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. Boneh, Partap and Rotem give constructions from Shamir's secret sharing and Blakely'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 by a factor of 2.

1 Introduction

Threshold secret sharing, introduced by Shamir [Sha79] and Blakely [Bla79], allows a dealer to split a secret s into n shares $\mathsf{sh}_1, \dots, \mathsf{sh}_n$ 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 [GSS21] 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 [GSS21] 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 [BPR24] propose a new definition of traceable secret sharing, which allows them to give the first practical constructions from Shamir's secret sharing and Blakely'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 Blakely'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 Blakely'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 [Mig83] and the Asmuth-Bloom secret sharing scheme [AB83]. 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 satisfies *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 exceed the threshold [Ift06]. Shamir's and Blakely'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 [BL90]: 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 [Ift06]. 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 $i \in \{1, 2, 3, 4\}$.

1.1 Our Contribution

In this work we present the first traceable secret sharing scheme based on the Chinese remainder theorem: a traceable version of Mignotte's secret sharing scheme. We prove its security in the random oracle model. The scheme is efficient since the size of the shares of the traceable scheme is only twice the size of the shares in the original secret sharing scheme. This is because the sharing algorithm of our scheme is almost the same as in the original Mignotte 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.

The sharing algorithm of the 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 [BPR24], 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 in 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 Bezout'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 Bezout coefficients of $p_1 \cdot \ldots \cdot p_{t-1}$ and p_t. A careful analysis of this relation yields that the following system of equations over $\mathbb{Z}$ with indeterminates X and Y is solvable with at least constant probability:

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

Denote the solution of the system by (x, y). We will show that the corrupted $p_1, \ldots, p_{t-1}$ always divide x 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 x, 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 x. 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.

We note that the tracing algorithm is a procedure that is only used rarely and so we do not need to optimize its efficiency. As long as its running time is feasible, it serves its purpose of deterring parties from leaking their shares.

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 $p_1, \ldots, p_n$. 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.

Making the scheme publicly traceable. The traceability notion of Boneh, Partap and Rotem [BPR24] 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 [BPR24] 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 [BPR24], the tracing key consists of the values $F(x_1), \ldots, F(x_n)$, 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.

There are two ways to turn our construction into a publicly verifiable scheme. The first one 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 [BPR24] publicly traceable since here the box only needs to perform one function evaluation to check if the input is consistent with the tracing key.

The second possibility is to remove the tracing key altogether and only let the dealer construct a private verification key containing the pairs (i, p_i) in the clear and give it to a trusted authority. Then the tracer can send all the p_i it has found to the trusted authority and the trusted authority checks which p_i are part of real shares. In this case the sequence $\mathcal{P}$ only needs to be large enough such that the probability of guessing f identifiers p_i that belong to real shares is negligible. This would also eliminate the need for a random oracle. In the traceable version of Shamir's scheme in [BPR24] this is not as straightforward because the tracer uses the tracing key to find a certain polynomial in a large set of polynomials. If one wanted to remove the tracing key, the tracer would need to send this large set to the trusted authority and outsource a significant amount of work to it.

1.2 Other Related Work

Traitor-tracing schemes. Traitor tracing for broadcast encryption schemes, introduced by Chor, Fiat and Naor [CFN94], allow a tracer to trace back leaked decryption keys. The techniques used in the long line of traitor tracing schemes [BS95, KD98, NP98, BF99, FT99, SW00, KY01, NNL01, KY02, DF03, CPP05, BSW06, BN08, GKSW10, BZ14, GKW18, CVW+18, Zha20, Wee20, GLW23], however, are different from the one used in [BPR24] and in this work. See [BPR24] for a more detailed description of the techniques.

Weighted CRT based secret sharing schemes. Zuo et al. [ZMB+11] extend CRT based secret sharing schemes to allow for weighted multi-secret sharing. Garg et al. [GJM+23] 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. [NMH+18] extended CRT based secret sharing over $\mathbb{Z}_N$ to polynomial rings over finite fields.

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 (Bezout'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 *Bezout 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 Bezout 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 $X = v_1 bp_2 + v_2 ap_1$.

2.2 (Traceable) Threshold Secret Sharing

We mostly follow the definition in [BPR24]. 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 efficient 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.

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

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 TTS.

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

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 TTS.

We call an input $(\mathsf{sh}_{i_1}, \ldots, \mathsf{sh}_{i_{t-f}})$ to the reconstruction box R *good* 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

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

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

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

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 **GTrace**$_{\mathcal{A},\mathsf{TTS},\epsilon,\delta}(\lambda)$ 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 **GNon-Imputability**$_{\mathcal{A},\mathsf{TTS}}(\lambda)$ 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 $s_i := S \mod p_i$.
- 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 $s < \alpha < p_{i_1} \cdots p_{i_t}$. 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 [Kra86, page 9].

Lemma 1. *For any integers $2 \leq t \leq n$, 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
$$\alpha = p_1 \cdots p_t \geq P_\ell^{(t^2-1)/t} > P_\ell^{t-1} \geq p_{n-t+2} \cdot \ldots \cdot p_n = \beta.$$
Further, we get that
$$\frac{\alpha-\beta}{\beta} \geq \frac{p_1^t}{p_n^{t-1}} - 1 \geq \frac{P_\ell^{(t^2-1)/t}}{P_\ell^{t-1}} - 1 = \frac{P_\ell}{P_\ell^{1/t}} - 1.$$
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 MTTS 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.

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 $\mathcal{P} = \{p_1, \ldots, p_{2^\kappa}\}$, $\rho \geq 3$ and $t \geq \rho + 1$, we get that MTTS 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 $\boldsymbol{GTrace}_{\mathcal{A},\mathsf{MTTS},\epsilon}(\lambda)$ is at most $n/2^\kappa \cdot 1/(2^\rho - 1)$.*

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

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, outputs $\perp$. 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 $I, \pi = \emptyset$.
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 $j \in [2, t-f]$.
3. Sample $z_1 \leftarrow [1, q_1 - 1]$ and $z_1' \leftarrow [1, q_1 - 1] \setminus \{z_1\}$ 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 $\mathbb{Z}$:

$$\begin{cases} u = X + z_1 q_1 Y \\ u' = X + z_1' q_1 Y. \end{cases} \tag{1}$$

   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 x. 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 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 π.
7. If $|I| = f$, output I and π. 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: MTTS: Traceable Mignotte secret sharing when f shares are corrupted.

2. 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.*

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

Proof. Correctness and $1/(2^{\rho}-1)$-privacy of the scheme follows by correctness and privacy of the original scheme. We begin with proving the first property. The proof of traceability consists of three steps:

I We first show that the system (1) has a unique solution with probability at least $1/2$ whenever the pair (u, u') is *good*, i.e., whenever both u and u' correspond to the output of the reconstruction algorithm given a good input.

II We show that in this case all corrupted p_j do divide x and with all but at most $n/2^{\kappa}$ probability none of the not corrupted p_j divides x. Hence, whenever (u, u') is good, we have that Trace finds exactly f corrupted shares and terminates with probability at least $1/2 - n/2^{\kappa+1}$.

III We 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.

Afterwards we compute the probability of (u, u') being good, which determines the expected running time of Trace. We begin with Step I. Let $p_{i_1}, \ldots, p_{i_f}$ denote the p_i's corresponding to the corrupted shares. 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 Bezout 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

$$u = a u_0 p_{i_1} \cdots p_{i_{t-1}} + b z_1 q_1 \mod q_1 \prod_{j=1}^{t-1} p_{i_j}. \tag{2}$$

Similarly, we have that

$$u' = a u_0 p_{i_1} \cdots p_{i_{t-1}} + b z_1' q_1 \mod q_1 \prod_{j=1}^{t-1} p_{i_j}. \tag{3}$$

Let $a u_0 = k q_1 + r$ for some $k, r \in \mathbb{Z}$ with $|r| < q_1$. Note that $r \neq 0$ because $u \neq 0 \mod q_1$ by construction of the tracing algorithm. Plugging into (2) and (3) we get

$$u = r p_{i_1} \cdots p_{i_{t-1}} + b z_1 q_1 \mod q_1 \prod_{j=1}^{t-1} p_{i_j} \tag{4}$$

and

$$u' = r p_{i_1} \cdots p_{i_{t-1}} + b z_1' q_1 \mod q_1 \prod_{j=1}^{t-1} p_{i_j}. \tag{5}$$

Now we have that $\left| r p_{i_1} \cdots p_{i_{t-1}} \right| < q_1 \prod_{j=1}^{t-1} p_{i_j}$ and $|b z_1 q_1|, |b z_1' q_1| < 3 q_1 < q_1 \prod_{j=1}^{t-1} p_{i_j}$. The last inequality follows by definition of Mignotte sequences and the fact that $t > 3$. 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 $rp_{i_1}\cdots p_{i_{t-1}}$ and bz_1q_1. It follows that over $\mathbb{Z}$ we have either (case 1)

$$u = rp_{i_1}\cdots p_{i_{t-1}} + bz_1q_1$$

or (case 2)

$$\begin{aligned} u - q_1\prod_{j=1}^{t-1} p_{i_j} &= rp_{i_1}\cdots p_{i_{t-1}} + bz_1q_1 \\ \Leftrightarrow u &= (r+q_1)p_{i_1}\cdots p_{i_{t-1}} + bz_1q_1. \end{aligned}$$

And similarly for u' we have either (case 1)

$$u' = rp_{i_1}\cdots p_{i_{t-1}} + bz_1'q_1$$

or (case 2)

$$\begin{aligned} u' - q_1\prod_{j=1}^{t-1} p_{i_j} &= rp_{i_1}\cdots p_{i_{t-1}} + bz_1'q_1 \\ \Leftrightarrow u' &= (r+q_1)p_{i_1}\cdots p_{i_{t-1}} + bz_1'q_1. \end{aligned}$$

If the first case holds for both u and u', we have that system (1)

$$\begin{cases} u = X + z_1q_1Y \\ u' = X + z_1'q_1Y \end{cases}$$

has the unique solution $(rp_{i_1}\cdots p_{i_{t-1}}, b)$. Similarly, if the second case holds for both u and u', the above system has the unique solution $((r+q_1)p_{i_1}\cdots p_{i_{t-1}}, b)$. If u and u' are in different cases, the system is not solvable. In the worst case we have that for exactly half of the possible choices for share q_1 case 1 holds and for the other half case 2 holds, which means that the probability that the system is solvable for a good pair (u,u') is at least $1/2$.

We continue with Step II of the proof of traceability. If (u,u') is good and the system is solvable, then either $x = rp_{i_1}\cdots p_{i_{t-1}}$ or $x = (r+q_1)p_{i_1}\cdots p_{i_{t-1}}$, where $|r| < q_1$. It is obvious that all of the corrupted p_i divide x in both cases. Since $|r| < q_1$ and the elements of $\mathcal{P}$ form a Mignotte sequence, we have that at most one more element $p \in \mathcal{P}$ can divide x. The probability that this 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 (1) is solvable.

Finally, in Step III, we conclude the proof of traceability with the following observation: 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 $\mathbf{GTrace}_{\mathcal{A},\mathsf{TTS},\epsilon,\delta}(\lambda)$ 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.

We now compute the probability of (u,u') being good to determine the running time of Trace, i.e., prove the second property stated in the theorem. 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 $\mathbf{GTrace}_{\mathcal{A},\mathsf{TTS},\epsilon,\delta}(\lambda)$, 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

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

From this and the proof of traceability, we follow that the probability that Trace in one fixed round is at least

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

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 $O(\epsilon^{-1} \cdot f \cdot 2^\kappa)$

It remains to prove the third property. The scheme is non-imputable in the random oracle model by the following observation: Any adversary $\mathcal{A}$ that wins the game $\mathbf{GNon\text{-}Imputability}_{\mathcal{A},\mathsf{TTS}}(\lambda)$ 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 $1/2^{2\kappa}$. $\square$

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.

References

[AB83] C. Asmuth and J. Bloom. A modular approach to key safeguarding. *IEEE Transactions on Information Theory*, 29(2):208–210, 1983. 2

[BF99] Dan Boneh and Matthew K. Franklin. An efficient public key traitor tracing scheme. In Michael J. Wiener, editor, *Advances in Cryptology – CRYPTO'99*, volume 1666 of *Lecture Notes in Computer Science*, pages 338–353, Santa Barbara, CA, USA, August 15–19, 1999. Springer, Heidelberg, Germany. 4

[BL90] Josh Cohen Benaloh and Jerry Leichter. Generalized secret sharing and monotone functions. In Shafi Goldwasser, editor, *Advances in Cryptology – CRYPTO'88*, volume 403 of *Lecture Notes in Computer Science*, pages 27–35, Santa Barbara, CA, USA, August 21–25, 1990. Springer, Heidelberg, Germany. 2

[Bla79] G. R. Blakley. Safeguarding cryptographic keys. *Proceedings of AFIPS 1979 National Computer Conference*, 48:313–317, 1979. 1

[BN08] Dan Boneh and Moni Naor. Traitor tracing with constant size ciphertext. In Peng Ning, Paul F. Syverson, and Somesh Jha, editors, *ACM CCS 2008: 15th Conference on Computer and Communications Security*, pages 501–510, Alexandria, Virginia, USA, October 27–31, 2008. ACM Press. 4

[BPR24] Dan Boneh, Aditi Partap, and Lior Rotem. Traceable secret sharing: Strong security and efficient constructions. Cryptology ePrint Archive, Paper 2024/405, 2024. https://eprint.iacr.org/2024/405 (to appear at CRYPTO'24). 1, 3, 4, 5

[BS95] Dan Boneh and James Shaw. Collusion-secure fingerprinting for digital data (extended abstract). In Don Coppersmith, editor, *Advances in Cryptology – CRYPTO'95*, volume 963 of *Lecture Notes in Computer Science*, pages 452–465, Santa Barbara, CA, USA, August 27–31, 1995. Springer, Heidelberg, Germany. 4

[BSW06] Dan Boneh, Amit Sahai, and Brent Waters. Fully collusion resistant traitor tracing with short ciphertexts and private keys. In Serge Vaudenay, editor, *Advances in Cryptology – EUROCRYPT 2006*, volume 4004 of *Lecture Notes in Computer Science*, pages 573–592, St. Petersburg, Russia, May 28 – June 1, 2006. Springer, Heidelberg, Germany. 4

[BZ14] Dan Boneh and Mark Zhandry. Multiparty key exchange, efficient traitor tracing, and more from indistinguishability obfuscation. In Juan A. Garay and Rosario Gennaro, editors, *Advances in Cryptology – CRYPTO 2014, Part I*, volume 8616 of *Lecture Notes in Computer Science*, pages 480–499, Santa Barbara, CA, USA, August 17–21, 2014. Springer, Heidelberg, Germany. 4

[CFN94] Benny Chor, Amos Fiat, and Moni Naor. Tracing traitors. In Yvo Desmedt, editor, *Advances in Cryptology – CRYPTO'94*, volume 839 of *Lecture Notes in Computer Science*, pages 257–270, Santa Barbara, CA, USA, August 21–25, 1994. Springer, Heidelberg, Germany. 4

[CPP05] Hervé Chabanne, Duong Hieu Phan, and David Pointcheval. Public traceability in traitor tracing schemes. In Ronald Cramer, editor, *Advances in Cryptology – EUROCRYPT 2005*, volume 3494 of *Lecture Notes in Computer Science*, pages 542–558, Aarhus, Denmark, May 22–26, 2005. Springer, Heidelberg, Germany. 4

[CVW+18] Yilei Chen, Vinod Vaikuntanathan, Brent Waters, Hoeteck Wee, and Daniel Wichs. Traitor-tracing from LWE made simple and attribute-based. In Amos Beimel and Stefan Dziembowski, editors, *TCC 2018: 16th Theory of Cryptography Conference, Part II*, volume 11240 of *Lecture Notes in Computer Science*, pages 341–369, Panaji, India, November 11–14, 2018. Springer, Heidelberg, Germany. 4

[DF03] Yevgeniy Dodis and Nelly Fazio. Public key trace and revoke scheme secure against adaptive chosen ciphertext attack. In Yvo Desmedt, editor, *PKC 2003: 6th International Workshop on Theory and Practice in Public Key Cryptography*, volume 2567 of *Lecture Notes in Computer Science*, pages 100–115, Miami, FL, USA, January 6–8, 2003. Springer, Heidelberg, Germany. 4

[FT99] Amos Fiat and Tamir Tassa. Dynamic traitor training. In Michael J. Wiener, editor, *Advances in Cryptology – CRYPTO'99*, volume 1666 of *Lecture Notes in Computer Science*, pages 354–371, Santa Barbara, CA, USA, August 15–19, 1999. Springer, Heidelberg, Germany. 4

[GJM+23] Sanjam Garg, Abhishek Jain, Pratyay Mukherjee, Rohit Sinha, Mingyuan Wang, and Yinuo Zhang. Cryptography with weights: MPC, encryption and signatures. In *Advances in Cryptology – CRYPTO 2023, Part I*, Lecture Notes in Computer Science, pages 295–327, Santa Barbara, CA, USA, August 2023. Springer, Heidelberg, Germany. 4

[GKSW10] Sanjam Garg, Abishek Kumarasubramanian, Amit Sahai, and Brent Waters. Building efficient fully collusion-resilient traitor tracing and revocation schemes. In Ehab Al-Shaer, Angelos D. Keromytis, and Vitaly Shmatikov, editors, *ACM CCS 2010: 17th Conference on Computer and Communications Security*, pages 121–130, Chicago, Illinois, USA, October 4–8, 2010. ACM Press. 4

[GKW18] Rishab Goyal, Venkata Koppula, and Brent Waters. Collusion resistant traitor tracing from learning with errors. In Ilias Diakonikolas, David Kempe, and Monika Henzinger, editors, *50th Annual ACM Symposium on Theory of Computing*, pages 660–670, Los Angeles, CA, USA, June 25–29, 2018. ACM Press. 4

[GLW23] Junqing Gong, Ji Luo, and Hoeteck Wee. Traitor tracing with $N^{1/3}$-size ciphertexts and $O(1)$-size keys from k-Lin. In *Advances in Cryptology – EUROCRYPT 2023, Part III*, Lecture Notes in Computer Science, pages 637–668. Springer, Heidelberg, Germany, June 2023. 4

[GSS21] Vipul Goyal, Yifan Song, and Akshayaram Srinivasan. Traceable secret sharing and applications. In Tal Malkin and Chris Peikert, editors, *Advances in Cryptology – CRYPTO 2021, Part III*, volume 12827 of *Lecture Notes in Computer Science*, pages 718–747, Virtual Event, August 16–20, 2021. Springer, Heidelberg, Germany. 1

[Ift06] Sorin Iftene. General secret sharing based on the chinese remainder theorem with applications in e-voting. In Catalin Dima, Marius Minea, and Ferucio Laurentiu Tiplea, editors, *Proceedings of the First Workshop in Information and Computer Security, ICS@SYNASC 2006, Timisoara, Romania, September 30, 2006*, volume 186 of *Electronic Notes in Theoretical Computer Science*, pages 67–84. Elsevier, 2006. 2

[KD98] Kaoru Kurosawa and Yvo Desmedt. Optimum traitor tracing and asymmetric schemes. In Kaisa Nyberg, editor, *Advances in Cryptology – EUROCRYPT'98*, volume 1403 of *Lecture Notes in Computer Science*, pages 145–157, Espoo, Finland, May 31 – June 4, 1998. Springer, Heidelberg, Germany. 4

[Kra86] Evangelos Kranakis. *Primality and cryptography.* John Wiley & Sons, Inc., USA, 1986. 7

[KY01] Aggelos Kiayias and Moti Yung. Self protecting pirates and black-box traitor tracing. In Joe Kilian, editor, *Advances in Cryptology – CRYPTO 2001*, volume 2139 of *Lecture Notes in Computer Science*, pages 63–79, Santa Barbara, CA, USA, August 19–23, 2001. Springer, Heidelberg, Germany. 4

[KY02] Aggelos Kiayias and Moti Yung. Traitor tracing with constant transmission rate. In Lars R. Knudsen, editor, *Advances in Cryptology – EUROCRYPT 2002*, volume 2332 of *Lecture Notes in Computer Science*, pages 450–465, Amsterdam, The Netherlands, April 28 – May 2, 2002. Springer, Heidelberg, Germany. 4

[Mig83] Maurice Mignotte. How to share a secret? In Thomas Beth, editor, *Advances in Cryptology – EUROCRYPT'82*, volume 149 of *Lecture Notes in Computer Science*, pages 371–375, Burg Feuerstein, Germany, March 29 – April 2, 1983. Springer, Heidelberg, Germany. 2

[NMH+18] Yu Ning, Fuyou Miao, Wenchao Huang, Keju Meng, Yan Xiong, and Xingfu Wang. Constructing ideal secret sharing schemes based on chinese remainder theorem. In Thomas Peyrin and Steven Galbraith, editors, *Advances in Cryptology – ASIACRYPT 2018, Part III*, volume 11274 of *Lecture Notes in Computer Science*, pages 310–331, Brisbane, Queensland, Australia, December 2–6, 2018. Springer, Heidelberg, Germany. 5

[NNL01] Dalit Naor, Moni Naor, and Jeffery Lotspiech. Revocation and tracing schemes for stateless receivers. In Joe Kilian, editor, *Advances in Cryptology – CRYPTO 2001*, volume 2139 of *Lecture Notes in Computer Science*, pages 41–62, Santa Barbara, CA, USA, August 19–23, 2001. Springer, Heidelberg, Germany. 4

[NP98] Moni Naor and Benny Pinkas. Threshold traitor tracing. In Hugo Krawczyk, editor, *Advances in Cryptology – CRYPTO'98*, volume 1462 of *Lecture Notes in Computer Science*, pages 502–517, Santa Barbara, CA, USA, August 23–27, 1998. Springer, Heidelberg, Germany. 4

[Sha79] Adi Shamir. How to share a secret. *Commun. ACM*, 22(11):612–613, nov 1979. 1

[SW00] Reihaneh Safavi-Naini and Yejing Wang. Sequential traitor tracing. In Mihir Bellare, editor, *Advances in Cryptology – CRYPTO 2000*, volume 1880 of *Lecture Notes in Computer Science*, pages 316–332, Santa Barbara, CA, USA, August 20–24, 2000. Springer, Heidelberg, Germany. 4

[Wee20] Hoeteck Wee. Functional encryption for quadratic functions from k-lin, revisited. In Rafael Pass and Krzysztof Pietrzak, editors, *TCC 2020: 18th Theory of Cryptography Conference, Part I*, volume 12550 of *Lecture Notes in Computer Science*, pages 210–228, Durham, NC, USA, November 16–19, 2020. Springer, Heidelberg, Germany. 4

[Zha20] Mark Zhandry. New techniques for traitor tracing: Size $N^{1/3}$ and more from pairings. In Daniele Micciancio and Thomas Ristenpart, editors, *Advances in Cryptology – CRYPTO 2020, Part I*, volume 12170 of *Lecture Notes in Computer Science*, pages 652–682, Santa Barbara, CA, USA, August 17–21, 2020. Springer, Heidelberg, Germany. 4

[ZMB+11] Xukai Zou, Fabio Maino, Elisa Bertino, Yan Sui, Kai Wang, and Feng Li. A new approach to weighted multi-secret sharing. In *2011 Proceedings of 20th International Conference on Computer Communications and Networks (ICCCN)*, pages 1–6, 2011. 4