



# Multiparty Cardinality Testing for Threshold Private Intersection

Pedro Branco<sup>1(✉)</sup>, Nico Döttling<sup>2</sup>, and Sihang Pu<sup>2</sup>

<sup>1</sup> IT, IST - University of Lisbon, Lisbon, Portugal

<sup>2</sup> Helmholtz Center for Information Security (CISPA), Saarbrücken, Germany

**Abstract.** Threshold Private Set Intersection (PSI) allows multiple parties to compute the intersection of their input sets if and only if the intersection is larger than  $n - t$ , where  $n$  is the size of each set and  $t$  is some threshold. The main appeal of this primitive is that, in contrast to standard PSI, known upper-bounds on the communication complexity only depend on the threshold  $t$  and not on the sizes of the input sets. Current threshold PSI protocols split themselves into two components: A Cardinality Testing phase, where parties decide if the intersection is larger than some threshold; and a PSI phase, where the intersection is computed. The main source of inefficiency of threshold PSI is the former part.

In this work, we present a new Cardinality Testing protocol that allows  $N$  parties to check if the intersection of their input sets is larger than  $n - t$ . The protocol incurs in  $\tilde{O}(Nt^2)$  communication complexity. We thus obtain a Threshold PSI scheme for  $N$  parties with communication complexity  $\tilde{O}(Nt^2)$ .

## 1 Introduction

Suppose Alice holds a set  $S_A$  and Bob a set  $S_B$ . Private set intersection (PSI) is a cryptographic primitive that allows each party to learn the intersection  $S_A \cap S_B$  and nothing else. In particular, Alice gets no information about  $S_B \setminus S_A$  (and vice-versa). The problem has attracted a lot of attention through the years, with an extended line of work proposing solutions in a variety of different settings (e.g., [11–13, 15–17, 21, 25–27, 31–36]). Also, numerous applications have been proposed for PSI such as contact discovery, advertising, etc. (see for example [22] and references therein). More recently, PSI has also been proposed as a solution for private contact tracing (e.g., [2]).

*Threshold PSI.* In this work, we focus on a special setting of PSI called *Threshold PSI*. Here, the parties involved in the protocol learn the output if the size of the intersection between the input sets of the parties is very large, say larger than  $n - t$ , where  $n$  is the size of the input sets and  $t$  is some *threshold* such that  $t \ll n$ ; Otherwise, they learn nothing about the intersection. This is in contrast

with standard PSI where the parties always get the intersection, no matter its size.

The main reason for considering this problem (apart from its numerous applications which we discuss next) is that the amount of communication needed is much smaller than for standard PSI: In particular, there are threshold PSI protocols whose communication complexity depends only on the threshold  $t$  and not on the size of the input sets as for standard PSI [17].

Despite its theoretical and practical appeal, there are just a few works that consider this problem [16, 17, 20], and just one of them achieves communication complexity independent of  $n$  [17], in the two party setting.

## 1.1 Applications of Threshold PSI

A wide number of applications has been suggested for threshold PSI in previous works such as applications to dating apps or biometric authentication mechanisms [17].

One of the most interesting applications for threshold PSI is its use in carpooling (or ridesharing) apps. Suppose two (or more) parties are using a carpooling app, which allows them to share a vehicle if their routes have a large intersection. However, due to privacy issues, they do not want to make their itinerary public. Threshold PSI solves this problem in a simple way [20]: The parties can engage in a threshold PSI protocol, learn the intersection of the routes and, if the intersection is large enough, share a vehicle. Otherwise, they learn nothing and their privacy is maintained.

*PSI Using Threshold PSI.* Most of current protocols for threshold PSI (including ours) are splitted into two parts: i) a *Cardinality Testing*, where parties decide if the intersection is larger than  $n - t$ ; and ii) secure computation of the intersection of the input sets (which we refer to as the PSI part). The communication complexity of these two parts should depend only on the threshold  $t$  and not on the input sets' size  $n$ .

Threshold PSI protocols of this form can be used to efficiently compute the intersection, even when no threshold on the intersection is known a priori by the parties, by doing an exponential search for the *right* threshold. In this case, parties can proceed as follows:

1. Run a Cardinality Testing for some  $t$  (say  $t = 1$ ).
2. If it succeeds, perform the PSI part. Else, run again the Cardinality Test for  $t = 2t$ .
3. Repeat Step 2 until the Cardinality Testing succeeds for some threshold  $t$  and the set intersection is computed.

By following this blueprint, parties are sure that they overshoot the right threshold by a factor of at most 2. That is, if the intersection is larger than  $n - t'$ , then the Cardinality Testing will succeed for  $t$  such that  $t \geq t' > t/2$ . Thus, they can compute the intersection incurring only in a factor of 2 overhead

over the best insecure protocol. In other words, PSI protocols can be computed with communication complexity depending on the size of the intersection, and not on the size of the sets.

This approach can be useful in scenarios where parties suspect that the intersection is large but they do not know exactly how large it is.

## 1.2 Our Contributions

In the following,  $N$  denotes the number of parties in a multi-party protocol and  $t$  is the threshold in a threshold PSI protocol. Below, we briefly describe our results.

*Multi-party Cardinality Testing.* We develop a new Cardinality Testing scheme that allows  $N$  parties to check if the intersection of their input sets, each having size  $n$ , is larger than  $n - t$  for some threshold  $t \ll n$ . The protocol needs  $\tilde{O}(Nt^2)$  bits of information to be exchanged.

Along the way, we develop new protocols to securely compute linear algebra related functions (such as compute the rank of an encrypted matrix, invert a encrypted matrix or even solve an encrypted linear system). Our protocols build on ideas of previous works [24, 29], except that our protocols are specially crafted for the multi-party case. Technically, we rely heavily on Threshold Public-Key Encryption schemes which are additively homomorphic (such schemes can be constructed from DDH [14], DCR [30], or from several pairings assumptions [3, 4]) to perform linear operations.

*Multi-party Threshold PSI.* We then show how our Cardinality Testing protocol can be used to build a Threshold PSI protocol in the multi-party setting. Our construction achieves communication complexity of  $\tilde{O}(Nt^2)$ .

**Concurrent Work.** Recently, Ghosh and Simkin [18] updated their paper with a generalization to the multi-party case which is similar to the one presented in this paper in Sect. 4. However, they leave as a major open problem the design of a new Cardinality Testing that extends nicely to multiple parties, a problem on which we make relevant advances in this work.

In a concurrent work, Badrinarayanan *et al.* [1] also proposed new protocols for threshold PSI in the multi-party setting. Their results complement ours. In particular, they propose an FHE-based approach to solve the same problem as we do with a communication complexity of  $\mathcal{O}(Nt)$ , where  $N$  is the number of parties and  $t$  is the threshold. However, we remark that the goal of our work was to reduce the assumptions needed for threshold PSI. They also propose a TPKE-based protocol that solves a slightly different problem: the parties learn the intersection if and only if the set difference among the sets is large, that is,  $|\left(\bigcup_{i=1}^N S_i\right) \setminus \left(\bigcap_{i=1}^N S_i\right)|$  is large<sup>1</sup>, which is denoted as  $\mathcal{F}_{\text{TPSI-int}}$  in [1]. This

<sup>1</sup> It is a slightly different problem from the one we solve in this work. Here, we want to disclose the intersection  $\bigcap_{i=1}^N S_i$  if  $|\bigcap_{i=1}^N S_i| \geq n - t$ , which is denoted as  $\mathcal{F}_{\text{TPSI-diff}}$  in [1].

protocol achieves communication complexity of  $\tilde{\mathcal{O}}(Nt)$ . They achieve that result using completely different techniques from ones used in this work. Namely, they noticed that computing the determinant of a Hankel matrix can be done in sublinear time in the size of the matrix. This implies that the cardinality testing of [17] can actually be realized in time  $\tilde{\mathcal{O}}(Nt)$ .

### 1.3 Technical Outline

We now give a high-level overview of the techniques we use to achieve the results discussed above.

**Threshold PSI: The Protocol of [17].** Consider two parties Alice and Bob, with their respective input sets  $S_A$  and  $S_B$  of size  $n$ . Suppose that they want to know the intersection  $S_A \cap S_B$  iff  $|S_A \cap S_B| \geq n - t$  for some threshold  $t \ll n$ . To compute the intersection, both parties encode their sets into polynomials  $P_A(x) = \prod_i^n (x - a_i)$  and  $P_B(x) = \prod_i^n (x - b_i)$  over a large finite field  $\mathbb{F}$ , where  $a_i \in S_A$  and  $b_i \in S_B$ . The main observation of Ghosh and Simkin [17] is that *set reconciliation techniques* (developed by Minsky et al. [28]) can be applied in this scenario: if  $|S_A \cap S_B| \geq n - t$ , then

$$\frac{P_A(x)}{P_B(x)} = \frac{P_{A \cap B}(x) P_{A \setminus B}(x)}{P_{A \cap B}(x) P_{B \setminus A}(x)} = \frac{P_{A \setminus B}(x)}{P_{B \setminus A}(x)}$$

and, moreover,  $\deg P_{A \setminus B} = \deg P_{B \setminus A} = t$ . Hence, Alice and Bob just need to (securely) compute  $\mathcal{O}(t)$  evaluation points of the rational function  $P_A(x)/P_B(x) = P_{A \setminus B}(x)/P_{B \setminus A}(x)$  and, after interpolating over these points, Bob can recover the denominator (which reveals the intersection).

Of course, Bob should not be able to recover the numerator  $P_{A \setminus B}$ , otherwise security is compromised. So, [17] used an Oblivious Linear Evaluation (OLE) scheme to *mask* the numerator with a random polynomial that hides  $P_{A \setminus B}$  from Bob.

This protocol is only secure if Alice and Bob are absolutely sure that  $|S_A \cap S_B| \geq n - t$ . Otherwise, additional information could be leaked about the respective inputs. Consequently, Alice and Bob should perform a *Cardinality Testing* protocol, which reveals if  $|S_A \cap S_B| \geq n - t$  and nothing else.

*Limitations of the Protocol when Extending to the Multi-party Setting.* It turns out that the main source of inefficiency when extending Ghosh and Simkin protocol to the multi-party setting is the Cardinality Testing they use. In [17], Alice and Bob encode their sets into polynomials  $Q_A(X) = \sum_i^n x^{a_i}$  and  $Q_B(X) = \sum_i^n x^{b_i}$ , respectively, where  $a_i \in S_A$  and  $b_i \in S_B$ . Then, they can check if  $\tilde{Q}(x) = Q_A(x) - Q_B(x)$  is a *sparse* polynomial. If it is, we conclude that the set  $(S_A \cup S_B) \setminus (S_A \cap S_B)$  is small. By disposing  $\mathcal{O}(t)$  evaluations of the polynomial  $\tilde{Q}(x)$  in a Hankel matrix [19] and securely computing its determinant (via a generic secure linear algebra protocol from [24]), both parties can

determine if  $|S_A \cap S_B| \geq n - t$ . The total communication complexity of this protocol is  $\mathcal{O}(t^2)$ .<sup>2</sup>

However, if we were to *naively* extend this approach to the multi-party setting, we would have  $N$  parties computing, say,

$$\tilde{Q}(x) = NQ_1(x) - Q_2(x) - \dots - Q_N(x)$$

which is a sparse polynomial only if  $N$  is small. Moreover, if we were to compute the sparsity of this polynomial using the same approach, we would have a protocol with communication complexity  $\mathcal{O}((Nt)^2)$ .

**Our Approach.** Given the state of affairs presented in the previous section, it seems we need to take a different approach from the one of [17] if we want to design an efficient threshold PSI protocol for multiple parties.

*Interlude: Secure Linear Algebra.* Recall that in the setting of *secure linear algebra* (as in [29] and [24]), there are two parties, one holding an encryption of a matrix  $\text{Enc}(\text{pk}, \mathbf{M})$  and the other one holding the corresponding secret key  $\text{sk}$ . Their goal is to compute an encryption of a (linear algebra related) function of the matrix  $\mathbf{M}$ , such as the rank, the determinant of  $\mathbf{M}$ , or, most importantly, find a solution  $\mathbf{x}$  for the linear system  $\mathbf{M}\mathbf{x} = \mathbf{y}$  where both  $\mathbf{M}$  and  $\mathbf{y}$  are encrypted. We can easily extend this problem to the multi-party case: Consider  $N$  parties,  $P_1, \dots, P_N$ , each one holding a share of the secret key of a threshold PKE scheme. Additionally,  $P_1$  has an encrypted matrix. The goal of all the parties is to compute an encryption of a (linear algebra related) function of the encrypted matrix.

We observe that the protocols for secure linear algebra presented in [24] can be extended to the multiparty setting by replacing the use of an (additively homomorphic) PKE and garbled circuits for an (additively homomorphic) threshold PKE<sup>3</sup>. Hence, our protocols allow  $N$  parties to solve a linear system of the form  $\mathbf{M}\mathbf{x} = \mathbf{y}$  under the hood of a threshold PKE scheme.

*Cardinality Testing via Degree Test of a Rational Function.* Consider again the encodings  $P_{S_i}(x) = \prod_j^n (x - a_j^{(i)})$  where  $a_j^{(i)} \in S_i$ , for  $N$  different sets, and the rational function<sup>4</sup>

$$\frac{P_{S_1} + \dots + P_{S_N}}{P_{S_1}} = \frac{P_{S_1 \setminus (\cap_{j=1}^N S_j)} + \dots + P_{S_N \setminus (\cap_{j=1}^N S_j)}}{P_{S_1 \setminus (\cap_{j=1}^N S_j)}}.$$

<sup>2</sup> Given this, we conclude that the communication complexity of the threshold PSI protocol of [17] is dominated by this Cardinality Testing protocol.

<sup>3</sup> We need a bit-conversion protocol such as [37] to convert between binary circuits and algebra operations.

<sup>4</sup> We actually need to randomize the polynomials in the numerator to guarantee correctness, that is, we need to multiply each term in the numerator by a uniformly chosen element. This is in contrast with the two-party setting where correctness holds even without randomizing the numerator. However, we omit this step for simplicity.

Note that, if the intersection  $\cap S_i$  is larger than  $n - t$ , then  $\deg P_{S_1 \setminus (\cap_{j=1}^N S_j)} = \dots = \deg P_{S_N \setminus (\cap_{j=1}^N S_j)} \leq t$ .

Therefore, the Cardinality Testing boils down to the following problem: Given a rational function  $f(x) = \tilde{P}_1(x)/\tilde{P}_2(x)$ , can we securely decide if  $\deg \tilde{P}_1 = \deg \tilde{P}_2 \leq t$  having access to  $\mathcal{O}(t)$  evaluation points of  $f(x)$ ?

Our crucial observation is that, if we interpolate two different rational functions  $f_V$  and  $f_W$  on different two support sets  $V = \{v_i, f(v_i)\}$  and  $W = \{w_i, f(w_i)\}$  each one of size  $2t$ , then we have:

1.  $f_V = f_W$  if  $\deg P_1 = \deg P_2 \leq t$
2.  $f_V \neq f_W$  if  $\deg P_1 = \deg P_2 > t$

except with negligible probability over the uniform choice of  $v_i, w_i$ .

Moreover, interpolating a rational function can be reduced to solving a linear system of equations. Hence, by using the Secure Linear Algebra tools developed before, we can perform the *degree test* revealing nothing else than the output. In other words, we can decide if the size of the intersection is smaller than  $n - t$  while revealing no additional information about the parties' input sets.

*Security of the Protocol.* We prove security of our Cardinality Testing in the UC framework [7]. However, there is a subtle issue in our security proof. Namely, our secure linear algebra protocols cannot be proven UC-secure since the inputs are encrypted under a public key which, in the UC setting, needs to come from somewhere.

We solve this problem by using the Externalized UC framework [8]. In this framework, the secure linear algebra ideal functionalities all share a common setup which, in our case, is the public key (and the corresponding secret key shares). We prove security of our secure linear algebra protocols in this setting.

Since the secure linear algebra protocols are secure if they all share the same public key, then, on the Cardinality Testing, we just need to create this public key and share it over these functionalities. Thus, we prove standard UC-security of our Cardinality Testing.

Badrinarayanan et al. [1] also encounter the same problem as we did and they opted to not prove security of each subprotocol individually, but rather prove security only for their main protocol (where the public key is created and shared among these smaller protocols).

*Multi-party PSI.* Having developed a Cardinality Testing, we can now focus on securely computing the intersection. In fact, our protocol for computing the intersection can be seen as a *generalization* of Gosh and Simkin protocol [17]. Again, by encoding the sets as above (that is,  $P_{S_i}(x) = \prod_j^n (x - a_j^{(i)})$  where  $a_j^{(i)} \in S_j$  and  $S_j$  is the set of party  $P_j$ ) and knowing that the intersection is larger than  $n - t$ , parties can securely compute the rational function<sup>5</sup>  $(P_{S_1} + \dots + P_{S_N})/P_{S_1}$ .

<sup>5</sup> Again, we omit the randomization of the polynomials. Actually, without randomization, these methods (including [17]) are exactly the same as the technique for set reconciliation problem in [28].

By interpolating the rational function on any  $\mathcal{O}(t)$  points, party  $P_1$  can recover the denominator and compute the intersection.

The main difference between our protocol and the one in [17] is that we replace the OLE calls used in [17] by a threshold additively homomorphic PKE scheme (which can be seen as the multi-party replacement of OLE).

## 1.4 Other Related Work

*Oblivious Linear Algebra.* Cramer and Damgård [9] proposed a constant-round protocol to securely solve a linear system of unknown rank over a finite field. Since they were mainly focused on round-optimality, the communication cost of their proposal is  $\Omega(t^3)$  for  $\mathcal{O}(t^2)$  input size. Bouman et al. [5] recently constructed a secure linear algebra protocol for multiple parties, however they focused on computational complexity.

Other secure linear algebra schemes in the two-party setting were presented by Nissim and Weinreb in [29] and Kiltz et al. in [24]. In the following, consider (square) matrices of size  $t$  over a field  $\mathbb{F}$ . These two works take different approaches: [29] obviously solves linear algebra related problems directly via Gaussian elimination in  $\mathcal{O}(t^2)$  communication complexity, for a square matrix of size  $t$ . However, their approach has an error probability that decreases polynomially with  $t$ . In other words, the error probability is only sufficiently small when applied to linear system with large matrices. Whereas [24] has error probability decreases polynomially with  $|\mathbb{F}|$ , which is negligible when  $\mathbb{F}$  is of exponentially size.<sup>6</sup>

## 2 Preliminaries

If  $S$  is a finite set, then  $x \leftarrow_s S$  denotes an element  $x$  sampled from  $S$  according to a uniform distribution and  $|S|$  denotes the cardinality of  $S$ . If  $\mathcal{A}$  is an algorithm,  $y \leftarrow \mathcal{A}(x)$  denotes the output  $y$  after running  $\mathcal{A}$  on input  $x$ . For  $N \in \mathbb{N}$ , we define  $[N] = \{1, \dots, N\}$ .

Given two distributions  $D_1, D_2$ , we say that they are computationally indistinguishable, denoted as  $D_1 \approx D_2$ , if no probabilistic polynomial-time (PPT) algorithm is able to distinguish them.

Throughout this work, we denote the security parameter by  $\lambda$ .

---

<sup>6</sup> This is important to us since, in the thresholds PSI setting,  $t \ll n$  where  $t$  is the threshold and  $n$  is the set size. Kiltz et al. solve linear algebra problems via minimal polynomials, and use adaptors between garbled circuits and additive homomorphic encryption to reduce round complexity. In this work, we extend Kiltz's protocol to the multiparty case without using garbled circuits (otherwise the circuit size would depend on number of parties) while preserving the same communication complexity for each party ( $\mathcal{O}(t^2)$ ).

## 2.1 Threshold Public-Key Encryption

We present some ideal functionalities regarding threshold public-key encryption (TPKE) schemes. In the following,  $N$  is the number of parties.

Let  $\mathcal{F}_{\text{Gen}}$  be the ideal functionality that distributes a secret share of the secret key and the corresponding public key. That is, on input  $(\text{sid}, P_i)$ ,  $\mathcal{F}_{\text{Gen}}$  outputs  $(\text{pk}, \text{sk}_i)$  to each party  $P_i$  where  $(\text{pk}, \text{sk}_1, \dots, \text{sk}_N) \leftarrow \text{TPKE.Gen}(1^\lambda, N)$ .

Moreover, we define the functionality  $\mathcal{F}_{\text{DecZero}}$ , which allows  $N$  parties, each of them holding a secret share  $\text{sk}_i$ , to learn if a ciphertext is an encryption of 0 and nothing else. That is,  $\mathcal{F}_{\text{DecZero}}$  receives as input a ciphertext  $c$  and the secret shares of each of the parties. It outputs 0, if  $0 \leftarrow \text{Dec}(\text{sk}_1, \dots, \text{Dec}(\text{sk}_N, c) \dots)$ , and 1 otherwise. Note that these functionalities can be securely realized on various PKE schemes such as El Gamal PKE or Pailler<sup>7</sup> PKE [21].

We also assume that the underlying TPKE (or plain PKE) is always additively homomorphic, unless stated otherwise (see Supplementary Material A.1).

## 2.2 UC Framework and Ideal Functionalities

In this work, we use the UC framework by Canetti [7] to analyze the security of our protocols.<sup>8</sup> Throughout this work, we only consider semi-honest adversaries, unless stated otherwise. We denote the underlying environment by  $\mathcal{Z}$ . For a protocol  $\pi$  and a real-world adversary  $\mathcal{A}$ , we denote the real-world ensemble by  $\text{EXEC}_{\pi, \mathcal{A}, \mathcal{Z}}$ . Similarly, for an ideal functionality  $\mathcal{F}$  and a simulator  $\text{Sim}$ , we denote the ideal-world ensemble by  $\text{IDEAL}_{\mathcal{F}, \text{Sim}, \mathcal{Z}}$ .

**Definition 1.** *We say that a protocol  $\pi$  UC-realizes  $\mathcal{F}$  if for every PPT adversary  $\mathcal{A}$  there is a PPT simulator  $\text{Sim}$  such that for all PPT environments  $\mathcal{Z}$ ,*

$$\text{IDEAL}_{\mathcal{F}, \text{Sim}, \mathcal{Z}} \approx \text{EXEC}_{\pi, \mathcal{A}, \mathcal{Z}}$$

where  $\mathcal{F}$  is an ideal functionality.

In the following, we present some ideal functionalities that will be recurrent for the rest of the paper.

*Multi-party Threshold Private Set Intersection.* This ideal functionality implements the multi-party version of the functionality above. Here, each of the  $N$  parties input a set and they learn the intersection if and only if the intersection is large enough.

<sup>7</sup> We will assume the message space of Paillier’s cryptosystem as a field as also mentioned in [24].

<sup>8</sup> We refer the reader to [7] for a detailed explanation of the framework.



**$\mathcal{F}_{\text{MTPSI}}$  functionality**

*Parameters:*  $\text{sid}, N, t \in \mathbb{N}$  known to both parties.

- Upon receiving  $(\text{sid}, P_i, S_i)$  from party  $P_i$ ,  $\mathcal{F}_{\text{MTPSI}}$  stores  $S_i$  and ignores future messages from  $P_i$  with the same  $\text{sid}$ .
- Once  $\mathcal{F}_{\text{MTPSI}}$  has stored all inputs  $S_i$ , for  $i \in [n]$ , it does the following: If  $|S_1 \setminus (\cap_{i=2}^N S_i)| \leq t$ ,  $\mathcal{F}_{\text{MTPSI}}$  outputs  $S_\cap = \cap_{i=1}^N S_i$ . Else, it outputs  $\perp$ .

**Externalized UC Protocol with Global Setup.** We introduce a notion of protocol emulation from [8], called externalized UC emulation (EUC), which is a simplified version of UC with global setup (GUC).

**Definition 2** (EUC-Emulation [8]). *We say that  $\pi$  EUC-realizes  $\mathcal{F}$  with respect to shared functionality  $\bar{\mathcal{G}}$  (or, in shorthand, that  $\pi$   $\bar{\mathcal{G}}$ -EUC-emulates  $\phi$ ) if for any PPT adversary  $\mathcal{A}$  there exists a PPT adversary  $\text{Sim}$  such that for any shared functionality  $\bar{\mathcal{G}}$ , we have:*

$$\text{IDEAL}_{\mathcal{F}, \text{Sim}, \mathcal{Z}}^{\bar{\mathcal{G}}} \approx \text{EXEC}_{\pi, \mathcal{A}, \mathcal{Z}}^{\bar{\mathcal{G}}}$$

Notice that the formalism implies that the shared functionality  $\bar{\mathcal{G}}$  exists both in the model for executing  $\pi$  and also in the model for executing the ideal protocol for  $\mathcal{F}$ ,  $\text{IDEAL}_{\mathcal{F}}$ .

We remark that the notion of  $\bar{\mathcal{G}}$ -EUC-emulation can be naturally extended to protocols that use several different shared functionalities (instead of only one).

### 2.3 Polynomials and Interpolation

We present a series of results that will be useful to analyze correctness and security of the protocols presented in this work.

The following lemma show how we can mask a polynomial of degree less than  $t$  using a uniformly random polynomial.

**Lemma 1** ([25]). *Let  $\mathbb{F}_p$  be a prime order field,  $P(x), Q(x)$  be two polynomials over  $\mathbb{F}_p$  such that  $\deg P = \deg Q = d \leq t$  and  $\gcd(P, Q) = 1$ . Let  $R_1, R_2 \xleftarrow{\$} \mathbb{F}_p$  such that  $\deg R_1 = \deg R_2 = t$ . Then  $U(x) = P(x)R_1(x) + Q(x)R_2(x)$  is a uniformly random polynomial with  $\deg U \leq 2t$ .*

Note that this result also applies for multiple polynomials as long as they don't share a common factor (referring to Theorem 2 and Theorem 3 of [25] for more details).

We say that  $f$  is a rational function if  $f(x) = \frac{P(x)}{Q(x)}$  for two polynomials  $P$  and  $Q$ .

The next two lemmata show that we can recover a rational function via interpolation and that this function is unique.

**Lemma 2** ([28]). *Let  $f(x) = P(x)/Q(x)$  be rational function where  $\deg P(x) = m$  and  $\deg Q(x) = n$ . Then  $f(x)$  can be uniquely recovered (up to constants) via interpolation from  $m + n + 1$  points. In particular, if  $P(x)$  and  $Q(x)$  are monic,  $f(x)$  can be uniquely recovered from  $m + n$  points.*

**Lemma 3** ([28]). *Choose  $V$  to be a support set<sup>9</sup> of cardinality  $m_1 + m_2 + 1$ . Then, there is a unique rational function  $f(x) = P(x)/Q(x)$  that can be interpolated from  $V$ , and  $P(x)$  has degree at most  $m_1$  and  $Q(x)$  has degree at most  $m_2$ .*

### 3 Oblivious Degree Test for Rational Functions

Suppose we have a rational function  $f(x) = P(x)/Q(x)$  where  $P(x)$  and  $Q(x)$  are two polynomials with the same degree. In this section, we present a protocol that allows several parties to check if  $\deg P(x) = \deg Q(x) \leq t$  for some threshold  $t \in \mathcal{Z}$ . To this end, and inspired by the works of [24, 29], we present a multi-party protocol to obliviously solve a linear system  $\mathbf{M}\mathbf{x} = \mathbf{y}$  over a finite field  $\mathbb{F}$  with communication complexity  $O(t^2 k \lambda N)$ , where  $\mathbf{M} \in \mathbb{F}^{t \times t}$ ,  $\log |\mathbb{F}| = k$  and  $N$  is the number of parties involved in the protocol.

#### 3.1 Oblivious Linear Algebra

In this section, we state the Secure Linear Algebra protocols that we need to build our degree test protocol. For the sake of brevity, the protocols are presented in Appendix B. These protocols all have the following form: There is a public key of a TPKE that encrypts a matrix  $\mathbf{M}$  and every party involved in the protocol has a share of the secret key.

Note that if we let parties  $P_i$  input their encrypted matrix  $\text{Enc}(\mathbf{M})$ , then the ideal functionality  $\mathcal{F}$  has to know the secret key (by receiving secret key shares from all parties), otherwise  $\mathcal{F}$  cannot compute the corresponding function correctly. However, this will cause an unexpected problem in security proof as mentioned in our introduction and [1]: The environment  $\mathcal{Z}$  will learn the secret key as well since it can choose inputs for all parties. We fix this by relying on global UC framework where exists a shared functionality  $\bar{\mathcal{G}}$  in charge of distributing key pairs ( $\mathcal{F}_{\text{Gen}}$  from Sect. 2.1).

**Oblivious Matrix Multiplication.** We begin by presenting the ideal functionality for a multi-party protocol to jointly compute the product of two matrices, under a TPKE. The protocol is presented in Appendix B.1.

---

<sup>9</sup> A support set is a set of pairs  $(x, y)$ .

*Ideal Functionality.* The ideal functionality for oblivious matrix multiplication is presented below.

$\mathcal{F}_{\text{OMM}}$ functionality
<p><i>Parameters:</i> <math>\text{sid}, N, q, t \in \mathbb{N}</math> and <math>\mathbb{F}</math>, where <math>\mathbb{F}</math> is a field of order <math>q</math>, known to the <math>N</math> parties involved in the protocol.</p>
<p><i>Global Setup:</i> <math>\text{pk}</math> public-key of a threshold PKE scheme and <math>\text{sk}_i</math> distributed to each party <math>P_i</math> via <math>\mathcal{F}_{\text{Gen}}</math>.</p>
<ul style="list-style-type: none"> <li>– Upon receiving <math>(\text{sid}, P_1, \text{Enc}(\text{pk}, \mathbf{M}_l), \text{Enc}(\text{pk}, \mathbf{M}_r))</math> from party <math>P_1</math> (where <math>\mathbf{M}_l, \mathbf{M}_r \in \mathbb{F}^{t \times t}</math>), <math>\mathcal{F}_{\text{OMM}}</math> outputs <math>\text{Enc}(\text{pk}, \mathbf{M}_l \cdot \mathbf{M}_r)</math> to <math>P_1</math> and <math>(\text{Enc}(\text{pk}, \mathbf{M}_l), \text{Enc}(\text{pk}, \mathbf{M}_r), \text{Enc}(\text{pk}, \mathbf{M}_l \cdot \mathbf{M}_r))</math> to all other parties <math>P_i</math>, for <math>i = 2, \dots, N</math>.</li> </ul>

**Securely Compute the Rank of a Matrix.** We present the ideal functionality to obliviously compute the rank of an encrypted matrix. The protocol is presented in Appendix B.2.

*Ideal Functionality.* The ideal functionality of oblivious rank computation is defined below.

$\mathcal{F}_{\text{ORank}}$ functionality
<p><i>Parameters:</i> <math>\text{sid}, N, q, t \in \mathbb{N}</math> and <math>\mathbb{F}</math>, where <math>\mathbb{F}</math> is a field of order <math>q</math>, known to the <math>N</math> parties involved in the protocol.</p>
<p><i>Global Setup:</i> <math>\text{pk}</math> public-key of a threshold PKE scheme and <math>\text{sk}_i</math> distributed to each party <math>P_i</math> via <math>\mathcal{F}_{\text{Gen}}</math>.</p>
<ul style="list-style-type: none"> <li>– Upon receiving <math>(\text{sid}, P_1, \text{Enc}(\text{pk}, \mathbf{M}))</math> from party <math>P_1</math> (where <math>\mathbf{M} \in \mathbb{F}^{t \times t}</math>), <math>\mathcal{F}_{\text{ORank}}</math> outputs <math>\text{Enc}(\text{pk}, \text{rank}(\mathbf{M}))</math> to <math>P_1</math> and <math>(\text{Enc}(\text{pk}, \mathbf{M}), \text{Enc}(\text{pk}, \text{rank}(\mathbf{M})))</math> to all other parties <math>P_i</math>, for <math>i = 2, \dots, N</math>.</li> </ul>

**Oblivious Linear System Solver.** We now show how  $N$  parties can securely solve a linear system using the multiplication protocol above. We follow the ideas from [24] to reduce the problem to minimal polynomials, and the only difference is we focus on multiparty setting.

The protocol is presented in Appendix B.5. Informally, we evaluate an arithmetic circuit following the ideas of [10], and for the unary representation, a binary-conversion protocol [37] is required. All of above protocols can be based on Paillier cryptosystem.

*Ideal Functionality.* We give an ideal functionality of oblivious linear system solver for multiparty as follows.

**$\mathcal{F}_{\text{OLS}}$  functionality**

*Parameters:*  $\text{sid}, N, q, t \in \mathbb{N}$  and  $\mathbb{F}$ , where  $\mathbb{F}$  is a field of order  $q$ , known to the  $N$  parties involved in the protocol.  $\text{pk}$  public-key of a threshold PKE scheme.

*Global Setup:*  $\text{pk}$  public-key of a threshold PKE scheme and  $\text{sk}_i$  distributed to each party  $P_i$  via  $\mathcal{F}_{\text{Gen}}$ .

- Upon receiving  $(\text{sid}, P_1, \text{Enc}(\text{pk}, \mathbf{M}), \text{Enc}(\text{pk}, \mathbf{y}))$  from party  $P_1$  (assuming there is a solution  $\mathbf{x}$  for  $\mathbf{M}\mathbf{x} = \mathbf{y}$ ),  $\mathcal{F}_{\text{OLS}}$  outputs  $\text{Enc}(\text{pk}, \mathbf{x})$  such that  $\mathbf{M}\mathbf{x} = \mathbf{y}$ .

### 3.2 Oblivious Degree Test

We now present the main protocol of this section and the one that will be using in the construction of threshold PSI. Given a rational function  $P(x)/Q(x)$  (for two polynomials  $P(x)$  and  $Q(x)$  with the same degree) and two support sets  $V_1, V_2$ , the protocol allows us to test if the degree of the polynomials is less than some threshold  $t$ . Of course, we can do this using generic approaches like garbled circuits. However, we are interested in solutions with communication complexity depending on  $t$  (even when the degree of  $P(x)$  or  $Q(x)$  is much larger than  $t$ ).

*Ideal Functionality.* The ideal functionality for degree test of rational functions is presented below.

**$\mathcal{F}_{\text{SDT}}$  functionality**

*Parameters:*  $\text{sid}, N, q, n, t \in \mathbb{N}$ ,  $\mathbb{F}$  is a field of order  $q$  and  $t$  is a pre-defined threshold, known to the  $N$  parties involved in the protocol.  $\text{pk}$  public-key of a threshold PKE scheme.  $\alpha_1, \dots, \alpha_{4t+2} \xleftarrow{\$} \mathbb{F}$  known to the  $N$  parties.

*Global Setup:*  $\text{pk}$  public-key of a threshold PKE scheme and  $\text{sk}_i$  distributed to each party  $P_i$  via  $\mathcal{F}_{\text{Gen}}$ .

- Upon receiving  $(\text{sid}, P_1, \text{Enc}(\text{pk}, f_1), \dots, \text{Enc}(\text{pk}, f_{4t+2}))$  from party  $P_1$  (where  $f_i = P_1(\alpha_i)/P_2(\alpha_i)$ , and  $P_1, P_2$  are two co-prime polynomials with same degree  $t'$  (additionally,  $P_2$  is monic),  $\mathcal{F}_{\text{SDT}}$  outputs 0 if  $t' \leq t$ ; otherwise it outputs 1.

*Protocol.* We present the Protocol 1 for secure degree test which we denote by secDT. The main idea of the protocol is to interpolate the rational function on two different support sets and check if the result is the same in both experiments.

Recall that interpolating a rational function boils down to solve a linear equation. We can thus use the secure linear algebra tools developed to allow the parties to securely solve a linear equation.

Also recall that two rational functions  $C_v^{(1)}/C_v^{(2)} = C_w^{(1)}/C_w^{(2)}$  are equivalent if  $C_v^{(1)}C_w^{(2)} - C_w^{(1)}C_v^{(2)} = 0$ . Thus, in the end, parties just need to securely check if  $C_v^{(1)}C_w^{(2)} - C_w^{(1)}C_v^{(2)}$  is equal to 0.

*Comments.* Suppose that, for an interpolation point  $\alpha_i$ , the rational function  $f(x) = P(x)/Q(x)$  is well-defined but  $Q(\alpha_i) = P(\alpha_i) = 0$  such that we cannot compute  $f(\alpha_i)$  by division. In this case<sup>12</sup>, the parties evaluate  $\tilde{P}(x) = P(x)/(x - \alpha_i)$  and  $\tilde{Q}(x) = Q(x)/(x - \alpha_i)$  on  $\alpha_i$  and set  $f(\alpha_i) = \tilde{P}(\alpha_i)/\tilde{Q}(\alpha_i)$ . These points are called *tagged values* and this strategy is used in [28]. In more details, instead of using  $\text{Enc}(\text{pk}, f_i)$  for  $\alpha_i$ , we will use a tagged pair  $(\text{Enc}(\text{pk}, s_i^{(1)}), \text{Enc}(\text{pk}, s_i^{(2)}))$  where  $s_i^{(1)} = \frac{P_1(\alpha_i)}{x - \alpha_i}$  and  $s_i^{(2)} = \frac{P_2(\alpha_i)}{x - \alpha_i}$ . Correspondingly, replace each row of  $\text{Enc}(\text{pk}, \mathbf{M}_r)$  and  $\text{Enc}(\text{pk}, \mathbf{y}_r)$  with

$$\text{Enc}\left(\text{pk}, \left[ s_i^{(2)} r_i^t \dots s_i^{(2)} - s_i^{(1)} r_i^{t-1} \dots - s_i^{(1)} \right]\right)$$

and  $\text{Enc}\left(\text{pk}, \left[ s_i^{(1)} r_i^t \right]\right)$ , respectively.

Also, note that the protocol easily generalizes to rational functions  $f(x) = P(x)/Q(x)$  with  $\deg P \neq \deg Q$  (which is actually what we use in the following sections). We present the version where  $\deg P = \deg Q$  for simplicity. In fact, the case where  $\deg P \neq \deg Q$  can be reduced to the presented case by multiplying the least degree polynomial by a uniformly chosen  $R(x)$  of degree  $\max\{\deg P(x) - \deg Q(x), \deg Q(x) - \deg P(x)\}$ .

Moreover, if  $t' > t$ , the linear system for rational interpolation might be unsolvable. In this case, there is no solution which means we cannot interpolate an appropriate rational function on certain support set. Therefore, the parties just return 0.

*Analysis.* We analyze correctness, security and communication complexity of the protocol. We begin the analysis with the following auxiliary lemma.

**Lemma 4.** *Let  $\mathbb{F}$  be a field with  $|\mathbb{F}| = \omega(2^{\log \lambda})$ . Let  $V = \{(v_i, f(v_i)) | \forall i \in [1, 2t + 1]\}$  and  $W = \{(w_i, f(w_i)) | \forall i \in [1, 2t + 1]\}$  be two support sets each of*

<sup>10</sup> Note that this is the linear system that we need to solve in order to perform rational interpolation [28].

<sup>11</sup> The polynomial multiplication can be expressed as matrix multiplication.

<sup>12</sup> In the case that only  $Q(\alpha_i) = 0$ , use a different tagged pair  $(\text{Enc}(\text{pk}, s_i^{(1)}), \text{Enc}(\text{pk}, 0))$ , and this can be noticed by the party who owns polynomial  $Q(x)$ . In our PSI setting, it is party  $P_1$ .

---

**Protocol 1** Secure Degree Test **secDT**


---

**Setup:** Each party has a secret key share  $sk_i$  for a public key  $pk$  of a TPKE  $TPKE = (\text{Gen}, \text{Enc}, \text{Dec})$ . The parties have access to the ideal functionalities  $\mathcal{F}_{\text{ORank}}$ ,  $\mathcal{F}_{\text{OLS}}$ ,  $\mathcal{F}_{\text{OMM}}$  and  $\mathcal{F}_{\text{DecZero}}$ . The values  $\{\alpha_1, \dots, \alpha_{4t+2}\} \leftarrow_{\$} \mathbb{F}^{4t+2}$  are public, from which also sampling a random point  $\alpha^* \leftarrow_{\$} \{\alpha_1, \dots, \alpha_{4t+2}\}$ .

**Input:** Party  $P_1$  inputs  $\{(\alpha_1, \text{Enc}(pk, f_1)), \dots, (\alpha_{4t+2}, \text{Enc}(pk, f_{4t+2}))\}$ , where  $f_i = \frac{P_1(\alpha_i)}{P_2(\alpha_i)}$ , where  $P_1(x), P_2(x)$  are two polynomials with degree  $\deg(P_1) = \deg(P_2) = t' = \text{poly}(\log |\mathbb{F}|)$  and such that  $P_2(\alpha_i) \neq 0$  for all  $i \in [2t]$ .

- 1:  $P_1$  sets  $\{(\alpha_j, \text{Enc}(pk, f_j))\}_{j \in [2t+1]} = \{(v_j, \text{Enc}(pk, f_{v,j}))\}_{j \in [2t+1]}$ , and  $\{(\alpha_j, \text{Enc}(pk, f_j))\}_{j \in \{2t+2, \dots, 4t+2\}} = \{(w_j, \text{Enc}(pk, f_{w,j}))\}_{j \in [2t+1]}$ . It homomorphically generates an encrypted linear system consisting of

$$\text{Enc}(pk, \mathbf{M}_r) = \text{Enc} \left( pk, \begin{bmatrix} r_1^t & \dots & 1 & -f_{r,1} \cdot r_1^{t-1} & \dots & -f_{r,1} \\ \vdots & & \vdots & \vdots & & \vdots \\ r_{2t+1}^t & \dots & 1 & -f_{r,2t+1} \cdot r_{2t+1}^{t-1} & \dots & -f_{r,2t+1} \end{bmatrix} \right)$$

and

$$\text{Enc}(pk, \mathbf{y}_r) = \text{Enc} \left( pk, \begin{bmatrix} f_{r,1} \cdot r_1^t \\ \vdots \\ f_{r,2t+1} \cdot r_{2t+1}^t \end{bmatrix} \right)$$

for  $r = \{v, w\}$ .<sup>10</sup> Here  $\mathbf{M}_r$  is a square matrix with dimension  $2t + 1$  and  $\mathbf{y}_r$  a  $2t + 1$ -sized vector.

- 2: All parties jointly compute  $\text{Enc}(pk, \text{rank}(\mathbf{M}_r) - \text{rank}([\mathbf{M}_r | \mathbf{y}_r])$  for  $r \in \{v, w\}$  through two invocations of  $\mathcal{F}_{\text{ORank}}$  and mutually decrypt the ciphertext via  $\mathcal{F}_{\text{DecZero}}$ . If the result is different from 0, they abort the protocol.
- 3: All parties mutually solve the two linear systems above using  $\mathcal{F}_{\text{OLS}}$  such that each party gets  $\text{Enc} \left( pk, \left( \mathbf{c}_v^{(1)} || \mathbf{c}_v^{(2)} \right) \right)$  and  $\text{Enc} \left( pk, \left( \mathbf{c}_w^{(1)} || \mathbf{c}_w^{(2)} \right) \right)$ , where  $\mathbf{M}_r \begin{bmatrix} \mathbf{c}_r^{(1)} \\ \mathbf{c}_r^{(2)} \end{bmatrix} = \mathbf{y}_r$ , for  $r \in \{v, w\}$ . Besides,  $\mathbf{c}_r^{(1)}$  and  $\mathbf{c}_r^{(2)}$  are  $t + 1$ - and  $t$ -sized vectors, respectively.
- 4: All parties compute the polynomials  $C_r^{(1)}(x) = \sum_{j=0}^t \mathbf{c}_{r,j}^{(1)} x^{t-j}$ , and  $C_r^{(2)}(x) = x^t + \sum_{j=1}^t \mathbf{c}_{r,j-1}^{(2)} x^{t-j}$ , for  $r \in \{v, w\}$ , then compute

$$\text{Enc}(pk, z) = \text{Enc}(pk, C_v^{(1)}(x) \cdot C_w^{(2)}(x) - C_w^{(1)}(x) \cdot C_v^{(2)}(x))$$

by invoking  $\mathcal{F}_{\text{OMM}}$ .<sup>11</sup> Here  $C_r^{(b)}(x)$  are evaluated on a random selected point  $\alpha^* \leftarrow_{\$} \{\alpha_1, \dots, \alpha_{4t+2}\}$ .

- 5: All parties jointly use  $\mathcal{F}_{\text{DecZero}}$  to check if  $z = 0$ . If it is, output 1. Otherwise, output 0.
- 

them with  $2t + 1$  elements over a field  $\mathbb{F}$ , with  $w_i \leftarrow_{\$} \mathbb{F}$ , and  $f(x) := \frac{P(x)}{Q(x)}$  is some unknown reduced rational function (i.e.,  $P(x), Q(x)$  are co-prime), where  $\deg(P) = \deg(Q) = t'$  and  $t < t'$  where  $t, t' \in \text{poly}(\lambda)$ . We also require  $Q(x)$  to be monic (to fit in our application). Additionally, assume that  $Q(v_i) \neq 0$  and  $Q(w_i) \neq 0$  for every  $i \in [2t + 1]$ .

If we recover two rational function  $f_V(x), f_W(x)$  by interpolation on  $V, W$ , respectively, then

$$\Pr [f_V(x) = f_W(x)] \leq \text{negl}(\lambda)$$

over the choice of  $v_i, w_i$ .

*Proof.* Let  $f_V(x) = A(x)/B(x)$  the rational function recovered by rational interpolation over the support set  $V$  and let  $f(x) = P(x)/Q(x)$  be the rational function interpolated over any  $2t' + 1$  interpolation points. We have that  $f_V(v_i) = f(v_i)$  for all  $i \in [2t + 1]$  and hence

$$\frac{A(v_i)}{B(v_i)} = \frac{P(v_i)}{Q(v_i)} \Leftrightarrow A(v_i)Q(v_i) = P(v_i)B(v_i).$$

Since  $\gcd(P(x), Q(x)) = 1$ , then the polynomial  $\tilde{P}(x) = A(x)Q(x) - P(x)B(x)$  is different from the null polynomial (as  $\deg(P) = t' > t = \deg(A)$ ). Moreover,  $v_i$  is a root of  $\tilde{P}(x)$ , for all  $i \in [2t + 1]$ , and  $\deg \tilde{P}(x) \leq t + t'$  (which means that  $\tilde{P}(x)$  has at most  $t + t'$  roots).

Analogously, let  $f_W = C(x)/D(x)$  be the rational function resulting from interpolating over the support set  $W$  and let  $\tilde{Q}(x) = C(x)Q(x) - D(x)P(x)$ . We have that  $\tilde{Q}(w_i) = 0$  for all  $i \in [2t + 1]$ . Hence, if  $f_V(x) = f_W(x)$ , then we have that the points  $w_i$  are also roots of  $\tilde{P}(x)$ . But, since the points  $w_i$  are chosen uniformly at random from  $\mathbb{F}$  (which is of exponential size when compared to  $t, t'$ ), then there is a negligible probability that all  $w_i$ 's are roots of  $\tilde{P}(x)$ .

Concretely,

$$\begin{aligned} \Pr [f_V = f_W] &\leq \Pr [\tilde{P}(w_i) = 0 \forall i \in [2t + 1]] \\ &= \prod_i^{2t+1} \Pr [\tilde{P}(w_i) = 0] \leq \left( \frac{\deg \tilde{P}}{|\mathbb{F}|} \right)^{2t+1} \end{aligned}$$

which is negligible for  $|\mathbb{F}| \in \omega(2^{\log \lambda})$ .  $\square$

**Theorem 1 (Correctness).** *The protocol secDT is correct.*

*Proof.* The protocol interpolates two polynomials from two different support sets. Then, it checks if the two interpolated polynomials are the same by computing

$$C_v^{(1)}(x) \cdot C_w^{(2)}(x) - C_w^{(1)}(x) \cdot C_v^{(2)}(x)$$

which should be equal to 0 if  $C_v^{(1)}(x)/C_v^{(2)}(x) = C_w^{(1)}(x)/C_w^{(2)}(x)$ .

If  $t' \leq t$ , then by Lemma 3, there is a unique rational function can be recovered thus the final output of the algorithm should be 1. On the other hand, if  $t' > t$ , the linear system can be either unsolvable or solvable but yielding two different solutions with overwhelming probability by Lemma 4. In this case, the protocol outputs 0.  $\square$

**Theorem 2.** *The protocol  $\text{secDT}$  EUC-securely realizes  $\mathcal{F}_{\text{SDT}}$  with shared ideal functionality  $\mathcal{F}_{\text{Gen}}$  in the  $(\mathcal{F}_{\text{ORank}}, \mathcal{F}_{\text{OMM}}, \mathcal{F}_{\text{OLS}}, \mathcal{F}_{\text{DecZero}})$ -hybrid model against semi-honest adversaries corrupting at most  $N - 1$  parties, given that TPKE is IND-CPA.*

*Proof (Sketch).* The simulator sends the corrupted parties' input to the ideal functionality and obtains the output (either 0 or 1). Then, it simulates the ideal functionalities  $(\mathcal{F}_{\text{ORank}}, \mathcal{F}_{\text{OMM}}, \mathcal{F}_{\text{OLS}}, \mathcal{F}_{\text{DecZero}})$  so that the output in the real-world execution is the same as in the ideal-world execution. In particular, the simulator is able to recover the secret key shares via  $\mathcal{F}_{\text{ORank}}, \mathcal{F}_{\text{OMM}}, \mathcal{F}_{\text{OLS}}$  and, thus, simulate  $\mathcal{F}_{\text{DecZero}}$  in the right way.

Indistinguishability of executions holds given that TPKE is IND-CPA.  $\square$

*Communication Complexity.* When we instantiate  $\mathcal{F}_{\text{OLS}}$  with the protocol from the previous section, the communication complexity of  $\text{secDT}$  is  $\mathcal{O}(Nt^2)$ .

## 4 Multi-party Threshold Private Set Intersection

We present our protocol for Threshold PSI in the multi-party setting. Our protocol to privately compute the intersection can be seen as a generalization of Ghosh and Simkin protocol [17] where we replace the OLE by a TPKE (which fits nicer in a multi-party setting). The main difference between our protocol and theirs is in the cardinality test protocol used.

We begin by presenting the protocol to securely compute a cardinality testing between  $N$  sets. Then, we plug everything together in a PSI protocol.

### 4.1 Secure Cardinality Testing

*Ideal Functionality.* The ideal functionality for Secure Cardinality Testing receives the sets from all the parties and outputs 1 if and only if the intersection between these sets is larger than some threshold. Else, no information is disclosed. The ideal functionality for multi-party cardinality testing is given as follows.

#### $\mathcal{F}_{\text{MPCT}}$ functionality

*Parameters:*  $\text{sid}, N, n, t \in \mathbb{N}$  known to both parties.

- Upon receiving  $(\text{sid}, P_i, S_i)$  from party  $P_i$ ,  $\mathcal{F}_{\text{MPCT}}$  stores  $S_i$  and ignores future messages from  $P_i$  with the same  $\text{sid}$ ;
- Once  $\mathcal{F}_{\text{MPCT}}$  has stored all inputs  $S_i$ , for  $i \in [N]$ , it does the following: If  $|S_\cap| \geq n - t$ ,  $\mathcal{F}_{\text{MPCT}}$  outputs 1 to all parties, where  $|S_\cap| = \left| \bigcap_{i=1}^N S_i \right|$ . Else, it returns 0.



---

**Protocol 2** Private Cardinality Test for Multi-party MPCT
 

---

**Setup:** Values  $\alpha_1, \dots, \alpha_{4t+2} \leftarrow_{\$} \mathbb{F}$ , threshold  $t \in \mathbb{N}$  and  $N$  parties. Functionalities  $\mathcal{F}_{\text{Gen}}$  and  $\mathcal{F}_{\text{SDT}}$ , and a IND-CPA TPKE  $\text{TPKE} = (\text{Gen}, \text{Enc}, \text{Dec})$ .

**Input:** Each party  $P^i$  inputs a set  $S_i = \{a_i^{(1)}, \dots, a_i^{(n)}\} \in \mathbb{F}^n$ .

- 1: Each party  $P_i$  sends request  $(\text{sid}, \text{request}_i)$  to  $\mathcal{F}_{\text{Gen}}$  and receives a secret key share  $\text{sk}_i$  and a public key  $\text{pk}$ , which is known to every party involved in the protocol.
  - 2: Each party  $P_i$  encodes its set as a polynomial  $P_i(x) = \prod_{j=1}^n (x - a_i^{(j)})$  and evaluates it on  $4t + 2$  points. That is, it computes  $P_i(\alpha_1), \dots, P_i(\alpha_{4t+2})$ . It encrypts the points, that is,  $c_i^{(j)} \leftarrow \text{Enc}(\text{pk}, r_i \cdot P_i(\alpha_j))$  for a uniformly chosen  $r_i \leftarrow_{\$} \mathbb{F}$ . Finally, it broadcasts  $\{c_i^{(j)}\}_{j \in [4t+2]}$ .
  - 3: Party  $P_1$  computes  $d^{(j)} = (\sum_{i=1}^N c_i^{(j)}) / P_1(\alpha_j)$  for each  $j \in [4t + 2]$ . Then, sends  $\{\alpha_i, d^{(j)}\}_j$  for every  $j$ , and  $\text{sk}_1$  to the ideal functionality  $\mathcal{F}_{\text{SDT}}$ .<sup>13</sup> Each party  $P_i$ , for  $i = 2, \dots, N$ , send  $\text{sk}_i$  to  $\mathcal{F}_{\text{SDT}}$  to check if the degree of the numerator (and the denominator) is at most  $t$ .
  - 4: Upon receiving  $b \in \{0, 1\}$  from the ideal functionality  $\mathcal{F}_{\text{SDT}}$ , every party outputs  $b$ .
- 

*Protocol.* We introduce our multiparty Protocol 2 (based on degree test protocol). In the following,  $\mathcal{F}_{\text{Gen}}$  be the ideal functionality defined in Sect. 2.1 and  $\mathcal{F}_{\text{SDT}}$  be the functionality defined in Sect. 3.2.

*Analysis.* We now proceed to the analysis of the protocol described above.

**Lemma 5.** *Given  $n$  characteristic polynomials with same degree from  $\mathbb{F}[x]$ , denoted as  $P_1(x), \dots, P_n(x)$ , we argue that, for any  $j$ ,  $P'(x) = \sum_{i=1}^n r_i \cdot P_i(x)$  and  $P_j(x)$  are relatively prime with probability  $1 - \text{negl}(\log |\mathbb{F}|)$  if  $P_1(x), \dots, P_n(x)$  are mutually relatively prime, where  $r_i \leftarrow_{\$} \mathbb{F}$  is a uniformly random element.*

*Proof.* Supposing there is a common divisor of two polynomials  $P'(x)$  and  $P_j(x)$ , since  $P_j(x)$  is a characteristic polynomial, we denote  $(x - s)$  the common divisor. Therefore, we have  $P'(s) = 0$  which can be represented as  $\sum_{i=1}^n r_i \cdot P_i(s) = 0$ . However, from the mutually relative primality of  $P_1(x), \dots, P_n(x)$ , we know that  $P_i(s)$  cannot be zero simultaneously which means there exists at least one  $i^*$  to make  $P_{i^*}(s) \neq 0$ . Moreover,  $r_i$  are all sampled uniformly from  $\mathbb{F}$ , the weighted sum of  $r_i$  will not be zero with all but negligible probability. This is a contradiction. Therefore,  $P'(x)$  and  $P_j(x)$  will share a common divisor only with negligible probability.  $\square$

**Theorem 3 (Correctness).** *The protocol MPCT described above is correct.*

*Proof.* Note that the encryption  $d^{(j)}$  computed by party  $P_1$  are equal to

$$d^{(j)} = \text{Enc} \left( \text{pk}, \left( \sum_{i=1}^N r_i \cdot P_i(\alpha_j) \right) / P_1(\alpha_j) \right).$$

---

<sup>13</sup> Here,  $\mathcal{F}_{\text{SDT}}$  has shared functionality  $\mathcal{F}_{\text{Gen}}$ .

Also, observe that

$$\begin{aligned} \frac{\sum_{i=1}^N r_i \cdot P_i(\alpha_j)}{P_1(\alpha_j)} &= \frac{P_{\cap_i S_i}(\alpha_j) \cdot \sum_i^N r_i \cdot P_{S_i \setminus (\cap_{k \neq i} S_k)}(\alpha_j)}{P_{\cap_i S_i}(\alpha_j) \cdot P_{S_1 \setminus (\cap_{k \neq 1} S_k)}} \\ &= \frac{\sum_i^N r_i \cdot P_{S_i \setminus (\cap_{k \neq i} S_k)}(\alpha_j)}{P_{S_1 \setminus (\cap_{k \neq 1} S_k)}(\alpha_j)}, \end{aligned}$$

in this way, we make the numerator and denominator relatively prime except with negligible probability by Lemma 5.

Observe that  $\deg \sum_i^N r_i \cdot P_{S_i \setminus (\cap_{k \neq i} S_k)}(x) \leq t$  and  $\deg P_{S_1 \setminus (\cap_{k \neq 1} S_k)}(x) \leq t$  if and only if  $S_\cap \geq n - t$ . Hence, by the correctness of  $\mathcal{F}_{\text{SDT}}$ , the protocol outputs 1 if  $S_\cap \geq n - t$ , and 0 otherwise.  $\square$

**Theorem 4.** *The protocol MPCT securely realizes functionality  $\mathcal{F}_{\text{MPCT}}$  in the  $(\mathcal{F}_{\text{Gen}}, \mathcal{F}_{\text{SDT}})$ -hybrid model against any semi-honest adversaries corrupting up to  $N - 1$  parties, given that TPKE is IND-CPA.*

*Proof.* Assume that the adversary is corrupting  $N - k$  parties in the protocol, for  $k = 1, \dots, N - 1$ . The simulator creates the secret keys and the public key of a threshold PKE in the setup phase while simulating  $\mathcal{F}_{\text{Gen}}$  and distributes the secret keys between every party. The simulator Sim takes the inputs (which are sets of size  $n$ , say  $S_{i_1}, \dots, S_{i_{N-k}}$ ) of the corrupted parties and send them to the ideal functionality  $\mathcal{F}_{\text{MPCT}}$ . It receives the output  $b$  from the ideal functionality. If  $b = 0$ , the simulator chooses  $k$  uniformly chosen sets such that  $|\cap_{i=1}^N S_i| < n - t$  and proceed the simulation as the honest parties would do. If  $b = 1$ , the simulator chooses  $k$  uniformly chosen random sets such that  $|\cap_{i=1}^N S_i| \geq n - t$  and proceed the simulation as the honest parties would do. Note that it can simulate the ideal functionality  $\mathcal{F}_{\text{SDT}}$  since it knows all the secret keys of the threshold PKE.

Indistinguishability of executions follows immediately from the IND-CPA property of the underlying threshold PKE scheme.  $\square$

*Communication Complexity.* When we instantiate the  $\mathcal{F}_{\text{SDT}}$  with the protocol from the previous section, each party broadcasts  $\tilde{\mathcal{O}}(t^2)$ . Hence, the total communication complexity is  $\tilde{\mathcal{O}}(Nt^2)$ , assuming a broadcast channel.

## 4.2 Multi-party Threshold Private Set Intersection Protocol

In this section, we extend Ghosh and Simkin protocol [17] to the multi-party setting using TPKE. We make use of the cardinality testing designed above to get the Protocol 3.

*Analysis.* We now proceed to the analysis of the protocol described above. We start by analyzing the correctness of the protocol and then its security.

**Theorem 5 (Correctness).** *The protocol MTPSI is correct.*

---

**Protocol 3** Multi-Party Threshold PSI MTPSI
 

---

**Setup:** Given public parameters as follows: Values  $\alpha_1, \dots, \alpha_{3t+1} \leftarrow_{\$} \mathbb{F}$ , threshold  $t \in \mathbb{N}$  and  $N$  parties. Functionalities  $\mathcal{F}_{\text{Gen}}$  and  $\mathcal{F}_{\text{MPCT}}$ , and a threshold additively PKE TPKE = (Gen, Enc, Dec).

**Input:** Each party  $P_i$  inputs a set  $S_i = \{a_i^{(1)}, \dots, a_i^{(n)}\} \in \mathbb{F}^n$ .

- 1: Each party  $P_i$  sends its set  $S_i$  to  $\mathcal{F}_{\text{MPCT}}$ . If the functionality  $\mathcal{F}_{\text{MPCT}}$  outputs 0, then every party  $P_i$  outputs  $\perp$  and terminates the protocol.
  - 2: Each party  $P_i$  sends request  $(\text{sid}, \text{request}_i)$  to  $\mathcal{F}_{\text{Gen}}$  and receives a secret key share  $\text{sk}_i$  and a public key  $\text{pk}$ , which is known to every party involved in the protocol.
  - 3: **for all** Party  $P_i$  **do**
  - 4: It encodes its set as a polynomial  $P_i(x) = \prod_{j=1}^n (x - a_i^{(j)})$  and evaluates it on  $3t + 1$  points. That is, it computes  $P_i(\alpha_1), \dots, P_i(\alpha_{3t+1})$ .
  - 5: It samples  $R_i(x) \leftarrow_{\$} \mathbb{F}[x]$  such that  $\deg R_i(x) = t$ .
  - 6: It encrypts these points using  $\text{pk}$ , that is, it computes  $c_i^{(j)} = \text{Enc}(\text{pk}, R_i(\alpha_j) \cdot P_i(\alpha_j))$  for every  $j \in [3t + 1]$ .
  - 7: It broadcasts  $\{c_i^{(j)}\}_{j \in [3t+1]}$ .
  - 8: **end for**
  - 9: Party  $P_1$  adds the ciphertexts to get  $d^{(j)} = \sum_i^N c_i^{(j)}$  for each  $j \in [3t + 1]$ . It broadcasts  $\{d^{(j)}\}_{j \in [3t+1]}$ .
  - 10: They mutually decrypt  $\{d^{(j)}\}_{j \in [3t+1]}$  to learn  $V^{(j)} \leftarrow \text{Dec}(\text{sk}, d_N^{(j)})$  for  $j \in [3t + 1]$ .
  - 11:  $P_1$  computes the points  $\tilde{V}^{(j)} = V^{(j)} / P_1(\alpha_j)$  for  $j \in [3t + 1]$ .
  - 12:  $P_1$  interpolates a rational function using the pairs of points  $(\alpha_j, \tilde{V}^{(j)})$ .
  - 13:  $P_1$  recovers the polynomial  $P_{S_1 \setminus (\cap_i S_i)}(x)$  in the denominator.
  - 14:  $P_1$  evaluates  $P_{S_1 \setminus \cap_i S_i}(x)$  on every point of its set  $\{a_1^{(1)}, \dots, a_1^{(n)}\}$  to compute  $\cap_i S_i$ . That is, whenever  $P_{S_1 \setminus \cap_i S_i}(a_1^j) \neq 0$ , then  $a_1^j \in \cap_i S_i$ .
  - 15: It broadcasts the output  $\cap_i S_i$ .
- 

*Proof.* Assume that  $|S_1 \setminus (\cap_{i=2}^N S_i)| \leq t$  (note that this condition is guaranteed after resorting to the functionality  $\mathcal{F}_{\text{MPCT}}$  in the first step of the protocol). After the execution of the protocol, party  $P_1$  obtains the points  $V^{(j)} = \sum_i^N P_i(\alpha_j) \cdot R_i(\alpha_j)$ . Then,

$$\begin{aligned}
 \tilde{V}^{(j)} &= \frac{V^{(j)}}{P_1(\alpha_j)} = \frac{\sum_i^N P_i(\alpha_j) \cdot R_i(\alpha_j)}{P_1(\alpha_j)} \\
 &= \frac{P_{\cap_i S_i}(\alpha_j) \cdot \sum_i^N P_{S_i \setminus (\cap_{k \neq i} S_k)}(\alpha_j) \cdot R_i(\alpha_j)}{P_{\cap_i S_i}(\alpha_j) \cdot P_{S_1 \setminus (\cap_{k \neq 1} S_k)}(\alpha_j)} \\
 &= \frac{\sum_i^N P_{S_i \setminus (\cap_{k \neq i} S_k)}(\alpha_j) \cdot R_i(\alpha_j)}{P_{S_1 \setminus (\cap_{k \neq 1} S_k)}(\alpha_j)}.
 \end{aligned}$$

Since  $P_1$  has  $3t + 1$  evaluated points of the rational function above, then it can interpolate a rational function to recover the polynomial  $P_{S_1 \setminus (\cap_{k \neq 1} S_k)}$ . This is possible because of Lemma 2 and the fact that

$$\deg \left( \sum_i^N P_{S_i \setminus (\cap_{k \neq i} S_k)}(\alpha_j) \cdot R_i(\alpha_j) \right) \leq 2t \quad \text{and} \quad \deg (P_{S_1 \setminus (\cap_{k \neq 1} S_k)}(\alpha_j)) \leq t.$$

Having computed the polynomial  $P_{S_1 \setminus (\cap_{k \neq 1} S_k)}$ , party  $P_1$  can compute the intersection because the roots of this polynomial are exactly the elements in  $S_1 \setminus (\cap_{k \neq 1} S_k)$ .  $\square$

**Theorem 6.** *The protocol MTPSI securely realizes functionality  $\mathcal{F}_{\text{MTPSI}}$  in the  $(\mathcal{F}_{\text{Gen}}, \mathcal{F}_{\text{MPCT}})$ -hybrid model against any semi-honest adversary corrupting up to  $N - 1$  parties.*

*Proof.* Let  $\mathcal{A}$  be an adversary corrupting up to  $k$  parties involved in the protocol, for any  $k \in [N - 1]$ . Let  $P_{i_1}, \dots, P_{i_k}$  be the corrupted parties. The simulator  $\text{Sim}$  works as follows:

1. It sends the inputs of the corrupted parties,  $S_{i_1}, \dots, S_{i_k}$ , to the ideal functionality  $\mathcal{F}_{\text{MTPSI}}$ .  $\text{Sim}$  either receives  $\perp$  or  $\cap_i S_i$  from the ideal functionality  $\mathcal{F}_{\text{MTPSI}}$ .
2.  $\text{Sim}$  waits for  $\mathcal{A}$  to send the corrupted parties' inputs to the ideal functionality  $\mathcal{F}_{\text{MPCT}}$ . If  $\text{Sim}$  has received  $\perp$  from  $\mathcal{F}_{\text{MPCT}}$ , then  $\text{Sim}$  leaks 0 to  $\mathcal{A}$  (and  $\mathcal{Z}$ ) and terminates the protocol. Else,  $\text{Sim}$  leaks 1 and continues.
3.  $\text{Sim}$  waits for  $\mathcal{A}$  to send a request  $(\text{sid}, \text{request}_{i_j})$  for each of the corrupted parties (that is, for  $j \in [k]$ ) to  $\mathcal{F}_{\text{Gen}}$ . Upon receiving such requests,  $\text{Sim}$  generates  $(\text{pk}, \text{sk}_1, \dots, \text{sk}_N) \leftarrow \text{Gen}(1^\lambda, N)$  and returns  $(\text{pk}, \text{sk}_{i_j})$  for each of the requests.
4. For each party  $P_\ell$  such that  $\ell \neq i_j$  (where  $j \in [k]$ ),  $\text{Sim}$  picks a random polynomial  $U_\ell(x)$  of degree  $n - |\cap_i S_i| + t$  and sends  $\text{Enc}(\text{pk}, R_\ell(\alpha_j) \cdot P_{\cap_i S_i}(\alpha_j) \cdot U_\ell(\alpha_j))$ , where  $R_\ell(x)$  is chosen uniformly at random such that  $\deg R_\ell(x) = t$ . From now on,  $\text{Sim}$  simulates the dummy parties as in the protocol.

We now argue that both the simulation and the real-world scheme are indistinguishable from the point-of-view of any environment  $\mathcal{Z}$ . In the real-world scheme, party  $P_1$  obtains the polynomial

$$V(x) = P_{\cap_i S_i}(x) \cdot \sum_i^N P_{S_i \setminus (\cap_{k \neq i} S_k)}(x) \cdot R_i(x)$$

evaluated in  $3t + 1$  points. Assume that  $P_1$  is corrupted by  $\mathcal{A}$ . Even in this case, there is an index  $\ell$  for which  $\mathcal{A}$  does not know the polynomial  $R_\ell(x)$ . More precisely, we have that

$$V(x) = P_{\cap_i S_i}(x) \cdot \left( \left( \sum_{i \neq \ell} P_{S_i \setminus (\cap_{k \neq i} S_k)}(x) \cdot R_i(x) \right) + P_{S_\ell \setminus (\cap_{k \neq \ell} S_k)}(x) \cdot R_\ell(x) \right).$$

First, note that

$$\begin{aligned} \deg \left( \sum_{i \neq \ell} P_{S_i \setminus (\cap_{k \neq i} S_k)}(x) \cdot R_i(x) \right) &= \deg P_{S_\ell \setminus (\cap_{k \neq \ell} S_k)}(x) \cdot R_\ell(x) \\ &= n - |\cap_i S_i| + t \leq 2t. \end{aligned}$$

Moreover, we have for any  $i \in [N]$  that  $\deg P_{S_i \setminus (\cap_{k \neq i} S_k)} \leq t$ ,  $\deg R_i(x) = t$  and  $\gcd(P_{S_i \setminus (\cap_{k \neq i} S_k)}, P_{S_j \setminus (\cap_{k \neq j} S_k)}) = 1$  for any  $j \neq i$ . Hence, by Lemma 1, we can build a sequence of hybrids where we replace  $V(x)$  by the polynomial  $V'(x) = P_{\cap_i S_i}(x) \cdot U(x)$ , where  $\deg U(x) = n - |\cap_i S_i| + t$ , as in the ideal-world execution. Indistinguishability of executions follows.  $\square$

*Communication Complexity.* When we instantiate the ideal functionality  $\mathcal{F}_{\text{MPCT}}$  with the protocol from the previous section the scheme has communication complexity  $\tilde{O}(Nt^2)$ .

**Acknowledgment.** *Pedro Branco:* Part of this work was done while the author was at CISP. The author is supported by DP-PMI and FCT (Portugal) through the grant PD/BD/135181/2017. This work is supported by Security and Quantum Information Group of Instituto de Telecomunicações, by the Fundação para a Ciência e a Tecnologia (FCT) through national funds, by FEDER, COMPETE 2020, and by Regional Operational Program of Lisbon, under UIDB/50008/2020.

*Nico Döttling:* Part of this work was done while visiting Simons Institute, Berkeley, California. This work is partially funded by the Helmholtz Association within the project “Trustworthy Federated Data Analytics” (TFDA) (funding number ZT-I-OO14).

*Sihang Pu:* Part of this work was done while visiting Simons Institute, Berkeley, California.

## A Preliminaries Cont’d

### A.1 Threshold Public-Key Encryption

In this work, we will use Public-Key Encryption schemes and a variant of it: Threshold Public-key Encryption. We now define Threshold Public-key Encryption. Such schemes can be instantiated from several hardness assumptions such as DDH, DCR or pairing-based assumptions [21].

**Definition 3** (Threshold Public-Key Encryption). *A Threshold Public-Key Encryption (TPKE) scheme is defined by the following algorithms:*

- $(\text{pk}, \text{sk}_1, \dots, \text{sk}_N) \leftarrow \text{Gen}(1^\lambda, N)$  takes as input a security parameter. It outputs a public key  $\text{pk}$  and  $N$  secret keys  $(\text{sk}_1, \dots, \text{sk}_N)$ .
- $c \leftarrow \text{Enc}(\text{pk}, m)$  takes as input a public key  $\text{pk}$  and a message  $m \in \{0, 1\}^*$ . It outputs a ciphertext  $c$ .
- $c' \leftarrow \text{Dec}(\text{sk}_i, c)$  takes as input one of the secret keys  $\text{sk}_i$  and a ciphertext. It outputs a share decryption  $c'$  of  $c$ .

*Correctness.* For any  $N \in \mathbb{N}$  and any permutation  $\pi : [N] \rightarrow [N]$ , we have that

$$\Pr \left[ m \leftarrow \text{Dec}(\text{sk}_{\pi(N)}, \text{Dec}(\text{sk}_{\pi(N-1)}, \dots \text{Dec}(\text{sk}_{\pi(1)}, \text{Enc}(\text{pk}, m)) \dots) \right) ] = 1$$

where  $(\text{pk}, \text{sk}_1, \dots, \text{sk}_N) \leftarrow \text{Gen}(1^\lambda, N)$ .

*IND-CPA Security.* For any  $N \in \mathbb{N}$ , any permutation  $\pi : [N] \rightarrow [N]$  and any adversary  $\mathcal{A}$ , we require that

$$\Pr \left[ \begin{array}{l} (\mathbf{pk}, \mathbf{sk}_1, \dots, \mathbf{sk}_N) \leftarrow \text{Gen}(1^\lambda, N) \\ b \leftarrow \mathcal{A}(c, \mathbf{st}) : \begin{array}{l} (m_0, m_1, \mathbf{st}) \leftarrow \mathcal{A}(\mathbf{pk}, \mathbf{sk}_{\pi(1)}, \dots, \mathbf{sk}_{\pi(k)}) \\ b \leftarrow_{\mathfrak{s}} \{0, 1\} \\ c \leftarrow \text{Enc}(\mathbf{pk}, m_b) \end{array} \end{array} \right] \leq \text{negl}(\lambda)$$

for any  $k < N$ .

*Additive Homomorphism.* We also assume that the TPKE (or plain PKE) is homomorphic for additive operation.<sup>14</sup> That is, for all  $(\mathbf{pk}, \mathbf{sk}_1, \dots, \mathbf{sk}_N) \leftarrow \text{Gen}(1^\lambda, N)$ , we can define two groups  $(\mathcal{M}, \oplus), (\mathcal{C}, \otimes)$  such that, given two ciphertexts  $c_1 \leftarrow \text{Enc}(\mathbf{pk}, m_1)$  and  $c_2 \leftarrow \text{Enc}(\mathbf{pk}, m_2)$ , we require that

$$c_1 \otimes c_2 = \text{Enc}(\mathbf{pk}, m_1 \oplus m_2).$$

By abuse of notation, we usually denote the operations of  $\mathcal{M}$  and  $\mathcal{C}$  as  $+$ .

## A.2 Linear Algebra

We first introduce minimal polynomials of a sequence and of a matrix. Then we present how they can be used to solve linear algebra related problems.

**Minimal Polynomial of a Matrix.** The minimal polynomial of a sequence  $\mathbf{a}$  is the least degree polynomial  $m$  such that  $\langle m \rangle = \text{Ann}(\mathbf{a})$  where  $\text{Ann}(\mathbf{a})$  is the annihilator ideal of  $\mathbf{a}$  (that is, the ideal such that every element  $f$  of  $\text{Ann}(\mathbf{a})$  satisfies  $f \cdot \mathbf{a} = 0$ ).

**Lemma 6** (Lemma 3 in [24]). *Let  $\mathbf{A} \in \mathbb{F}^{n \times n}$  and let  $m_{\mathbf{A}}$  be the minimal polynomial of matrix  $\mathbf{A}$ . For  $\mathbf{u}, \mathbf{v} \leftarrow_{\mathfrak{s}} \mathbb{F}^n$ , we have  $m_{\mathbf{A}} = m_{\mathbf{a}'}$  with probability at least  $1 - 2 \deg(m_{\mathbf{A}}) / |\mathbb{F}|$ , where  $\mathbf{a}' = (\mathbf{u}^T \mathbf{A}^i \mathbf{v})_{i \in \mathbb{N}}$ . Moreover,  $m_{\mathbf{a}'}$  can be calculated using a Boolean circuit of size  $\mathcal{O}(nk \log n \log k \log \log k)$  where  $k = \log |\mathbb{F}|$ .*

## Compute the Rank of a Matrix and Solve a Linear System

**Lemma 7** ([23]). *Let  $\mathbf{A} \in \mathbb{F}^{n \times n}$  of (unknown) rank  $r$ . Let  $\mathbf{U}$  and  $\mathbf{L}$  be randomly chosen unit upper triangular and lower triangular Toeplitz matrices in  $\mathbb{F}^{n \times n}$ , and let  $\mathbf{B} = \mathbf{UAL}$ . Let us denote the  $i \times i$  leading principal of  $\mathbf{B}$  by  $\mathbf{B}_i$ . The probability that  $\det(\mathbf{B}_i) \neq 0$  for all  $1 \leq i \leq r$  is greater than  $1 - n^2 / |\mathbb{F}|$ .*

**Lemma 8** ([23]). *Let  $\mathbf{B} \in \mathbb{F}^{n \times n}$  with leading invertible principals up to  $\mathbf{B}_r$ , where  $r$  is the (unknown) rank of  $\mathbf{B}$ . Let  $\mathbf{X}$  be a randomly chosen diagonal matrix in  $\mathbb{F}^{n \times n}$ . Then,  $r = \deg(m_{\mathbf{XB}}) - 1$  with probability greater than  $1 - n^2 / |\mathbb{F}|$ .*

<sup>14</sup> From now on, we always assume that PKE and TPKE used in this work fulfill this property, unless stated otherwise.

## B Oblivious Linear Algebra

### B.1 Oblivious Matrix Multiplication

*Protocol.* The following Protocol 4 allows several parties to jointly compute the (encrypted) product of two encrypted matrices. Note that the protocol can also be used to compute the encryption of the product of two encrypted values in  $\mathbb{F}$ .

---

#### Protocol 4 Secure Multiplication secMult

---

**Setup:** Each party  $P_i$  has a secret share  $sk_i$  of a secret key for a public key  $pk$  of a TPKE scheme  $TPKE = (\text{Gen}, \text{Enc}, \text{Dec})$ .

**Input:** Party  $P_1$  inputs  $\text{Enc}(pk, \mathbf{M}_l)$  and  $\text{Enc}(pk, \mathbf{M}_r)$ , where  $\mathbf{M}_l, \mathbf{M}_r \in \mathbb{F}^{t \times t}$ .

**Goal:** Every one knows the product  $\text{Enc}(\mathbf{M}_l \cdot \mathbf{M}_r)$ .

- 1: **for all** party  $P_i$  **do**
  - 2: It samples two random matrices  $\mathbf{R}_l^{(i)}, \mathbf{R}_r^{(i)} \leftarrow_{\$} \mathbb{F}^{t \times t}$ .
  - 3: It computes  $c_l^{(i)} = \text{Enc}(pk, \mathbf{R}_l^{(i)})$ ,  $c_r^{(i)} = \text{Enc}(pk, \mathbf{R}_r^{(i)})$ ,  $d_r^{(i)} = \text{Enc}(pk, \mathbf{M}_l \cdot \mathbf{R}_r^{(i)})$ ,  $d_l^{(i)} = \text{Enc}(pk, \mathbf{R}_l^{(i)} \cdot \mathbf{M}_r)$ .
  - 4: It broadcasts  $\{c_l^{(i)}, c_r^{(i)}, d_l^{(i)}, d_r^{(i)}\}$ .
  - 5: **end for**
  - 6: Each party  $P_i$  computes  $\tilde{c}^{(i)} = \text{Enc}(pk, \sum_{j \neq i} \mathbf{R}_l^{(i)} \cdot \mathbf{R}_r^{(j)})$  (using  $c_r^{(j)}$  and  $\mathbf{R}_l^{(i)}$ ) and broadcasts  $\tilde{c}^{(i)}$ .
  - 7: All parties mutually decrypt i)  $\text{Enc}(\mathbf{M}'_l) := \text{Enc}(pk, \mathbf{M}_l) + \sum_j c_l^{(j)}$  (to obtain  $\mathbf{M}'_l \in \mathbb{F}^{t \times t}$ ), ii)  $\text{Enc}(\mathbf{M}'_r) := \text{Enc}(pk, \mathbf{M}_r) + \sum_j c_r^{(j)}$  (to obtain  $\mathbf{M}'_r \in \mathbb{F}^{t \times t}$ )
  - 8: **for all** party  $P_i$  **do**
  - 9: It computes  $\tilde{d} = \text{Enc}(pk, \mathbf{M}'_l \cdot \mathbf{M}'_r)$ .
  - 10: It outputs  $e = \tilde{d} - \sum_j d_l^{(j)} - \sum_j d_r^{(j)} - \sum_j \tilde{c}^{(j)}$
  - 11: **end for**
- 

*Analysis.* We proceed to the analysis of the protocol described above.

**Lemma 9** (Correctness). *The protocol secMult is correct.*

*Proof* The correctness is straightforward. □

**Lemma 10** (Security). *The protocol secMult securely EUC-realizes  $\mathcal{F}_{\text{OMM}}$  with shared ideal functionality  $\mathcal{F}_{\text{Gen}}$  against semi-honest adversaries corrupting up to  $N - 1$  parties, given that TPKE is IND-CPA.*

*Proof (Sketch).* Assume that the adversary corrupts  $N - k$  parties. The simulator takes the inputs from these parties and send them to the ideal functionality. Upon receiving the encrypted value  $\text{Enc}(pk, \mathbf{M}_l \cdot \mathbf{M}_r)$ , it simulates the protocol as the honest parties would do.

We now prove that no set of at most  $N - 1$  colluding parties can extract information about  $\mathbf{M}_l, \mathbf{M}_r$ . First, observe that any set of  $N - 1$  parties cannot extract any information about encrypted values that are not decrypted during

the protocol (because there is always a missing secret key share) given that TPKE is IND-CPA. Second, we analyze the matrix  $\mathbf{M}'_l$  (which is decrypted during the protocol). We have that  $\mathbf{M}'_l = \mathbf{M}_l + \sum_j \mathbf{R}_l^{(j)}$ . Hence, there is always at least one matrix  $\mathbf{R}_l^{(\ell)}$  which is unknown to the adversary and that perfectly hides the matrix  $\mathbf{M}_l$  (the same happens  $\mathbf{M}'_r$ ).  $\square$

*Complexity.* The communication complexity of the protocol is dominated by the messages carrying the (encrypted) matrix. Hence, assuming a broadcast channel between the parties, the protocol has communication complexity of  $\mathcal{O}(Nt^2)$  where  $t$  is the size of the input matrices and  $N$  the number of parties involved in the protocol.

## B.2 Compute the Rank of a Matrix

*Protocol.* We present this protocol in the full version of this paper in [6].

*Complexity.* Each party broadcasts  $\mathcal{O}(t^2 k \log t)$  bits of information, where  $k = \log |\mathbb{F}|$ . To see this, note that the communication of the protocol is dominated by the computation of the circuit that computes the degree of  $\mathbf{a}$  and this can be implemented with communication cost of  $\mathcal{O}(t^2 k \log t)$  [24]. Assuming a broadcast channel, the communication complexity is  $\mathcal{O}(Nt^2)$ .

## B.3 Invert a Matrix

In this section, we present and analyze a protocol that allows  $N$  parties to invert an encrypted matrix. In this setting, each of the  $N$  parties holds a secret share of a public key  $\text{pk}$  of a TPKE. Given an encrypted matrix, they want to compute an encryption of the inverse of this matrix.

*Ideal Functionality.* The ideal functionality of oblivious rank computation is defined below.

### $\mathcal{F}_{\text{OInv}}$ functionality

*Parameters:*  $\text{sid}, N, q, t \in \mathbb{N}$  and  $\mathbb{F}$ , where  $\mathbb{F}$  is a field of order  $q$ , known to the  $N$  parties involved in the protocol.  $\text{pk}$  public-key of a threshold PKE scheme.

- Upon receiving  $(\text{sid}, P_1, \text{Enc}(\text{pk}, \mathbf{M}))$  from party  $P_1$  (where  $\mathbf{M} \in \mathbb{F}^{t \times t}$  is a non-singular matrix),  $\mathcal{F}_{\text{ORank}}$  outputs  $\text{Enc}(\text{pk}, \mathbf{M}^{-1})$  to  $P_1$  and  $(\text{Enc}(\text{pk}, \mathbf{M}), \text{Enc}(\text{pk}, \mathbf{M}^{-1}))$  to all other parties  $P_i$ , for  $i = 2, \dots, N$ .

*Protocol.* This protocol allows  $N$  parties to jointly compute the encryption of the inverse of a matrix, given that this matrix is non-singular. Please refer to the full version of this paper in [6] to see details.



*Analysis.* The proofs of the following lemmas follow the same lines as the proofs in the analysis of `secMult` protocol. We state the lemmas but omit the proofs for brevity.

**Lemma 11.** *The protocol `seclnv` is correct.*

**Lemma 12.** *The protocol `seclnv` securely EUC-realizes  $\mathcal{F}_{\text{Olnv}}$  with shared ideal functionality  $\mathcal{F}_{\text{Gen}}$  against semi-honest adversaries corrupting up to  $N-1$  parties, given that TPKE is IND-CPA.*

*Complexity.* Each party broadcasts  $\mathcal{O}(t^2)$  bits of information. The communication complexity of the protocol is  $\mathcal{O}(Nt^2)$ , assuming a broadcast channel.

## B.4 Secure Unary Representation

Following [24], we present a protocol that allows to securely compute the unary representation of a matrix.

*Ideal Functionality.* The ideal functionality for Secure Unary Representation is given below.

### $\mathcal{F}_{\text{SUR}}$ functionality

*Parameters:*  $\text{sid}, N, q, t \in \mathbb{N}$  and  $\mathbb{F}$ , where  $\mathbb{F}$  is a field of order  $q$ , known to the  $N$  parties involved in the protocol.  $\text{pk}$  public-key of a threshold PKE scheme.

- Upon receiving  $(\text{sid}, P_1, \text{Enc}(\text{pk}, r))$  from party  $P_1$  (where  $r \in \mathbb{F}$  and  $r \leq t$ ),  $\mathcal{F}_{\text{SUR}}$  computes  $(\text{Enc}(\text{pk}, \delta_1), \dots, \text{Enc}(\text{pk}, \delta_t))$  such that  $\delta_i = 1$  if  $i \leq r$ , and  $\delta_i = 0$  otherwise. The functionality outputs  $(\text{Enc}(\text{pk}, \delta_1), \dots, \text{Enc}(\text{pk}, \delta_t))$  to  $P_1$  and  $(\text{Enc}(\text{pk}, r), (\text{Enc}(\text{pk}, \delta_1), \dots, \text{Enc}(\text{pk}, \delta_t)))$  to all other parties  $P_i$ , for  $i = 2, \dots, N$ .

*Protocol.* A protocol for secure unary representation can be implemented with the help of a binary-conversion protocol [37]. That is, given  $\text{Enc}(\text{pk}, r)$ , all parties jointly compute  $\text{Enc}(\text{pk}, \delta_i)$ , where  $\delta_i = 1$ , if  $i \leq r$ , and  $\delta_i = 0$  otherwise, via a Boolean circuit (which can be securely implemented based on Paillier cryptosystem).

*Communication Complexity.* We can calculate the result using a Boolean circuit of size  $O(r \log t)$ , thus the communication complexity is  $O(Nr \log t)$ .

---

**Protocol 5** Secure Linear Solve **seCLS**


---

**Setup:** Each party has a secret key share  $sk_i$  for a public key  $pk$  of a TPKE  $TPKE = (\text{Gen}, \text{Enc}, \text{Dec})$ . The parties have access to the ideal functionalities  $\mathcal{F}_{\text{ORank}}$ ,  $\mathcal{F}_{\text{OInv}}$  and  $\mathcal{F}_{\text{SUR}}$ .

**Input:** Party  $P_1$  inputs  $\text{Enc}(pk, \mathbf{M})$  where  $\mathbf{M} \in \mathbb{F}^{t \times t}$  is a non-singular matrix.

- 1: All parties jointly compute an encryption of the rank  $\text{Enc}(pk, r)$  of  $\mathbf{M}$  via the ideal functionality  $\mathcal{F}_{\text{ORank}}$ .
  - 2: Set  $\text{Enc}(pk, \mathbf{M}') := \text{Enc}(pk, \mathbf{M})$  and  $\text{Enc}(pk, \mathbf{y}') := \text{Enc}(pk, \mathbf{y})$ .
  - 3: **for**  $i$  from 1 to  $N$  **do**
  - 4:  $P_i$  samples two non-singular matrices  $\mathbf{R}_i, \mathbf{Q}_i$  from  $\mathbb{F}^{t \times t}$ . It calculates  $\text{Enc}(pk, \mathbf{M}') = \text{Enc}(pk, \mathbf{R}_i \mathbf{M}' \mathbf{Q}_i)$  and  $\text{Enc}(pk, \mathbf{y}') = \text{Enc}(pk, \mathbf{R}_i \mathbf{y}')$ .  $P_i$  broadcasts  $\text{Enc}(pk, \mathbf{M}'), \text{Enc}(pk, \mathbf{y}')$ .
  - 5: **end for**
  - 6: All the parties jointly compute  $\text{Enc}(\delta_1), \dots, \text{Enc}(\delta_t)$  by invoking  $\mathcal{F}_{\text{SUR}}$  on input  $\text{Enc}(pk, r)$ . They set  $\text{Enc}(pk, \Delta) := \text{Enc}\left(pk, \begin{bmatrix} \delta_1 & \dots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \dots & \delta_t \end{bmatrix}\right)$ . Finally, they compute  $\text{Enc}(pk, \mathbf{N}) := \text{Enc}(pk, \mathbf{M}' \cdot \Delta + \mathbf{I}_t - \Delta)$ , where  $\mathbf{I}_t \in \mathbb{F}^{t \times t}$  is the identity matrix.
  - 7: All the parties jointly compute  $\text{Enc}(\mathbf{N}^{-1})$  by invoking  $\mathcal{F}_{\text{OInv}}$  on input  $\text{Enc}(pk, \mathbf{N})$ .
  - 8: Each party  $P^i$  samples  $\mathbf{u}_i \leftarrow \mathbb{F}^t$  and broadcasts  $(\text{Enc}(pk, \mathbf{M}' \mathbf{u}_i), \text{Enc}(pk, \mathbf{u}_i))$ .
  - 9: All parties jointly compute  $\text{Enc}(pk, \mathbf{u}') = \text{Enc}(pk, \mathbf{N}^{-1} \mathbf{y}'_r)$  by invoking  $\mathcal{F}_{\text{OMM}}$ , where  $\text{Enc}(pk, \mathbf{y}'_r) = \text{Enc}(pk, (\mathbf{y}' + \sum_j \mathbf{M}' \mathbf{u}_j) \Delta)$ . Then they set  $\text{Enc}(pk, \mathbf{x}) = \text{Enc}(pk, (\sum_j \mathbf{u}_j) - \mathbf{u}')$ .
  - 10: **for**  $i$  from  $N$  to 1 **do**
  - 11:  $P_i$  calculates  $\text{Enc}(pk, \mathbf{x}) = \text{Enc}(pk, \mathbf{Q}_i^{-1} \mathbf{x})$ .  $P_i$  broadcasts  $\text{Enc}(pk, \mathbf{x})$ .
  - 12: **end for**
  - 13:  $P_1$  outputs  $\text{Enc}(pk, \mathbf{x})$ .
- 

## B.5 Solve a Linear System

*Protocol.* We now present the Protocol 5 that allows multiple parties to solve an encrypted linear system. In the following, we assume that the system has at least one solution (note that this can be guaranteed using the **secRank** protocol).

**Lemma 13** (Correctness). *The protocol **seCLS** is correct.*

*Proof.* The proof follows directly from [23, 24]. □

**Lemma 14.** *The protocol **seCLS** securely EUC-realizes  $\mathcal{F}_{\text{OLS}}$  with shared ideal functionality  $\mathcal{F}_{\text{Gen}}$  in the  $(\mathcal{F}_{\text{ORank}}, \mathcal{F}_{\text{OInv}}, \mathcal{F}_{\text{SUR}})$ -hybrid model against semi-honest adversaries corrupting up to  $N - 1$  parties, given that TPKE is IND-CPA.*

*Communication Complexity.* Each party broadcasts  $\mathcal{O}(t^2 k \log t)$  bits of information where  $k = |\mathbb{F}|$ . The total communication complexity is  $\tilde{\mathcal{O}}(t^2)$ .

## References

1. Badrinarayanan, S., Miao, P., Raghuraman, S., Rindal, P.: Multi-party threshold private set intersection with sublinear communication. In: PKC 2021 (2021)

2. Berke, A., Bakker, M., Vepakomma, P., Larson, K., Pentland, A.S.: Assessing disease exposure risk with location data: a proposal for cryptographic preservation of privacy (2020). <https://arxiv.org/abs/2003.14412>
3. Boneh, D., Boyen, X., Shacham, H.: Short group signatures. In: Franklin, M. (ed.) CRYPTO 2004. LNCS, vol. 3152, pp. 41–55. Springer, Heidelberg (2004). [https://doi.org/10.1007/978-3-540-28628-8\\_3](https://doi.org/10.1007/978-3-540-28628-8_3)
4. Boneh, D., Goh, E.-J., Nissim, K.: Evaluating 2-DNF formulas on ciphertexts. In: Kilian, J. (ed.) TCC 2005. LNCS, vol. 3378, pp. 325–341. Springer, Heidelberg (2005). [https://doi.org/10.1007/978-3-540-30576-7\\_18](https://doi.org/10.1007/978-3-540-30576-7_18)
5. Bouman, N.J., de Vreede, N.: New protocols for secure linear algebra: pivoting-free elimination and fast block-recursive matrix decomposition. IACR Cryptology ePrint Archive 2018/703 (2018)
6. Branco, P., Döttling, N., Pu, S.: Multiparty cardinality testing for threshold private set intersection. Cryptology ePrint Archive, Report 2020/1307 (2020). <https://eprint.iacr.org/2020/1307>
7. Canetti, R.: Universally composable security: a new paradigm for cryptographic protocols. In: 42nd Annual Symposium on Foundations of Computer Science, Las Vegas, NV, USA, 14–17 October 2001, pp. 136–145. IEEE Computer Society Press. <https://doi.org/10.1109/SFCS.2001.959888>
8. Canetti, R., Dodis, Y., Pass, R., Walfish, S.: Universally composable security with global setup. In: Vadhan, S.P. (ed.) TCC 2007. LNCS, vol. 4392, pp. 61–85. Springer, Heidelberg (2007). [https://doi.org/10.1007/978-3-540-70936-7\\_4](https://doi.org/10.1007/978-3-540-70936-7_4)
9. Cramer, R., Damgård, I.: Secure distributed linear algebra in a constant number of rounds. In: Kilian, J. (ed.) CRYPTO 2001. LNCS, vol. 2139, pp. 119–136. Springer, Heidelberg (2001). [https://doi.org/10.1007/3-540-44647-8\\_7](https://doi.org/10.1007/3-540-44647-8_7)
10. Cramer, R., Damgård, I., Nielsen, J.B.: Multiparty computation from threshold homomorphic encryption. In: Pfitzmann, B. (ed.) EUROCRYPT 2001. LNCS, vol. 2045, pp. 280–300. Springer, Heidelberg (2001). [https://doi.org/10.1007/3-540-44987-6\\_18](https://doi.org/10.1007/3-540-44987-6_18)
11. Dachman-Soled, D., Malkin, T., Raykova, M., Yung, M.: Efficient robust private set intersection. In: Abdalla, M., Pointcheval, D., Fouque, P.-A., Vergnaud, D. (eds.) ACNS 2009. LNCS, vol. 5536, pp. 125–142. Springer, Heidelberg (2009). [https://doi.org/10.1007/978-3-642-01957-9\\_8](https://doi.org/10.1007/978-3-642-01957-9_8)
12. De Cristofaro, E., Kim, J., Tsudik, G.: Linear-complexity private set intersection protocols secure in malicious model. In: Abe, M. (ed.) ASIACRYPT 2010. LNCS, vol. 6477, pp. 213–231. Springer, Heidelberg (2010). [https://doi.org/10.1007/978-3-642-17373-8\\_13](https://doi.org/10.1007/978-3-642-17373-8_13)
13. Dong, C., Chen, L., Wen, Z.: When private set intersection meets big data: an efficient and scalable protocol. In: Sadeghi, A.R., Gligor, V.D., Yung, M. (eds.) ACM CCS 2013: 20th Conference on Computer and Communications Security, Berlin, Germany, 4–8 November 2013, pp. 789–800. ACM Press (2013). <https://doi.org/10.1145/2508859.2516701>
14. Elgamal, T.: A public key cryptosystem and a signature scheme based on discrete logarithms. IEEE Trans. Inf. Theory **31**(4), 469–472 (1985)
15. Freedman, M.J., Nissim, K., Pinkas, B.: Efficient private matching and set intersection. In: Cachin, C., Camenisch, J.L. (eds.) EUROCRYPT 2004. LNCS, vol. 3027, pp. 1–19. Springer, Heidelberg (2004). [https://doi.org/10.1007/978-3-540-24676-3\\_1](https://doi.org/10.1007/978-3-540-24676-3_1)

16. Ghosh, S., Nilges, T.: An algebraic approach to maliciously secure private set intersection. In: Ishai, Y., Rijmen, V. (eds.) EUROCRYPT 2019, Part III. LNCS, vol. 11478, pp. 154–185. Springer, Cham (2019). [https://doi.org/10.1007/978-3-030-17659-4\\_6](https://doi.org/10.1007/978-3-030-17659-4_6)
17. Ghosh, S., Simkin, M.: The communication complexity of threshold private set intersection. In: Boldyreva, A., Micciancio, D. (eds.) CRYPTO 2019, Part II. LNCS, vol. 11693, pp. 3–29. Springer, Cham (2019). [https://doi.org/10.1007/978-3-030-26951-7\\_1](https://doi.org/10.1007/978-3-030-26951-7_1)
18. Ghosh, S., Simkin, M.: The communication complexity of threshold private set intersection. Cryptology ePrint Archive, Report 2019/175 (2019). <https://eprint.iacr.org/2019/175>
19. Grigorescu, E., Jung, K., Rubinfeld, R.: A local decision test for sparse polynomials. Inf. Process. Lett. **110**(20), 898–901 (2010). <https://doi.org/10.1016/j.ipl.2010.07.012>
20. Hallgren, P., Orlandi, C., Sabelfeld, A.: Privatepool: privacy-preserving ridesharing. In: 2017 IEEE 30th Computer Security Foundations Symposium (CSF), pp. 276–291 (2017)
21. Hazay, C., Venkitasubramaniam, M.: Scalable multi-party private set-intersection. In: Fehr, S. (ed.) PKC 2017, Part I. LNCS, vol. 10174, pp. 175–203. Springer, Heidelberg (2017). [https://doi.org/10.1007/978-3-662-54365-8\\_8](https://doi.org/10.1007/978-3-662-54365-8_8)
22. Ion, M., Kreuter, B., Nergiz, E., Patel, S., Saxena, S., Seth, K., Shanahan, D., Yung, M.: Private intersection-sum protocol with applications to attributing aggregate ad conversions. Cryptology ePrint Archive, Report 2017/738 (2017). <http://eprint.iacr.org/2017/738>
23. Kaltofen, E., David Saunders, B.: On Wiedemann’s method of solving sparse linear systems. In: Mattson, H.F., Mora, T., Rao, T.R.N. (eds.) AAEC 1991. LNCS, vol. 539, pp. 29–38. Springer, Heidelberg (1991). [https://doi.org/10.1007/3-540-54522-0\\_93](https://doi.org/10.1007/3-540-54522-0_93)
24. Kiltz, E., Mohassel, P., Weinreb, E., Franklin, M.: Secure linear algebra using linearly recurrent sequences. In: Vadhan, S.P. (ed.) TCC 2007. LNCS, vol. 4392, pp. 291–310. Springer, Heidelberg (2007). [https://doi.org/10.1007/978-3-540-70936-7\\_16](https://doi.org/10.1007/978-3-540-70936-7_16)
25. Kissner, L., Song, D.: Privacy-preserving set operations. In: Shoup, V. (ed.) CRYPTO 2005. LNCS, vol. 3621, pp. 241–257. Springer, Heidelberg (2005). [https://doi.org/10.1007/11535218\\_15](https://doi.org/10.1007/11535218_15)
26. Kolesnikov, V., Kumaresan, R., Rosulek, M., Trieu, N.: Efficient batched oblivious PRF with applications to private set intersection. In: Weippl, E.R., Katzenbeisser, S., Kruegel, C., Myers, A.C., Halevi, S. (eds.) ACM CCS 2016: 23rd Conference on Computer and Communications Security, Vienna, Austria, 24–28 October 2016, pp. 818–829. ACM Press (2016). <https://doi.org/10.1145/2976749.2978381>
27. Meadows, C.: A more efficient cryptographic matchmaking protocol for use in the absence of a continuously available third party. In: 1986 IEEE Symposium on Security and Privacy, pp. 134–134 (1986)
28. Minsky, Y., Trachtenberg, A., Zippel, R.: Set reconciliation with nearly optimal communication complexity. IEEE Trans. Inf. Theory **49**(9), 2213–2218 (2003). <https://doi.org/10.1109/TIT.2003.815784>
29. Nissim, K., Weinreb, E.: Communication efficient secure linear algebra. In: Halevi, S., Rabin, T. (eds.) TCC 2006. LNCS, vol. 3876, pp. 522–541. Springer, Heidelberg (2006). [https://doi.org/10.1007/11681878\\_27](https://doi.org/10.1007/11681878_27)

30. Paillier, P.: Public-key cryptosystems based on composite degree residuosity classes. In: Stern, J. (ed.) EUROCRYPT 1999. LNCS, vol. 1592, pp. 223–238. Springer, Heidelberg (1999). [https://doi.org/10.1007/3-540-48910-X\\_16](https://doi.org/10.1007/3-540-48910-X_16)
31. Pinkas, B., Rosulek, M., Trieu, N., Yanai, A.: SpOT-light: lightweight private set intersection from sparse OT extension. In: Boldyreva, A., Micciancio, D. (eds.) CRYPTO 2019, Part III. LNCS, vol. 11694, pp. 401–431. Springer, Cham (2019). [https://doi.org/10.1007/978-3-030-26954-8\\_13](https://doi.org/10.1007/978-3-030-26954-8_13)
32. Pinkas, B., Schneider, T., Segev, G., Zohner, M.: Phasing: private set intersection using permutation-based hashing. In: Jung, J., Holz, T. (eds.) USENIX Security 2015: 24th USENIX Security Symposium, Washington, DC, USA, 12–14 August 2015, pp. 515–530. USENIX Association (2015)
33. Pinkas, B., Schneider, T., Weinert, C., Wieder, U.: Efficient circuit-based PSI via cuckoo hashing. In: Nielsen, J.B., Rijmen, V. (eds.) EUROCRYPT 2018, Part III. LNCS, vol. 10822, pp. 125–157. Springer, Cham (2018). [https://doi.org/10.1007/978-3-319-78372-7\\_5](https://doi.org/10.1007/978-3-319-78372-7_5)
34. Pinkas, B., Schneider, T., Zohner, M.: Faster private set intersection based on OT extension. In: Fu, K., Jung, J. (eds.) USENIX Security 2014: 23rd USENIX Security Symposium, San Diego, CA, USA, 20–22 August 2014, pp. 797–812. USENIX Association (2014)
35. Rindal, P., Rosulek, M.: Improved private set intersection against malicious adversaries. In: Coron, J.-S., Nielsen, J.B. (eds.) EUROCRYPT 2017, Part I. LNCS, vol. 10210, pp. 235–259. Springer, Cham (2017). [https://doi.org/10.1007/978-3-319-56620-7\\_9](https://doi.org/10.1007/978-3-319-56620-7_9)
36. Rindal, P., Rosulek, M.: Malicious-secure private set intersection via dual execution. In: Thuraisingham, B.M., Evans, D., Malkin, T., Xu, D. (eds.) ACM CCS 2017: 24th Conference on Computer and Communications Security, Dallas, TX, USA, 31 October–2 November 2017, pp. 1229–1242. ACM Press (2017). <https://doi.org/10.1145/3133956.3134044>
37. Schoenmakers, B., Tuyls, P.: Efficient binary conversion for Paillier encrypted values. In: Vaudenay, S. (ed.) EUROCRYPT 2006. LNCS, vol. 4004, pp. 522–537. Springer, Heidelberg (2006). [https://doi.org/10.1007/11761679\\_31](https://doi.org/10.1007/11761679_31)