable. We note that this is necessary in both schemes in [6] and also in our scheme since in all cases the tracing key allows one to check if any given input (claimed to be a share) really belongs to some party. For example, in the traceable version of Shamir's secret sharing scheme of [6], the tracing key consists of the values $F(x_1), \dots, F(x_n)$, where F is a one-way function and for all

$i \in [n]$, the shares are of the form (x_i, y_i) for random secret field elements x_i, y_i. Now, if the tracer queries R on a random input (x^*, y^*) and R has access to the tracing key, it can just compute $F(x^*)$ to check if a real share contains x^* and output $\perp$ whenever it does not. To ensure non-imputability, $x_1, \ldots, x_n$ need to be hidden from the tracer so the probability that the tracer chooses an x^* that is contained in $x_1, \ldots, x_n$ is very small. In our scheme the box R obtains as input pairs of the form (s^*, p^*). Given the tracing key, the box R can find out if the pair is an actual share by plugging s^* and p^* into the first two arguments of the hash function H and then iterating through all $r_i \in \{0,1\}^\kappa$ and checking if any $H(s^*, p^*, r_i)$ is contained in the tracing key.

One way to turn our construction into a publicly traceable scheme is to require the reconstruction box R to answer queries in a timely manner such that there is not enough time for R to iterate through all $r_i \in \{0,1\}^\kappa$ beforehand. Note that this would not make the traceable version of Shamir's scheme in [6] publicly traceable since here the box only needs to perform one function evaluation to check if the input is consistent with the tracing key.

Semi-publicly traceable secret sharing. If we don't want to introduce a timing restriction on the reconstruction box, it seems unlikely that we can achieve public traceability with our techniques. We therefore introduce a weaker notion that we call *semi-public traceability*. In a semi-publicly traceable secret sharing scheme we remove the tracing key but keep the private verification key. The tracing algorithm that interacts with the reconstruction box R outputs a proof π that incriminates the corrupted parties but it does not necessarily output the identifiers of the corrupted parties explicitly. The verifier, given the proof π and the verification key vk, extracts the corrupted parties from the proof and outputs the identifiers of the parties together with π.

We believe that semi-publicly traceable secret sharing is a very useful definition since anyone that gets access to the reconstruction box R can perform the tracing procedure, even the shareholders. In practice we probably cannot hope for the party that somehow gets access to the reconstruction box to also know the private tracing key. However, the definition is not strictly stronger than the definition in [6] since there the tracing algorithm is required to output the identifiers of the parties explicitly, whereas in semi-public tracing the tracer just outputs a proof that incriminates the parties and might not even learn the identifiers itself. We believe that this weakening is reasonable since the identifiers can eventually be published or sent to the tracer by the verifier. The fact that in a public tracing scheme the tracer does not learn who the corrupted parties are can even be a privacy feature: Imagine that Alice shared her secret among n parties and Bob gets access to a reconstruction box containing shares of Alice's secret. If Bob can find out which parties' shares are in R, then Bob would also learn that Alice has used the services of those parties.

A semi-public Mignotte secret sharing scheme. We construct a practical semi-publicly traceable secret sharing scheme based on the Chinese Remainder Theorem. The sharing algorithm is the same as the one of our first scheme, except that the dealer does not construct a tracing key and the verification key contains the p_i given to the parties in the clear. Then we adapt the tracing algorithm as follows: The tracer still tries to solve the system in Equation (1) as in the original scheme to obtain an element y that is divisible by all corrupted p_i. Once it has such a y, it can output is as the proof. Since the verifier knows which of the elements in $\mathcal{P}$ were given to parties by the verifier, it just needs to check which of those n elements divide y. If it finds f such elements, it outputs the corresponding identifiers and the element y. If the reconstruction box is not perfect, i.e., only outputs the honest y with probability ϵ, we cannot exclude the case that the reconstruction box R outputs values that are not correlated with the secret but are of the form such that System (1) is solvable. Unfortunately, in the semi-publicly traceable scheme, the tracer is not able to check if the y is of the correct form. To circumvent this problem we run trace until it has obtained sufficiently many outputs y such that with high probability at least one of the y is of the correct form. The tracer then publishes all the values as the proof and the verifier can find the correct y using the verification key. The scheme satisfies non-imputability since a malicious tracer that wants to incriminate an honest party needs to guess at least one p_i that was given to a party but is not contained in R. As long as the sequence $\mathcal{P}$ is large enough, the probability of this event is negligible.

Note that our semi-public scheme additionally improves on our first scheme in two ways: First, the tracing algorithm now runs in expected polynomial time. Secondly, this scheme is the first traceable secret sharing scheme in which non-imputability holds against computationally unbounded adversaries. We give an overview of all traceable secret sharing schemes in Table 1.

1.2 Other Related Work

Traitor-tracing schemes. Traitor tracing for broadcast encryption schemes, introduced by Chor, Fiat and Naor [12], allow a tracer to trace back leaked decryption keys. The techniques used in the long line of traitor tracing schemes [8, 24, 27, 4, 14, 29, 21, 26, 22, 13, 10, 7, 5, 16, 9, 18, 11, 32, 31, 17], however, are different from the one used in [6] and in this work. See [6] for a more detailed description of the techniques.

Weighted CRT based secret sharing schemes. Zuo et al. [33] extend CRT based secret sharing schemes to allow for weighted multi-secret sharing. Garg et al. [15] construct a weighted ramp secret-sharing scheme based on the CRT. A ramp secret sharing scheme is parameterized by two thresholds t and t', where t is the reconstruction threshold and any collection of parties with cumulative weight less than t' should learn nothing about the secret. Ning et al. [28] extended CRT based secret sharing over $\mathbb{Z}_N$ to polynomial rings over finite fields.

1.3 Organization of the Paper

We begin in Section 2 by recalling basic definitions and results from number theory and secret sharing schemes. In Section 3 we present the first traceable secret sharing scheme based on the Chinese remainder theorem. The scheme satisfies the notion of [6]. In Section 4 we present our definition of semi-public traceable secret sharing and construct the first practical scheme that satisfies this definition. In Section 5 we give an overview of all traceable secret sharing schemes. Finally, we conclude in Section 6 with a discussion of the open problems.

2 Preliminaries

2.1 Number Theory

We will need the following basic results from number theory.

Theorem 1 (Chinese Remainder Theorem). *Let $p_1, \ldots, p_k$ be pairwise coprime integers and $v_1, \ldots, v_k$ arbitrary integers. Then the system*

$$\begin{cases} X = v_1 \mod p_1 \\ \vdots \\ X = v_k \mod p_k \end{cases}$$

has a unique solution modulo $p_1 \cdots p_k$.

Theorem 2 (Bézout's Identity). *Let x, y be coprime integers. There exist integers a, b such that $ax + by = 1$.*

We call the integers a, b above *Bézout coefficients.* They are not unique.

Theorem 3 (Constructive Chinese Remainder Theorem for 2 equations). *Let p_1, p_2 be coprime integers and let a, b be Bézout coefficients of p_1, p_2, i.e., $ap_1 + bp_2 = 1$. Then the system*

$$\begin{cases} X = v_1 \mod p_1 \\ X = v_2 \mod p_2 \end{cases}$$

has a solution $X = v_1 b p_2 + v_2 a p_1$.

2.2 Traceable Threshold Secret Sharing

We mostly follow the definition in [6]. However, we need to weaken the required privacy notion since Mignotte's secret sharing scheme is not perfectly private.

Definition 1 (Traceable Threshold Secret Sharing)**.** A t-out-of-n traceable threshold secret sharing scheme is a tuple of algorithms $(\mathsf{Share}, \mathsf{Rec}, \mathsf{Trace}, \mathsf{Verify})$ defined as follows:

$\mathsf{Share}(1^\lambda, n, t, s) \to (\mathsf{sh}_1, \ldots, \mathsf{sh}_n, \mathsf{tk}, \mathsf{vk})$ is a randomized algorithm that takes as input the security parameter 1^λ, the number of parties n, the threshold $t \leq n$ and the secret $s \in \mathcal{S}$. It outputs n shares $\mathsf{sh}_1, \ldots, \mathsf{sh}_n$, a tracing key tk and a verification key vk.

$\mathsf{Rec}(\mathsf{sh}_{i_1}, \ldots, \mathsf{sh}_{i_t}) \to s$ is a deterministic algorithm that takes as input t shares $\mathsf{sh}_{i_1}, \ldots, \mathsf{sh}_{i_t}$ and outputs a secret s or $\perp$.

$\mathsf{Trace}^R(\mathsf{tk}) \to (I, \pi)$ is a randomized algorithm that takes as input the tracing key tk. It also gets oracle access to a reconstruction box R. It outputs a subset $I \subseteq [n]$ of indices that identify corrupted parties and a proof π.

$\mathsf{Verify}(\mathsf{vk}, I, \pi) \to \{0, 1\}$ is a deterministic algorithm that takes as input the verification key vk, a set of indices I and a proof π that the corresponding parties are corrupted. It outputs 0 or 1 indicating whether it accepts the proof or not.

We call an input $(\mathsf{sh}_{i_1}, \ldots, \mathsf{sh}_{i_{t-f}})$ to the reconstruction box R *consistent* if R contains f shares $(\mathsf{sh}_{i_{t-f+1}}, \ldots, \mathsf{sh}_{i_t})$ such that $(\mathsf{sh}_{i_1}, \ldots, \mathsf{sh}_{i_t})$ are pairwise distinct, $\mathsf{Rec}(\mathsf{sh}_{i_1}, \ldots, \mathsf{sh}_{i_t})$ outputs a valid secret and the distribution of $(\mathsf{sh}_{i_1}, \ldots, \mathsf{sh}_{i_{t-f}})$ is indistinguishable from the distribution of $t - f$ shares output by Share. We require a traceable threshold secret sharing scheme to satisfy the following properties:

Perfect Correctness: For any $T \subseteq [n]$ with $|T| = t$ and any secret $s \in \mathcal{S}$, it holds that

$$\Pr[\mathsf{Rec}(\mathsf{Share}(1^\lambda, n, t, s)_T) = s] = 1,$$

where the probability is taken over the random coins of Share.

$\varepsilon-$**Privacy:** For any $T^* \subseteq [n]$ with $|T^*| < t$, any unbounded adversary $\mathcal{A}$ and a uniformly random secret $s \leftarrow \mathcal{S}$, it holds that

$$\Pr[\mathcal{A}(\mathsf{Share}(1^\lambda, n, t, s)_{T^*}) = s] \leq \varepsilon,$$

where the probability is taken over the random coins of Share and $\mathcal{A}$. If $\varepsilon = 1/|\mathcal{S}|$, we call the scheme *perfectly private*.

Traceability: For every probabilistic polynomial time adversary $\mathcal{A}$, the probability that it wins the game $\mathbf{GTrace}_{\mathcal{A},\mathsf{TSS},\epsilon}(\lambda)$ defined in Figure 1 is negligible in λ.

Non-Imputability: For every probabilistic polynomial time adversary $\mathcal{A}$, the probability that it wins the game $\mathbf{GNon\text{-}Imputability}_{\mathcal{A},\mathsf{TSS}}(\lambda)$ defined in Figure 2 is negligible in λ.

2.3 Mignotte's Secret Sharing Scheme

Let t, n be integers such that $n \geq 2$ and $2 \leq t \leq n$. We call a sequence of pairwise coprime integers $p_1 < p_2 < \ldots < p_n$, where the product of any $t-1$ elements is strictly less than the product of any t elements, i.e., $p_{n-t+2} \cdot \ldots \cdot p_n < p_1 \cdot \ldots \cdot p_t$, a (t, n)-*Mignotte sequence.* Given a publicly known (t, n)-Mignotte sequence, Mignotte's secret sharing scheme is defined as follows.

- The secret s is a random integer, such that $\beta < s < \alpha$, where $\alpha := p_1 \cdot \ldots \cdot p_t$ and $\beta := p_{n-t+2} \cdot \ldots \cdot p_n$.
- The shares s_i are set to $s_i := s \mod p_i$.

GTrace$_{\mathcal{A},\mathsf{TSS},\epsilon}(\lambda)$

1. $\mathcal{A}(1^\lambda, n, t)$ outputs (I, state), where $I \subset [n]$ is the set of parties to corrupt and $|I| < t$.
2. Secret s is chosen uniformly at random from $\mathcal{S}$.
3. $\mathsf{Share}(1^\lambda, n, t, s)$ outputs $(\mathsf{sh}_1, \ldots, \mathsf{sh}_n, \mathsf{tk}, \mathsf{vk})$.
4. On input all shares of parties in I, $\mathcal{A}(\mathsf{state}, \mathsf{sh}_{i_1}, \ldots, \mathsf{sh}_{i_{|I|}})$ outputs reconstruction box R.
5. $\mathsf{Trace}^R(\mathsf{tk})$ outputs (I', π).
6. $\mathcal{A}$ wins if R reconstructs the secret from good inputs with probability at least ϵ and either $I \neq I'$ or $\mathsf{Verify}(\mathsf{vk}, I', \pi) = 0$.

Figure 1: The tracing game for traceable threshold secret sharing TSS.

GNon-Imputability$_{\mathcal{A},\mathsf{TSS}}(\lambda)$

1. $\mathcal{A}(1^\lambda, n, t)$ outputs (i^*, s, state).
2. $\mathsf{Share}(1^\lambda, n, t, s)$ outputs $(\mathsf{sh}_1, \ldots, \mathsf{sh}_n, \mathsf{tk}, \mathsf{vk})$.
3. On input all shares except for the i^*-th one and the keys tk and vk, $\mathcal{A}(\mathsf{state}, \mathsf{sh}_1, \ldots, \mathsf{sh}_{i^*-1}, \mathsf{sh}_{i^*+1}, \ldots, \mathsf{sh}_{i_n}, \mathsf{tk}, \mathsf{vk})$ outputs (I^*, π).
4. $\mathcal{A}$ wins if $i^* \in I^*$ and $\mathsf{Verify}(\mathsf{vk}, I^*, \pi) = 1$.

Figure 2: The non-imputability game for traceable threshold secret sharing TSS.

- Given t distinct shares $s_{i_1}, \ldots, s_{i_t}$ the secret is recovered as the unique solution modulo $p_{i_1} \cdots p_{i_t}$ of the system
$$\begin{cases} X = s_{i_1} \mod p_{i_1} \\ \vdots \\ X = s_{i_t} \mod p_{i_t} \end{cases}$$
using the Chinese Remainder Theorem.

The scheme is correct because s is an integer solution of the above scheme and $s < \alpha \le p_{i_1} \cdots p_{i_t}$. Given only $t-1$ distinct shares $s_{i_1}, \ldots, s_{i_{t-1}}$, one can only tell that $s = s_0 \mod p_{i_1} \cdots p_{i_{t-1}}$, for some $s_0 < \beta < s$. Hence, at least $(\alpha - \beta)/\beta$ possible secrets remain that all have the same probability, i.e., the scheme satisfies $\beta/(\alpha-\beta)$-privacy. Next we show how to construct a Mignotte sequence such that $(\alpha-\beta)/\beta$ is big enough. We need the following fact [23, page 9].

Lemma 1. *For any integers $2 \le t \le n$, there exist arbitrarily large integers ℓ such that P_ℓ is the ℓ-th prime number and there are at least n primes in the interval $(P_\ell^{(t^2-1)/t^2}, P_\ell]$.*

Let $p_1, \ldots, p_n$ be the n last primes from the interval $(P_\ell^{(t^2-1)/t^2}, P_\ell]$. They form a Mignotte sequence, since
$$\alpha = p_1 \cdots p_t \ge P_\ell^{(t^2-1)/t} > P_\ell^{t-1} \ge p_{n-t+2} \cdot \ldots \cdot p_n = \beta.$$
Further, we get that
$$\frac{\alpha - \beta}{\beta} \ge \frac{p_1^t}{p_n^{t-1}} - 1 \ge \frac{P_\ell^{(t^2-1)/t}}{P_\ell^{t-1}} - 1 = \frac{P_\ell}{P_\ell^{1/t}} - 1.$$

This means that given $t-1$ shares, there are $\frac{P_\ell}{P_\ell^{1/t}} - 1$ possible values for any other share. If we set the size of P_ℓ to $2^{t\rho/(t-1)}$ for some positive integer ρ, we get that the number of remaining possibilities is at least $2^\rho - 1$. We call the Mignotte sequence obtained with the procedure above a (t, n, ρ)-Mignotte sequence.

3 Traceable Mignotte Secret Sharing

The scheme MTSS is presented in Figure 3. For simplicity we assume that Trace obtains the number of corruptions f as input. We later explain how we can remove this requirement. We make the following changes to the sharing algorithm of the original secret sharing scheme: Instead of giving the algorithm a Mignotte sequence of size n as input, we give it a larger sequence and let Share randomly sample the p_i from the larger sequence. Further, Share also constructs a tracing key tk and a verification key vk using a hash function H. The reconstruction algorithm is the same as in the original scheme.

3.1 Proof of Traceability

Theorem 4. *Let κ be a positive integer and $p_1, \ldots, p_{2^\kappa}$ be a $(t, 2^\kappa, \rho)$-Mignotte sequence. Let H be a hash function with input space $\{0,1\}^{3\kappa}$. For $\mathcal{P} = \{p_1, \ldots, p_{2^\kappa}\}$ and $\rho \geq 3$ we get that MTSS is a t-out-of-n traceable threshold secret sharing scheme in the random oracle model with the following properties:*

1. *For any adversary $\mathcal{A}$, the probability of winning **GTrace**$_{\mathcal{A},\mathsf{MTSS},\epsilon}(\lambda)$ is at most $n/2^\kappa \cdot 1/(2^\rho - 1)$.*
2. *For any adversary $\mathcal{A}$, the probability of winning **GNon-Imputability**$_{\mathcal{A},\mathsf{MTSS}}(\lambda)$ is at most $1/2^{2\kappa}$.*

Before proving the theorem, we analyze the probability that Trace terminates in one round.

Definition 2 (Good pair of responses)**.** We call the pair of responses (u, u') output in Step 4 of MTSS *good* if $\beta < u, u' < \alpha$ and we have that u (and respectively u') corresponds to the output of $\mathsf{Rec}(\mathsf{sh}_{i_1}, \ldots, \mathsf{sh}_{i_t}, \alpha, \beta)$, where exactly $t-f$ of the shares $\{\mathsf{sh}_{i_j}\}_{j\in[t]}$ are of the form $((z_1, q_1), (z_2, q_2), \ldots, (z_{t-f}, q_{t-f}))$ (or respectively $((z_1', q_1), (z_2, q_2), \ldots, (z_{t-f}, q_{t-f}))$).

Lemma 2. *Let κ be a positive integer, $p_1, \ldots, p_{2^\kappa}$ a $(t, 2^\kappa, \rho)$-Mignotte sequence and set $\mathcal{P} = \{p_1, \ldots, p_{2^\kappa}\}$. Let (u, u') be a good pair of responses. Then we have that Trace of MTSS terminates with probability at least $1/4 - n/2^{\kappa+2}$.*

Proof. The proof consists of two steps. We first show that the system in Equation (2) has a unique solution with probability at least $1/4$. Then we show that in this case all corrupted p_j divide y and with all but at most $n/2^\kappa$ probability none of the not corrupted p_j that correspond to the shares divide y. Let $p_{i_1}, \ldots, p_{i_f}$ denote the p_i's corresponding to the corrupted shares. We start with the first step. By our definition of (u, u') being good we can assume that they do not intersect with $q_1, \ldots, q_{t-f}$. For simplicity of notation, let us relabel $p_{i_f+1} := q_2, p_{i_f+2} := q_3, \ldots, p_{i_{t-1}} := q_{t-f}$. Again, since (u, u') are good, we know that u is the unique solution modulo $q_1 \prod_{j=1}^{t-1} p_{i_j}$ of the system.

$$\begin{cases} S = u_0 \mod p_{i_1} \cdots p_{i_{t-1}} \\ S = z_1 \mod q_1 \end{cases}$$

for some $u_0 \in \mathbb{Z}_{p_{i_1} \cdots p_{i_{t-1}}}$. Let $a, b \in \mathbb{Z}$ be Bézout coefficients of q_1 and $p_{i_1} \cdots p_{i_{t-1}}$, i.e.,

$$1 = a \cdot q_1 + b \cdot p_{i_1} \cdots p_{i_{t-1}},$$

where $|a| < p_{i_1} \cdots p_{i_{t-1}}$ and $|b| < q_1$. They are guaranteed to exist by the extended euclidean algorithm. By Theorem 3 we know that

$$u = a u_0 q_1 + b z_1 p_{i_1} \cdots p_{i_{t-1}} \mod q_1 \prod_{j=1}^{t-1} p_{i_j}. \tag{3}$$

$\mathsf{Share}(1^\lambda, n, t, s, H, \mathcal{P})$:

```
1. For all $i \in [n]$ do:
    (a) Sample $p_i \leftarrow \mathcal{P}$ uniformly at random.
    (b) Set $s_i = s \mod p_i$ and $\mathsf{sh}_i = (s_i, p_i)$.
    (c) Sample $r_i \leftarrow \{0,1\}^\kappa$ uniformly at random.
    (d) Set $\mathsf{tk}_i = (i, h_i)$ for $h_i = H(s_i, p_i, r_i)$.
2. Set $\mathsf{tk} = \mathsf{vk} = (\mathsf{tk}_1, \ldots, \mathsf{tk}_n)$.
3. Output $(\mathsf{sh}_1, \ldots, \mathsf{sh}_n, \mathsf{tk}, \mathsf{vk})$.
```

$\mathsf{Rec}(\mathsf{sh}_{i_1}, \ldots, \mathsf{sh}_{i_t}, \alpha, \beta)$:

```
1. Try to solve the following system of equations using the Chinese Remainder Theorem:
```

$$\begin{cases} X = s_{i_1} \mod p_{i_1} \\ \vdots \\ X = s_{i_t} \mod p_{i_t}. \end{cases}$$

```
   If it is not possible, output $\perp$. Otherwise, denote the solution of the system by $x$.
2. If $x \in (\beta, \alpha)$, output $x$. Otherwise, output $\perp$.
```

$\mathsf{Trace}^R(f, \mathsf{tk})$:

```
1. Set $I, \pi = \emptyset$.
2. Choose $q_1, \ldots, q_{t-f} \leftarrow \mathcal{P}$ uniformly at random and independently sample $z_j \leftarrow [0, q_j - 1]$ uniformly at random for all $j \in [t-f]$.
3. Sample $z'_1 \leftarrow [0, q_1 - 1] \setminus \{z_1\}$ uniformly at random.
4. Query $R$ on $((z_1, q_1), (z_2, q_2), \ldots, (z_{t-f}, q_{t-f}))$ and on $((z'_1, q_1), (z_2, q_2), \ldots, (z_{t-f}, q_{t-f}))$. Let $u$ and $u'$ be the responses.
5. Try to solve the following system of equations with indeterminates $X$ and $Y$ over $\mathbb{Z}$:
```

$$\begin{cases} u = Xq_1 + z_1 Y \\ u' = Xq_1 + z'_1 Y. \end{cases} \tag{2}$$

```
   If it is not possible, go to Step 1. Otherwise denote the solution of the system by $(x, y)$.
6. For all $p_j \in \mathcal{P} \setminus \{q_1, \ldots, q_{t-f}\}$ check if $p_j$ divides $y$. If it does, compute $s_j := u \mod p_j$ and check for all $r \in \{0,1\}^\kappa$, if $H(s_j, p_j, r)$ is contained in $\mathsf{tk}$. If it is true that $h_j = H(s_j, p_j, r_j)$ for some $h_j \in \mathsf{tk}$ and some $r_j \in \{0,1\}^\kappa$, add the corresponding index $j$ to $I$ and add $(s_j, p_j, r_j)$ to $\pi$.
7. If $|I| = f$, output $I$ and $\pi$. Otherwise, go to Step 1.
```

$\mathsf{Verify}(\mathsf{vk}, I, \pi)$:

```
1. For all $j \in I$, check if $H(s_j, p_j, r_j) = h_j$.
2. If the above holds for all $j \in I$, output 1. Otherwise, output 0.
```

Figure 3: MTSS: Traceable Mignotte secret sharing when f shares are corrupted.

Similarly, we have that

$$u' = au_0q_1 + bz'_1p_{i_1}\cdots p_{i_{t-1}} \mod q_1\prod_{j=1}^{t-1}p_{i_j}. \tag{4}$$

Now we have that $|au_0q_1| < 2q_1\prod_{j=1}^{t-1}p_{i_j}$, $\left|bz_1p_{i_1}\cdots p_{i_{t-1}}\right| < 2q_1\prod_{j=1}^{t-1}p_{i_j}$ and $\left|bz_1p_{i_1}\cdots p_{i_{t-1}}\right| < 2q_1\prod_{j=1}^{t-1}p_{i_j}$. Note that since q_1 and all p_i are positive, we have that exactly one of a and b is positive and one is negative. Hence, the same holds for au_0q_1 and $bz_1p_{i_1}\cdots p_{i_{t-1}}$. It follows that over $\mathbb{Z}$ we have one of four cases:

Case 1

$$u = au_0q_1 + bz_1p_{i_1}\cdots p_{i_{t-1}}$$

Case 2

$$u - q_1\prod_{j=1}^{t-1}p_{i_j} = au_0q_1 + bz_1p_{i_1}\cdots p_{i_{t-1}}$$

$$\Leftrightarrow u = (au_0 + \prod_{j=1}^{t-1}p_{i_j})q_1 + bz_1p_{i_1}\cdots p_{i_{t-1}}$$

Case 3

$$u = (au_0 - \prod_{j=1}^{t-1}p_{i_j})q_1 + bz_1p_{i_1}\cdots p_{i_{t-1}}$$

Case 4

$$u = (au_0 + 2\prod_{j=1}^{t-1}p_{i_j})q_1 + bz_1p_{i_1}\cdots p_{i_{t-1}}.$$

Similarly, for u' we have one of the four cases over $\mathbb{Z}$:

Case 1

$$u' = au_0q_1 + bz'_1p_{i_1}\cdots p_{i_{t-1}}$$

Case 2

$$u' = (au_0 + \prod_{j=1}^{t-1}p_{i_j})q_1 + bz'_1p_{i_1}\cdots p_{i_{t-1}}$$

Case 3

$$u' = (au_0 - \prod_{j=1}^{t-1}p_{i_j})q_1 + bz'_1p_{i_1}\cdots p_{i_{t-1}}$$

Case 4

$$u' = (au_0 + 2\prod_{j=1}^{t-1}p_{i_j})q_1 + bz'_1p_{i_1}\cdots p_{i_{t-1}}.$$

If one of the cases holds for both u and u', we have that system (2)

$$\begin{cases} u = Xq_1 + z_1Y \\ u' = Xq_1 + z'_1Y \end{cases}$$

has a unique solution $(au_0 + \omega \prod_{j=1}^{t-1} p_{i_j}, bp_{i_1} \cdots p_{i_{t-1}})$ for some $\omega \in \{-1, 0, 1, 2\}$. If u and u' are in different cases, the system is not solvable. In the worst cases we have that each of the cases is equally likely once q_1 is fixed. Hence, we have that z_1 and z'_1 yield the same case with probability at least $1/4$. It follows that the system is solvable for a good pair (u, u') with probability at least $1/4$.

Now we consider the second step of the proof. If (u, u') is good and the system is solvable, then $y = bp_{i_1} \cdots p_{i_{t-1}}$. It is obvious that all of the corrupted p_i divide y. Now we have that $|b| < q_1$ and, since the elements of $\mathcal{P}$ form a Mignotte sequence, at most 1 more element $\tilde{p}$ can divide y. The probability that this $\tilde{p}$ is one of the p_i chosen by the dealer is at most $n/2^\kappa$. This means that with probability at least $1 - n/2^\kappa$, we have $|I| = f$ in Step 7 of Trace, whenever (u, u') is good and system (2) is solvable. It follows that Trace terminates after one round with probability at least $(1 - n/2^\kappa)/4 = 1/4 - n/2^{\kappa+2}$. □

We are now ready for the proof of our main theorem.

Proof of Theorem 4. Correctness and $1/(2^\rho - 1)$-privacy of the scheme follow by correctness and privacy of the original scheme. We begin with proving the first property of the theorem. By Lemma 2 we know that Trace finds exactly f corrupted shares and terminates in one fixed round with probability at least $1/4 - n/2^{\kappa+2}$. It remains to show that when (u, u') is not good, the probability that Trace terminates with a false set I of size f in Step 6 is at most $n/2^\kappa \cdot 1/(2^\rho - 1)$, since in this event, the adversary guesses at least one of the p_i that was chosen by the dealer (but not given to it) and the corresponding s_i correctly. Assume that (u, u') is not (necessarily) good, Trace terminates in Step 7 but some $i \in I$ output by Trace is not one of the corrupted parties. This means that one can use Trace and the adversary that plays the game $\mathbf{GTrace}_{\mathcal{A},\mathsf{TSS},\epsilon}(\lambda)$ to find the preimage of h_i. In particular, one can find one of the p_i that was chosen by the dealer and the corresponding s_i. If we model H as a random oracle, the probability of this event is at most $n/2^\kappa \cdot 1/(2^\rho - 1)$ because $n/2^\kappa$ is the probability of guessing a correct p_i and $1/(2^\rho - 1)$ is the probability of guessing the correct s_i given p_i and at most $t - 1$ shares.

It remains to prove the second property. The scheme is non-imputable in the random oracle model by the following observation: Any adversary $\mathcal{A}$ that wins the game $\mathbf{GNon\text{-}Imputability}_{\mathcal{A},\mathsf{TSS}}(\lambda)$ finds the preimage of some h_i for $(i, h_i) \in \mathsf{tk}$. If we model H as a random oracle, even given the secret s, it has min-entropy $2^{2\kappa}$, since the last two entries of H are uniform and independent. Hence, the probability of this event is at most $1/2^{2\kappa}$. □

Remark 1 (On the input f)**.** So far we have assumed for simplicity that the number of corruptions f is known by the tracer, which might not be the case in practice. We note that knowing the number of corruptions is not necessary for our tracing algorithm since we have seen in the proof of Theorem 4 that Trace mistakes an honest party for a corrupted one with probability at most $n/2^\kappa \cdot 1/(2^\rho - 1)$. By setting the parameters n, κ, ρ such that this probability is negligible, we can remove the input of f to Trace. The tracing algorithm can then learn f by trying $f = 1, 2, \ldots$ until it terminates in Step 7.

3.2 Complexity of Trace

Theorem 5. *Let κ be a positive integer and $p_1, \ldots, p_{2^\kappa}$ be a $(t, 2^\kappa, \rho)$-Mignotte sequence. Let H be a hash function with input space $\{0,1\}^{3\kappa}$. For $\mathcal{P} = \{p_1, \ldots, p_{2^\kappa}\}$, $\rho \geq 3$ and $t \geq \rho + 1$, we get that* Trace *runs in expected time $O(\epsilon^{-1} \cdot f \cdot 2^\kappa)$, where ϵ is the probability that R reconstructs the secret on a good input.*

To compute the running time of Trace we need to compute the probability of (u, u') being good. Then the claim follows from Lemma 2.

Lemma 3. *Let κ be a positive integer and $p_1, \ldots, p_{2^\kappa}$ be a $(t, 2^\kappa, \rho)$-Mignotte sequence. Let ϵ is the probability that R reconstructs the secret on a good input. For $\mathcal{P} = \{p_1, \ldots, p_{2^\kappa}\}$, $\rho \geq 3$ and $t \geq \rho + 1$, there exist a constant c such that a pair (u, u') is good with probability ϵ/c.*

Proof. The pair (u, u') is good, whenever all of the following events occur:

A: $q_1, \ldots, q_{t-f}$ are pairwise different and do not intersect $p_{i_1}, \ldots, p_{i_f}$.

B: The shares $(q_1, z_1), \ldots, (q_{t-f}, z_{t-f}), (p_{i_1}, s_{i_1}), \ldots, (p_{i_f}, s_{i_f})$ are consistent, i.e given those shares as input Rec recovers a secret $s \in (\beta, \alpha)$. The same needs to hold when replacing (q_1, z_1) with (q_1', z_1').

C: R outputs $u = \mathsf{Rec}((q_1, z_1), \ldots, (q_{t-f}, z_{t-f}), (p_{i_1}, s_{i_1}), \ldots, (p_{i-f}, s_{i-f}))$ and

$$u' = \mathsf{Rec}((q_1', z_1'), \ldots, (q_{t-f}, z_{t-f}), (p_{i_1}, s_{i_1}), \ldots, (p_{i-f}, s_{i-f})).$$

We start with event A. Fix $p_{i_1}, \ldots, p_{i_f}, q_1, \ldots, q_{t-f-1}$. The probability that a uniformly chosen $q_{t-f} \in \mathcal{P}$ is contained in that set is $(t-1)/2^\kappa$. By a union bound, we get that $p_{i_1}, \ldots, p_{i_f}, q_1, \ldots, q_{t-f}$ are pairwise distinct except with probability at most $(t-f)(t-1)/2^\kappa$.

Now consider event B. The probability that the shares are consistent is at least $(2^\rho - 1)/2^{t\rho/(t-1)} \geq 1/2 - 2^{-\rho}$. This can be seen by the following argument: Fix the first $t-2$ shares $((s_{i_1}, p_{i_1}), (s_{i_2}, p_{i_2}), \ldots, (s_{i_{t-2}}, p_{i_{t-2}}))$. Those shares determine that $u = \tilde{u} \mod \prod_{j\in[t-2]} p_{i_j}$ for some $\tilde{u} < \prod_{j\in[t-2]} p_{i_j}$. Hence, the number of possibilities for the secret u are now

$$\frac{\alpha - \beta}{\prod_{j\in[t-2]} p_{i_j}} \geq \frac{P_\ell^{(t^2-1)/t}}{P_\ell^{t-2}} - P_\ell = P_\ell(P_\ell^{1-1/t} - 1) \geq P_\ell \geq p_n.$$

We follow that any choice of the first $t-1$ shares is valid to obtain a consistent query. Now fix any choice for the first $t-1$ shares. We know that then there are $(\alpha - \beta/\beta) \geq P_\ell/P_\ell^{1/t} - 1$ possible secrets left. Hence, the probability that the last share (z_{t-f}, q_{t-f}) is consistent with the fixed shares is at least $P_\ell/P_\ell^{1+1/t} - 1/P_\ell = 1/P_\ell^{1/t} - 1/P_\ell$. Plugging in $P_\ell = 2^{t\rho/(t-1)}$ and $t \geq \rho + 1$ we get that the probability is at least $1/2 - 2^{-\rho-1}$.

We now consider event C. By definition of the game $\mathbf{GTrace}_{\mathcal{A},\mathsf{TSS},\epsilon}(\lambda)$, we know that on input $t-f$ real shares, R outputs the secret with probability at least ϵ. Note that, whenever the shares are consistent, the queries to R are at statistical distance at most $P_\ell^{(1-t^2)/t^2}$ from a real query, since the distribution is the same except for the fact that one query can never be 0.

The pair (u, u') is good if all of the events A, B, C hold. That is

$$\begin{aligned}\Pr[(u,u') \text{ is good}] &= \Pr[A] \cdot \Pr[B \mid A] \cdot \Pr[C \mid A \cap B] \\ &\geq \left(1 - \frac{(t-f)(t-1)}{2^\kappa}\right)\left(\frac{1}{2} - 2^{\rho-1}\right)^2 \epsilon,\end{aligned}$$

which concludes the proof. □

Proof of Theorem 5. From Lemma 3 and Lemma 2, we follow that the probability that Trace terminates in one fixed round is at least

$$\left(1 - \frac{(t-f)(t-1)}{2^\kappa}\right)\left(\frac{1}{2} - 2^{\rho-1}\right)^2\left(\frac{1}{4} - \frac{n}{2^{\kappa+2}}\right)\epsilon = \epsilon/c$$

for some constant c. In a single round Trace needs to make at most $2^\kappa(f+1)$ queries to H. We follow that Trace runs in expected time $O(\epsilon^{-1} \cdot f \cdot 2^\kappa)$. □

Remark 2 (Removing the bound on t)**.** Note that in Theorem 5 we need to assume that $t \geq \rho + 1$. This is necessary because for smaller t it could become infeasible to find a set of consistent shares. We can circumvent the need for a large t as follows: Use a $(\tilde{t}, 2^\kappa, \rho)$-Mignotte sequence, where $\tilde{t}$ is the first integer that's larger than $\rho + 1$ and divisible by t. The Share algorithm first constructs $\tilde{t}n/t$ shares as it would for a $\tilde{t}$-out-of-$(\tilde{t}n/t)$ secret sharing scheme. Then it gives $\tilde{t}/t$ shares to every party to obtain a t-out-of-n secret sharing scheme. Note that this increases the size of the shares by a factor of $\tilde{t}/t$. Hence, choosing a larger t is reasonable not only for security but also for efficiency of the traceable secret sharing scheme.

4 Semi-Public Traceable Secret Sharing

In this section we present our definition of semi-public traceable secret sharing and give a construction based on the Chinese Remainder Theorem.

4.1 Definition

In a semi-public traceable secret sharing scheme the tracing procedure can be performed by anyone that has access to the reconstruction box R. However, the verification can only be performed with a private verification key. The differences of our definition below compared to the definition of traceable secret sharing by Boneh, Partap and Rotem [6] are the following:

1. We remove the private tracing key.
2. Trace does not need to output the explicit identifiers of the corrupted parties but only a proof π that incriminates them.
3. Verify extracts the identifiers of the corrupted parties and outputs them together with π.
4. The adversary in the non-imputability game does not obtain vk as input. This is necessary since our scheme can be traced by anyone but the verification key is private.
5. We assume computationally unbounded adversaries in our security definitions. One could also consider a semi-public scheme that achieves traceability and non-imputability against polynomial time adversaries but we don't need to bound the adversaries in our security proofs.

Definition 3 (Semi-publicly traceable threshold secret sharing)**.** A t-out-of-n semi-publicly traceable threshold secret sharing scheme is a tuple of efficient algorithms $(\mathsf{Share}, \mathsf{Rec}, \mathsf{Trace}, \mathsf{Verify})$ defined as follows:

$\mathsf{Share}(1^\lambda, n, t, s) \to (\mathsf{sh}_1, \dots, \mathsf{sh}_n, \mathsf{vk})$ is a randomized algorithm that takes as input the security parameter 1^λ, the number of parties n, the threshold $t \leq n$ and the secret $s \in \mathcal{S}$. It outputs n shares $\mathsf{sh}_1, \dots, \mathsf{sh}_n$, a verification key vk.

$\mathsf{Rec}(\mathsf{sh}_{i_1}, \dots, \mathsf{sh}_{i_t}) \to s$ is a deterministic algorithm that takes as input t shares $\mathsf{sh}_{i_1}, \dots, \mathsf{sh}_{i_t}$ and outputs a secret s or $\perp$.

$\mathsf{Trace}^R(f) \to \pi$ is a randomized algorithm that gets oracle access to a reconstruction box R. It outputs a proof π.

$\mathsf{Verify}(\mathsf{vk}, \pi) \to (I, \pi)$ is a deterministic algorithm that takes as input the verification key vk and a proof π. From the proof it tries to extract the corrupted parties using vk. It outputs a set of indices I that identifies the corrupted parties and the corresponding proof π. If it fails to extract the corrupted parties, it ouputs $\perp$.

We call an input $(\mathsf{sh}_{i_1}, \dots, \mathsf{sh}_{i_{t-f}})$ to the reconstruction box R *consistent* if R contains f shares $(\mathsf{sh}_{i_{t-f+1}}, \dots, \mathsf{sh}_{i_t})$ such that $(\mathsf{sh}_{i_1}, \dots, \mathsf{sh}_{i_t})$ are pairwise distinct, $\mathsf{Rec}(\mathsf{sh}_{i_1}, \dots, \mathsf{sh}_{i_t})$ outputs a valid secret and the distribution of $(\mathsf{sh}_{i_1}, \dots, \mathsf{sh}_{i_{t-f}})$ is indistinguishable from the distribution of $t-f$ shares output by Share. We require a semi-publicly traceable threshold secret sharing scheme to satisfy the following properties:

Perfect Correctness: For any $T \subseteq [n]$ with $|T| = t$ and any secret $s \in \mathcal{S}$, it holds that

$$\Pr[\mathsf{Rec}(\mathsf{Share}(1^\lambda, n, t, s)_T) = s] = 1,$$

where the probability is taken over the random coins of Share.

GspTrace$_{\mathcal{A},\mathsf{SPTSS},\epsilon}(\lambda)$

1. $\mathcal{A}(1^\lambda, n, t)$ outputs (I, state), where $I \subset [n]$ is the set of parties to corrupt and $|I| < t$.
2. Secret s is chosen uniformly at random from $\mathcal{S}$.
3. $\mathsf{Share}(1^\lambda, n, t, s)$ outputs $(\mathsf{sh}_1, \dots, \mathsf{sh}_n, \mathsf{vk})$.
4. On input all shares of parties in I, $\mathcal{A}(\mathsf{state}, \mathsf{sh}_{i_1}, \dots, \mathsf{sh}_{i_{|I|}})$ outputs reconstruction box R.
5. $\mathsf{Trace}^R(f)$ outputs π.
6. $\mathcal{A}$ wins if $\mathsf{Verify}(\mathsf{vk}, \pi)$ outputs either $\perp$ or (I', π) for some $I' \neq I$.

Figure 4: The tracing game for semi-public traceable threshold secret sharing SPTSS.

GspNon-Imputability$_{\mathcal{A},\mathsf{SPTSS}}(\lambda)$

1. $\mathcal{A}(1^\lambda, n, t)$ outputs (i^*, s, state).
2. $\mathsf{Share}(1^\lambda, n, t, s)$ outputs $(\mathsf{sh}_1, \dots, \mathsf{sh}_n, \mathsf{vk})$.
3. On input all shares except for the i^*-th one, $\mathcal{A}(\mathsf{state}, \mathsf{sh}_1, \dots, \mathsf{sh}_{i^*-1}, \mathsf{sh}_{i^*+1}, \dots, \mathsf{sh}_{i_n})$ outputs π.
4. $\mathcal{A}$ wins if $\mathsf{Verify}(\mathsf{vk}, \pi)$ outputs (I^*, π) and $i^* \in I^*$.

Figure 5: The non-imputability game for semi-public traceable threshold secret sharing SPTSS.

$\varepsilon-$**Privacy:** For any $T^* \subseteq [n]$ with $|T^*| < t$, any unbounded adversary $\mathcal{A}$ and a uniformly random secret $s \leftarrow \mathcal{S}$, it holds that

$$\Pr[\mathcal{A}(\mathsf{Share}(1^\lambda, n, t, s)_{T^*}) = s] \leq \varepsilon,$$

where the probability is taken over the random coins of Share and $\mathcal{A}$. If $\varepsilon = 1/|\mathcal{S}|$, we call the scheme *perfectly private*.

Traceability: For any unbounded adversary $\mathcal{A}$, the probability that it wins the game **GspTrace**$_{\mathcal{A},\mathsf{SPTSS},\epsilon}(\lambda)$ defined in Figure 4 is negligible in λ.

Non-Imputability: For any unbounded adversary $\mathcal{A}$, the probability that it wins **GspNon-Imputability**$_{\mathcal{A},\mathsf{SPTSS}}(\lambda)$ defined in Figure 5 is negligible in λ.

4.2 Semi-publicly traceable Mignotte Secret Sharing

The scheme is presented is Figure 6. The sharing algorithm is similar to the one in MTSS, except that the dealer does not construct a tracing key and the verification key contains the identifiers of the parties together with the p_i given to the respective party. Then we adapt the tracing algorithm as follows: The tracer tries to solve the system in Equation (5) similarly to MTSS to obtain the element y that is divisible by all corrupted p_i. It repeats this until it finds $\lceil \lambda/\epsilon \rceil$ many y's and then outputs them as the proof. The $\lceil \lambda/\epsilon \rceil$ repetitions are necessary to handle the event that the reconstruction box outputs elements that do not correspond to any secret but make the tracer succeed in solving the system in Equation 5. Since the verifier knows which of the elements in $\mathcal{P}$ were given to parties by the verifier, it just needs to check which of those n elements divide the y's. If it finds one y that is divisible by the correct number of p_i it outputs those p_i together with the identifiers of the corrupted parties.

$\mathsf{Share}(1^\lambda, n, t, s, H, \mathcal{P})$:

1. For all $i \in [n]$ do:
   (a) Sample $p_i \leftarrow \mathcal{P}$ uniformly at random.
   (b) Set $s_i = s \mod p_i$ and $\mathsf{sh}_i = (s_i, p_i)$.
   (c) Set $\mathsf{vk}_i = (i, p_i)$.
2. Set $\mathsf{vk} = (\mathsf{tk}_1, \dots, \mathsf{tk}_n)$.
3. Output $(\mathsf{sh}_1, \dots, \mathsf{sh}_n, \mathsf{vk})$.

$\mathsf{Rec}(\mathsf{sh}_{i_1}, \dots, \mathsf{sh}_{i_t}, \alpha, \beta)$:

1. Try to solve the following system of equations using the Chinese Remainder Theorem:

$$\begin{cases} X = s_{i_1} \mod p_{i_1} \\ \vdots \\ X = s_{i_t} \mod p_{i_t}. \end{cases}$$

   If it is not possible, output $\perp$. Otherwise, denote the solution of the system by x.
2. If $x \in (\beta, \alpha)$, output x. Otherwise, output $\perp$.

$\mathsf{Trace}^R(f)$:

1. Set $\pi = \emptyset$.
2. Choose $q_1, \dots, q_{t-f} \leftarrow \mathcal{P}$ uniformly at random and independently sample $z_j \leftarrow [0, q_j - 1]$ uniformly at random for all $j \in [t-f]$.
3. Sample $z_1' \leftarrow [0, q_1 - 1] \setminus \{z_1\}$ uniformly at random.
4. Query R on $((z_1, q_1), (z_2, q_2), \dots, (z_{t-f}, q_{t-f}))$ and on $((z_1', q_1), (z_2, q_2), \dots, (z_{t-f}, q_{t-f}))$. Let u and u' be the responses.
5. Try to solve the following system of equations with indeterminates X and Y over $\mathbb{Z}$:

$$\begin{cases} u = Xq_1 + z_1 Y \\ u' = Xq_1 + z_1' Y. \end{cases} \tag{5}$$

   If it is not possible, go to Step 2. Otherwise denote the solution of the system by (x, y).
6. Set $\pi \leftarrow \pi \cup (y, q_1, \dots, q_{t-f})$. If $|\pi| < \lceil \lambda/\epsilon \rceil$, go to Step 2.
7. Output π.

$\mathsf{Verify}(\mathsf{vk}, \pi)$:

1. Parse the first tuple in π as $(y, q_1, \dots, q_{t-f})$.
2. Set $I = \emptyset$ and $y' = y(q_1 \cdot \ldots \cdot q_{t-f})^{-1}$.
3. For all $p_i \in \mathsf{vk}$, check if p_i divides y and if it does, set $I \leftarrow I \cup \{i\}$ and $y' \leftarrow y'/p_i$.
4. Check if $y' < p_n$. If so, output (I, π). Otherwise, remove $(y, q_1, \dots, q_{t-f})$ from π. If $\pi \neq \emptyset$ go to Step 1. Otherwise output $\perp$.

Figure 6: SPMTSS: Semi-Public Traceable Mignotte secret sharing.

Theorem 6. *Let λ be a positive integer and $p_1, \dots, p_{2^\lambda}$ be a $(t, 2^\lambda, \rho)$-Mignotte sequence. For $\mathcal{P} = \{p_1, \dots, p_{2^\kappa}\}$, $\rho \geq 3$ and $t \geq \rho + 1$, we get that* SPMTSS *is a t-out-of-n semi-publicly traceable threshold secret sharing scheme in the random oracle model with the following properties:*

1. *For any adversary $\mathcal{A}$ and some constant $c > 0$, the probability of winning* ***GspTrace***$_{\mathcal{A},\mathsf{SPMTSS},\epsilon}(\lambda)$ *is at most $e^{-(\lambda+\epsilon/c)} + \lceil \lambda/\epsilon \rceil n/2^\lambda$.*

2. Trace *runs in expected time $O(\epsilon^{-2}\lambda)$, where ϵ is the probability that R reconstructs the secret on consistent input.*

3. *For any adversary $\mathcal{A}$, the probability of winning* ***GspNon-Imputability***$_{\mathcal{A},\mathsf{SPMTSS}}(\lambda)$ *is at most $\lceil \lambda/\epsilon \rceil n(t+1)/2^\lambda$.*

Proof. Correctness and $1/(2^\rho - 1)$-privacy of the scheme follow again by correctness and privacy of the original scheme. By Lemma 2 we know that, given a good pair (u, u'), Trace succeeds in solving the system (5) and obtaining a y that is divisible by exactly f corrupted p_j with probability at least $1/4 - n/2^{\kappa+2}$. When (u, u') is not good, the probability that Trace obtains an element y that is divisible by at least one p_i that corresponds to some non-corrupted share is at most $n/2^\lambda$. By Lemma 3 we know that the probability of (u, u') being good is ϵ/c for some constant c. Hence, the probability that π contains no y that was obtained from a good pair (u, u') is at most

$$(1 - \epsilon/c)^{\lceil \lambda/\epsilon \rceil} \leq (e^{-\epsilon/c})^{\lceil \lambda/\epsilon \rceil} \leq e^{-(\lambda+\epsilon)/c}.$$

Adversary $\mathcal{A}$ wins the game **GspTrace**$_{\mathcal{A},\mathsf{SPMTSS},\epsilon}(\lambda)$ if Verify outputs either $\perp$ or (I', π) for some $I' \neq I$. By a union bound and the discussion above we follow that this happens with probability at most $e^{-(\lambda+\epsilon/c)} + \lceil \lambda/\epsilon \rceil n/2^\lambda$.

The second property follows again from Lemma 3, which says that the probability of (u, u') being good is ϵ/c. Since in this case Trace has a constant probability of solving system (5) and because the system needs to be solved $\lceil \lambda/\epsilon \rceil$ many times, the expected running time of Trace is $O(\lambda/\epsilon^2)$.

It remains to prove non-imputability of the scheme. Any $\mathcal{A}$ that wins **GspNon-Imputability**$_{\mathcal{A},\mathsf{SPMTSS}}(\lambda)$ outputs at least one y such that $y < p_n^{t+1}$. Since $\mathcal{P}$ is a Mignotte sequence, we have that at most $t + 1$ elements of $\mathcal{P}$ divide y. The probability that one of the p_i dividing y, which is not corrupted, is part of a share is at most $n/2^\lambda$. Hence, the probability that one of the $t + 1$ elements dividing y is of such form is at most $n(t + 1)/2^\lambda$. The claim follows by taking a union bound over all elements output by trace. □

Remark 3. We can remove the need for the input f to Trace similarly to before as explained in Remark 1 and we can again remove the bound on t as explained in Remark 2.

5 Overview of Traceable Secret Sharing Schemes

We give an overview of all traceable secret sharing schemes in Table 1. The scheme by Goyal, Song and Srinivasan [19] is based on Shamir's secret sharing scheme. It satisfies a public traceability notion but increases the share size by a factor of λ. The schemes by Boneh, Partap and Rotem [6] are based on Shamir's and Blakley's scheme, which encode the secret via polynomials or hyperplanes. Their schemes are efficient since the size of the shares is just twice the share size of the original schemes. The schemes in this work are based on the Chinese remainder theorem and hence allow for weighted access structures.

Our basic scheme MTSS satisfies the traceability notion of Boneh, Partap and Rotem and it is efficient in the sense that the share size only increases by a factor of 2. The tracing algorithm of this scheme is not efficient since it is exponential in $\kappa = \lambda/2$. We therefore obtain a quadratic gap between the running time of the tracing algorithm and the time needed to break security. This is sufficient for most applications, where the tracing algorithm is not run regularly but only in emergencies.

TSS scheme	$\vert\mathsf{sh}\vert$ increase	Trace	public?	any t?	weighted?	any $\mathcal{A}$?
GSS [19]	λ	$\mathsf{poly}(\lambda, \epsilon^{-1})$	yes	yes	no	no
T-Shamir [6]	2	$\mathsf{poly}(\lambda, \epsilon^{-1})$	no	yes	no	no
T-Blakely [6]	2	$\mathsf{poly}(\lambda, \epsilon^{-1})$	no	yes	no	no
MTSS	2	$\mathsf{poly}(2^\kappa, \epsilon^{-1})$	no	no	yes	no
SPMTSS	2	$\mathsf{poly}(\lambda, \epsilon^{-1})$	semi	no	yes	yes
MTSS for small t	$2\lambda/t$	$\mathsf{poly}(2^\kappa, \epsilon^{-1})$	no	yes	yes	no
SPMTSS for small t	$2\lambda/t$	$\mathsf{poly}(\lambda, \epsilon^{-1})$	semi	yes	yes	yes

Table 1: Overview of traceable secret sharing schemes. $|\mathsf{sh}|$ denotes the share size. "any t" indicates if the size of the threshold t is unrestricted. "any $\mathcal{A}$" indicates if security holds against computationally unbounded adversaries.

To obtain a scheme with an efficient tracing algorithm, we move to our semi-public traceability notion. The scheme SPMTSS satisfies this notion and is efficient since the share size only increases by a factor of 2 and the tracing algorithm runs in polynomial time.

In both MTSS and SPMTSS we need that the threshold t is at least of size λ. One can remove this requirement in both schemes at the cost of increasing the share size by a factor of $2\lambda/t$.

Finally, we note that our semi-public schemes are the only schemes that achieve non-imputability against computationally unbounded adversaries.

6 Conclusion and Open Problems

In this work we constructed the first traceable secret sharing schemes based on the Chinese remainder theorem, showing that the traceability notion of Boneh, Partap and Rotem can also be achieved for CRT based schemes. The schemes are efficient for large enough secret sharing threshold t since in this case the share size is only twice the share size of the original scheme. We conclude with two open problems:

First, note that the efficiency of our schemes decreases for very small t, i.e., $t \ll \lambda$ since in this case the share size increases by a multiplicative factor of $2\lambda/t$. One potential way of solving this problem is to construct a traceable version of the Asmuth-Bloom secret sharing scheme. This scheme is very similar to Mignotte's scheme, except that the secret is chosen such that it is smaller than all the p_i and then it is expanded using a secret integer α. In this scheme, given $t-1$ shares, the t-th share can still take on any value in its domain. The probability distribution on this domain is not uniform and hence the scheme is not perfectly private either. However, it has the advantage that every choice of input shares to R would yield a valid secret and hence we would only need to make sure that the distributions are indistinguishable, which might result in a weaker restriction on t. If the reconstruction box R using the Asmuth-Bloom scheme would also output the expanding factor α, one could use basically the same tracing algorithm as the one in our scheme. However, since α is not part of the secret, we cannot expect R to output it and it is not clear how to find out α by only having black box access to R. We leave this question for future work.

The second interesting avenue for future work is public traceability. In this work we achieve a semi-public traceability notion. Recall that we can also turn our first scheme into a publicly traceable scheme if we can impose a timing restriction on the reconstruction box. An interesting open problem is to construct a practical traceable secret sharing scheme that does not require private tracing or verification keys and no timing assumptions.

References

[1] Asmuth, C., Bloom, J.: A modular approach to key safeguarding. IEEE Transactions on Information Theory 29(2), 208–210 (1983) 2

[2] Benaloh, J.C., Leichter, J.: Generalized secret sharing and monotone functions. In: Goldwasser, S. (ed.) Advances in Cryptology – CRYPTO'88. Lecture Notes in Computer Science, vol. 403, pp. 27–35. Springer, Heidelberg, Germany, Santa Barbara, CA, USA (Aug 21–25, 1990) 2

[3] Blakley, G.R.: Safeguarding cryptographic keys. Proceedings of AFIPS 1979 National Computer Conference 48, 313–317 (1979) 1

[4] Boneh, D., Franklin, M.K.: An efficient public key traitor tracing scheme. In: Wiener, M.J. (ed.) Advances in Cryptology – CRYPTO'99. Lecture Notes in Computer Science, vol. 1666, pp. 338–353. Springer, Heidelberg, Germany, Santa Barbara, CA, USA (Aug 15–19, 1999) 6

[5] Boneh, D., Naor, M.: Traitor tracing with constant size ciphertext. In: Ning, P., Syverson, P.F., Jha, S. (eds.) ACM CCS 2008: 15th Conference on Computer and Communications Security. pp. 501–510. ACM Press, Alexandria, Virginia, USA (Oct 27–31, 2008) 6

[6] Boneh, D., Partap, A., Rotem, L.: Traceable secret sharing: Strong security and efficient constructions. In: Advances in Cryptology – CRYPTO 2024: 44th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 18–22, 2024, Proceedings, Part V. p. 221–256. Springer-Verlag, Berlin, Heidelberg (2024), https://doi.org/10.1007/978-3-031-68388-6_9 1, 2, 3, 4, 5, 6, 7, 14, 17, 18

[7] Boneh, D., Sahai, A., Waters, B.: Fully collusion resistant traitor tracing with short ciphertexts and private keys. In: Vaudenay, S. (ed.) Advances in Cryptology – EUROCRYPT 2006. Lecture Notes in Computer Science, vol. 4004, pp. 573–592. Springer, Heidelberg, Germany, St. Petersburg, Russia (May 28 – Jun 1, 2006) 6

[8] Boneh, D., Shaw, J.: Collusion-secure fingerprinting for digital data (extended abstract). In: Coppersmith, D. (ed.) Advances in Cryptology – CRYPTO'95. Lecture Notes in Computer Science, vol. 963, pp. 452–465. Springer, Heidelberg, Germany, Santa Barbara, CA, USA (Aug 27–31, 1995) 6

[9] Boneh, D., Zhandry, M.: Multiparty key exchange, efficient traitor tracing, and more from indistinguishability obfuscation. In: Garay, J.A., Gennaro, R. (eds.) Advances in Cryptology – CRYPTO 2014, Part I. Lecture Notes in Computer Science, vol. 8616, pp. 480–499. Springer, Heidelberg, Germany, Santa Barbara, CA, USA (Aug 17–21, 2014) 6

[10] Chabanne, H., Phan, D.H., Pointcheval, D.: Public traceability in traitor tracing schemes. In: Cramer, R. (ed.) Advances in Cryptology – EUROCRYPT 2005. Lecture Notes in Computer Science, vol. 3494, pp. 542–558. Springer, Heidelberg, Germany, Aarhus, Denmark (May 22–26, 2005) 6

[11] Chen, Y., Vaikuntanathan, V., Waters, B., Wee, H., Wichs, D.: Traitor-tracing from LWE made simple and attribute-based. In: Beimel, A., Dziembowski, S. (eds.) TCC 2018: 16th Theory of Cryptography Conference, Part II. Lecture Notes in Computer Science, vol. 11240, pp. 341–369. Springer, Heidelberg, Germany, Panaji, India (Nov 11–14, 2018) 6

[12] Chor, B., Fiat, A., Naor, M.: Tracing traitors. In: Desmedt, Y. (ed.) Advances in Cryptology – CRYPTO'94. Lecture Notes in Computer Science, vol. 839, pp. 257–270. Springer, Heidelberg, Germany, Santa Barbara, CA, USA (Aug 21–25, 1994) 6

[13] Dodis, Y., Fazio, N.: Public key trace and revoke scheme secure against adaptive chosen ciphertext attack. In: Desmedt, Y. (ed.) PKC 2003: 6th International Workshop on Theory and Practice in Public Key Cryptography. Lecture Notes in Computer Science, vol. 2567, pp. 100–115. Springer, Heidelberg, Germany, Miami, FL, USA (Jan 6–8, 2003) 6

[14] Fiat, A., Tassa, T.: Dynamic traitor training. In: Wiener, M.J. (ed.) Advances in Cryptology – CRYPTO'99. Lecture Notes in Computer Science, vol. 1666, pp. 354–371. Springer, Heidelberg, Germany, Santa Barbara, CA, USA (Aug 15–19, 1999) 6

[15] Garg, S., Jain, A., Mukherjee, P., Sinha, R., Wang, M., Zhang, Y.: Cryptography with weights: MPC, encryption and signatures. In: Advances in Cryptology – CRYPTO 2023, Part I. pp. 295–327. Lecture Notes in Computer Science, Springer, Heidelberg, Germany, Santa Barbara, CA, USA (Aug 2023) 6

[16] Garg, S., Kumarasubramanian, A., Sahai, A., Waters, B.: Building efficient fully collusion-resilient traitor tracing and revocation schemes. In: Al-Shaer, E., Keromytis, A.D., Shmatikov, V. (eds.) ACM CCS 2010: 17th Conference on Computer and Communications Security. pp. 121–130. ACM Press, Chicago, Illinois, USA (Oct 4–8, 2010) 6

[17] Gong, J., Luo, J., Wee, H.: Traitor tracing with $N^{1/3}$-size ciphertexts and $O(1)$-size keys from k-Lin. In: Advances in Cryptology – EUROCRYPT 2023, Part III. pp. 637–668. Lecture Notes in Computer Science, Springer, Heidelberg, Germany (Jun 2023) 6

[18] Goyal, R., Koppula, V., Waters, B.: Collusion resistant traitor tracing from learning with errors. In: Diakonikolas, I., Kempe, D., Henzinger, M. (eds.) 50th Annual ACM Symposium on Theory of Computing. pp. 660–670. ACM Press, Los Angeles, CA, USA (Jun 25–29, 2018) 6

[19] Goyal, V., Song, Y., Srinivasan, A.: Traceable secret sharing and applications. In: Malkin, T., Peikert, C. (eds.) Advances in Cryptology – CRYPTO 2021, Part III. Lecture Notes in Computer Science, vol. 12827, pp. 718–747. Springer, Heidelberg, Germany, Virtual Event (Aug 16–20, 2021) 1, 17, 18

[20] Iftene, S.: General secret sharing based on the chinese remainder theorem with applications in e-voting. In: Dima, C., Minea, M., Tiplea, F.L. (eds.) Proceedings of the First Workshop in Information and Computer Security, ICS@SYNASC 2006, Timisoara, Romania, September 30, 2006. Electronic Notes in Theoretical Computer Science, vol. 186, pp. 67–84. Elsevier (2006), https://doi.org/10.1016/j.entcs.2007.01.065 2

[21] Kiayias, A., Yung, M.: Self protecting pirates and black-box traitor tracing. In: Kilian, J. (ed.) Advances in Cryptology – CRYPTO 2001. Lecture Notes in Computer Science, vol. 2139, pp. 63–79. Springer, Heidelberg, Germany, Santa Barbara, CA, USA (Aug 19–23, 2001) 6

[22] Kiayias, A., Yung, M.: Traitor tracing with constant transmission rate. In: Knudsen, L.R. (ed.) Advances in Cryptology – EUROCRYPT 2002. Lecture Notes in Computer Science, vol. 2332, pp. 450–465. Springer, Heidelberg, Germany, Amsterdam, The Netherlands (Apr 28 – May 2, 2002) 6

[23] Kranakis, E.: Primality and cryptography. John Wiley & Sons, Inc., USA (1986) 8

[24] Kurosawa, K., Desmedt, Y.: Optimum traitor tracing and asymmetric schemes. In: Nyberg, K. (ed.) Advances in Cryptology – EUROCRYPT'98. Lecture Notes in Computer Science, vol. 1403, pp. 145–157. Springer, Heidelberg, Germany, Espoo, Finland (May 31 – Jun 4, 1998) 6

[25] Mignotte, M.: How to share a secret? In: Beth, T. (ed.) Advances in Cryptology – EUROCRYPT'82. Lecture Notes in Computer Science, vol. 149, pp. 371–375. Springer, Heidelberg, Germany, Burg Feuerstein, Germany (Mar 29 – Apr 2, 1983) 2

[26] Naor, D., Naor, M., Lotspiech, J.: Revocation and tracing schemes for stateless receivers. In: Kilian, J. (ed.) Advances in Cryptology – CRYPTO 2001. Lecture Notes in Computer Science, vol. 2139, pp. 41–62. Springer, Heidelberg, Germany, Santa Barbara, CA, USA (Aug 19–23, 2001) 6

[27] Naor, M., Pinkas, B.: Threshold traitor tracing. In: Krawczyk, H. (ed.) Advances in Cryptology – CRYPTO'98. Lecture Notes in Computer Science, vol. 1462, pp. 502–517. Springer, Heidelberg, Germany, Santa Barbara, CA, USA (Aug 23–27, 1998) 6

[28] Ning, Y., Miao, F., Huang, W., Meng, K., Xiong, Y., Wang, X.: Constructing ideal secret sharing schemes based on chinese remainder theorem. In: Peyrin, T., Galbraith, S. (eds.) Advances in Cryptology – ASIACRYPT 2018, Part III. Lecture Notes in Computer Science, vol. 11274, pp. 310–331. Springer, Heidelberg, Germany, Brisbane, Queensland, Australia (Dec 2–6, 2018) 6

[29] Safavi-Naini, R., Wang, Y.: Sequential traitor tracing. In: Bellare, M. (ed.) Advances in Cryptology – CRYPTO 2000. Lecture Notes in Computer Science, vol. 1880, pp. 316–332. Springer, Heidelberg, Germany, Santa Barbara, CA, USA (Aug 20–24, 2000) 6

[30] Shamir, A.: How to share a secret. Commun. ACM 22(11), 612–613 (nov 1979), `https://doi.org/10.1145/359168.359176` 1

[31] Wee, H.: Functional encryption for quadratic functions from k-lin, revisited. In: Pass, R., Pietrzak, K. (eds.) TCC 2020: 18th Theory of Cryptography Conference, Part I. Lecture Notes in Computer Science, vol. 12550, pp. 210–228. Springer, Heidelberg, Germany, Durham, NC, USA (Nov 16–19, 2020) 6

[32] Zhandry, M.: New techniques for traitor tracing: Size $N^{1/3}$ and more from pairings. In: Micciancio, D., Ristenpart, T. (eds.) Advances in Cryptology – CRYPTO 2020, Part I. Lecture Notes in Computer Science, vol. 12170, pp. 652–682. Springer, Heidelberg, Germany, Santa Barbara, CA, USA (Aug 17–21, 2020) 6

[33] Zou, X., Maino, F., Bertino, E., Sui, Y., Wang, K., Li, F.: A new approach to weighted multi-secret sharing. In: 2011 Proceedings of 20th International Conference on Computer Communications and Networks (ICCCN). pp. 1–6 (2011) 6