Multiparty Cardinality Testing for Threshold Private Set Intersection

Pedro Branco* Nico Döttling† Sihang Pu‡

October 21, 2020

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 the sets 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 $\tilde{\mathcal{O}}(Nt^2)$ communication complexity. We thus obtain a Threshold PSI scheme for N parties with communication complexity $\tilde{\mathcal{O}}(Nt^2)$.

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., [Mea86, FNP04, KS05, DMRY09, DKT10, DCW13, PSZ14, PSSZ15, KKRT16, RR17a, HV17, RR17b, PSWW18, GN19, GS19a, PRTY19]). Also, numerous applications have been proposed for PSI such as contact discovery, advertising, etc (see for example [IKN+17] and references therein). More recently, PSI has also been proposed as a solution for private contact tracing (e.g., [BBV+20]).

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 intersection between the input sets of the parties is very large, say $n - t$, where n is the size of the input sets and t is some *threshold* such that $t << n$. 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 [GS19a].

Despite its theoretical and practical appeal, there are just a few works that consider this problem [HOS17, GN19, GS19a], and just one of them achieves communication complexity independent of n [GS19a], in the two party setting.

*IT, University of Lisbon.

†Helmholtz Center for Information Security (CISPA).

‡Helmholtz Center for Information Security (CISPA).

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 [GS19a].

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 [HOS17]: 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 some threshold $n - t$; 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 $t' \leq 2t$. 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 $t << n$. The protocol needs $\tilde{\mathcal{O}}(Nt^2)$ 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 [NW06, KMWF07], 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 [Elg85], DCR [Pai99], or from several pairings assumptions [BBS04, BGN05]) 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 $\tilde{\mathcal{O}}(Nt^2)$.

1.2.1 Concurrent Work

Recently, Ghosh and Simkin [GS19b] updated their paper with a generalization to the multi-party case which is similar to the one presented in this paper in Section 4. However, they leave as a major open problem the design of a new cardinality-test that extends nicely to multiple parties, a problem on which we make relevant advances in this work.

In a concurrent work. Badrinarayanan, Miao and Rindal [BMR20] also proposed new protocos 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 $\mathcal{O}(Nt)$. However, we remark that the goal of our work was to reduce the assumptions needed for Threshold PSI. They also propose an TPKE-based protocol that solves a slightly different problem: the parties learn the intersection if and only if the set difference between the sets is large, that is, $\left(\cup_{i=0}^{N} S_i\right) \setminus \left(\cap_{i=0}^{N} S_i\right)$. This protocol achieves the same communication complexity as ours.

1.3 Technical Outline

We now give a high-level overview of the techniques we use to achieve the results discussed above. For precise statements, we refer the reader to the technical sections.

1.3.1 Threshold PSI: The Protocol of [GS19a]

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 [GS19a] is that *set reconciliation techniques* (developed by Minsky et al. [MTZ03]) can be applied in this scenario: if $|S_A \cap S_B| \geq n - t$, then

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

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 interpolation on these points, Bob can recover the denominator (which reveals the intersection).

Of course, Bob should not be able to recover the numerator $P_{A\setminus B}$. So, [GS19a] 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 $|S_A \cap S_N| \geq n - t$ 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 [GS19a], 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 [GJR10] and securely computing its determinant (via a generic secure linear algebra protocol from [KMWF07]), both parties can determine if $|S_A \cap S_B| \geq n - t$. The total communication complexity of this protocol is $\mathcal{O}(t^2)$.[1]

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

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

[1] Given this, we conclude that the communication complexity of the Threshold PSI protocol of [GS19a] is dominated by this Cardinality Testing protocol.

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 $\mathcal{O}((Nt)^2)$.

1.3.2 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 [GS19a] 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 [NW06] and [KMWF07]), there are two parties, one holding an encryption of a matrix $\mathsf{Enc}(\mathsf{pk}, \mathbf{M})$ and another one holding the corresponding secret key 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 $\mathbf{Mx} = \mathbf{y}$ where both $\mathbf{M}$ and $\mathbf{y}$ are encrypted. We can easily extend this problem to the multi-party case: Consider N parties, $\mathsf{P}_1, \dots, \mathsf{P}_N$, 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 [KMWF07] 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[2]. Hence, our protocols allow N parties to solve a linear system of the form $\mathbf{Mx} = \mathbf{y}$ 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 $a_j^{(i)} \in S_j$, for N different sets, and the rational function[3]

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

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

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 [Can01]. However, there is a subtle issua 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.

[2] We need a bit-conversion protocol such as [ST06] to convert between binary circuits and algebra operations.

[3] 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.

We solve this problem by using the Externalized UC framework [CDPW07]. 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. [BMR20] also encounter the same problem as we did and they opted to not prove security of the secure linear algebra protocols 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 [GS19a]. 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 j) and knowing that the intersection is larger than $n - t$, parties can securely compute the rational function[4] $(P_{S_1} + \cdots + P_{S_N})/P_{S_1}$. By interpolating the rational function on any $\mathcal{O}(t)$ points, party 1 can recover the denominator and compute the intersection.

The main difference between our protocol and the one in [GS19a] is that we replace the OLE calls used in [GS19a] 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 [CD01] 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. [BdV18] 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 [NW06] and Kiltz et al. in [KMWF07]. In the following, consider (square) matrices of size t over a field $\mathbb{F}$. These two works take different approaches: [NW06] obliviously 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 [KMWF07] has error probability decreases polynomially with $|\mathbb{F}|$, which is negligible when $\mathbb{F}$ is of exponentially size.[5]

2 Preliminaries

If S is a finite set, then $x \leftarrow_\$ S$ 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 $[N] = \{1, \ldots, N\}$.

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 λ.

[4] Again, we omit the randomization of the polynomials. Actually, without randomization, these methods (including [GS19a]) are exactly the same as the technique for set reconciliation problem in [MTZ03].

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

Moreover, we define the functionality $\mathcal{F}_{\mathsf{DecZero}}$, which allows N parties, each of them holding a secret share sk_i, to learn if a ciphertext is an encryption of 0 and nothing else. That is, $\mathcal{F}_{\mathsf{DecZero}}$ receives as input a ciphertext c and the secret shares of each of the parties. It outputs 0, if $0 \leftarrow \mathsf{Dec}(\mathsf{sk}_, \ldots \mathsf{Dec}(\mathsf{sk}_N, c) \ldots)$, and 1 otherwise. Note that these functionalities can be securely realized using on varies PKE schemes such as El Gamal PKE or Pailler[6]PKE [HV17].

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 [Can01] to analyze the security of our protocols.[7] Throughout this work, we only consider semi-honest adversaries, unless stated otherwise. We denote the underlying environment by $\mathcal{Z}$. For a protocol π and a real-world adversary $\mathcal{A}$, we denote the real-world ensemble by $\mathsf{EXEC}_{\pi,\mathcal{A},\mathcal{Z}}$ Similarly, for an ideal functionality $\mathcal{F}$ and a simulator Sim, we denote the ideal-world ensemble by $\mathsf{IDEAL}_{\mathcal{F},\mathsf{Sim},\mathcal{Z}}$.

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

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

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.

$\mathcal{F}_{\mathsf{MTPSI}}$ functionality

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

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

2.2.1 Externalized UC Protocol with Global Setup

We introduce a notion of protocol emulation from [CDPW07], called externalized UC emulation (EUC), which is a simplified version of UC with global setup (GUC).

[6]We will assume the message space of Paillier's cryptosystem as a field as also mentioned in [KMWF07].

[7]We refer the reader to [Can01] for a detailed explanation of the framework.

Definition 2 (EUC-Emulation [CDPW07]). *We say that π EUC-realizes $\mathcal{F}$ with respect to shared functionality $\bar{\mathcal{G}}$ (or, in shorthand, that π $\bar{\mathcal{G}}$-EUC-emulates ϕ) if for any PPT adversary $\mathcal{A}$ there exists a PPT adversary* Sim *such that for any shared functionality $\bar{\mathcal{G}}$, we have:*

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

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

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 ([KS05]). *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 $R_1, R_2 \leftarrow_{\$} \mathbb{F}_p$ 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 Theom.2 and Theom.3 of [KS05] 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 ([MTZ03]). *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 ([MTZ03]). *Choose V to be a support set*[8] *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 Obliviously 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 $t \in Z$. To this end, and inspired by the works of [NW06, KMWF07], we present a multi-party protocol to obliviously solve a linear system $\mathbf{Mx} = \mathbf{y}$ 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 briefness, the protocols are presented in Appendix B These protocol 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.

[8] A support set is a set of pairs (x, y).

Note that if we let parties P_i input their encrypted matrix $\mathsf{Enc}(\mathbf{M})$, 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 [BMR20]: 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 ($\mathcal{F}_{\mathsf{Gen}}$ from Section 2.1).

3.1.1 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.

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

$\mathcal{F}_{\mathsf{OMM}}$ **functionality**

Parameters: $\mathsf{sid}, N, q, t \in \mathbb{N}$ and $\mathbb{F}$, where $\mathbb{F}$ is a field of order q, known to the N parties involved in the protocol.

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

- Upon receiving $(\mathsf{sid}, \mathsf{P}_1, \mathsf{Enc}(\mathsf{pk}, \mathbf{M}_l), \mathsf{Enc}(\mathsf{pk}, \mathbf{M}_r))$ from party P_1 (where $\mathbf{M}_l, \mathbf{M}_r \in \mathbb{F}^{t\times t}$), $\mathcal{F}_{\mathsf{OMM}}$ outputs $\mathsf{Enc}(\mathsf{pk}, \mathbf{M}_l \cdot \mathbf{M}_r)$ to P_1 and $(\mathsf{Enc}(\mathsf{pk}, \mathbf{M}_l), \mathsf{Enc}(\mathsf{pk}, \mathbf{M}_r), \mathsf{Enc}(\mathsf{pk}, \mathbf{M}_l \cdot \mathbf{M}_r))$ to all other parties P_i, for $i = 2, \ldots, N$.

3.1.2 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.

$\mathcal{F}_{\mathsf{ORank}}$ **functionality**

Parameters: $\mathsf{sid}, N, q, t \in \mathbb{N}$ and $\mathbb{F}$, where $\mathbb{F}$ is a field of order q, known to the N parties involved in the protocol.

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

- Upon receiving $(\mathsf{sid}, \mathsf{P}_1, \mathsf{Enc}(\mathsf{pk}, \mathbf{M}))$ from party P_1 (where $\mathbf{M} \in \mathbb{F}^{t\times t}$), $\mathcal{F}_{\mathsf{ORank}}$ outputs $\mathsf{Enc}(\mathsf{pk}, \mathrm{rank}(\mathbf{M}))$ to P_1 and $(\mathsf{Enc}(\mathsf{pk}, \mathbf{M}), \mathsf{Enc}(\mathsf{pk}, \mathrm{rank}(\mathbf{M}))$ to all other parties P_i, for $i = 2, \ldots, N$.

3.1.3 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 [KMWF07] 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 [CDN01], and for the unary representation, a binary-conversion protocol [ST06] is required. All of above protocols can be based on Paillier cryptosystem.

Ideal Functionality. We give an ideal functionality of oblivious linear solve for multiparty as follows.

> **$\mathcal{F}_{\mathsf{OLS}}$ functionality**
>
> **Parameters:** $\mathsf{sid}, N, q, t \in \mathbb{N}$ and $\mathbb{F}$, where $\mathbb{F}$ is a field of order q , known to the N parties involved in the protocol. pk public-key of a threshold PKE scheme.
>
> **Global Setup:** pk public-key of a threshold PKE scheme and sk_i distributed to each party P_i via $\mathcal{F}_{\mathsf{Gen}}$.
>
> - Upon receiving $(\mathsf{sid}, \mathsf{P}_1, \mathsf{Enc}(\mathsf{pk}, \mathbf{M}), \mathsf{Enc}(\mathsf{pk}, \mathbf{y}))$ from party P_1 (assuming there is a solution $\mathbf{x}$ for $\mathbf{Mx} = \mathbf{y}$), $\mathcal{F}_{\mathsf{OLS}}$ outputs $\mathsf{Enc}(\mathsf{pk}, \mathbf{x})$ such that $\mathbf{Mx} = \mathbf{y}$.

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.

> **$\mathcal{F}_{\mathsf{SDT}}$ functionality**
>
> **Parameters:** $\mathsf{sid}, N, q, n, t \in \mathbb{N}$, $\mathbb{F}$ is a field of order q and t is a predefined threshold, known to the N parties involved in the protocol. pk public-key of a threshold PKE scheme. $\alpha_1, \ldots, \alpha_{4t+2} \leftarrow_{\$} \mathbb{F}$ known to the N parties.
>
> **Global Setup:** pk public-key of a threshold PKE scheme and sk_i distributed to each party P_i via $\mathcal{F}_{\mathsf{Gen}}$.
>
> - Upon receiving $(\mathsf{sid}, \mathsf{P}_1, \mathsf{Enc}(\mathsf{pk}, f_1), \ldots, \mathsf{Enc}(\mathsf{pk}, f_{4t+2}))$ 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), $\mathcal{F}_{\mathsf{SDT}}$ 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.

[9] Note that this is the linear system that we need to solve in order to perform rational interpolation [MTZ03].

[10] The polynomial multiplication can be expressed as matrix multiplication.

Protocol 1 Secure Degree Test secDT

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

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

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

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

and

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

for $r = \{v, w\}$.[9]

2: All parties jointly compute $\mathsf{Enc}(\mathsf{pk}, \mathrm{rank}(\mathbf{M}_r) - \mathrm{rank}\left([\mathbf{M}_r||\mathbf{y}]\right)$ for $r \in \{v, w\}$ through two invocations of $\mathcal{F}_{\mathsf{ORank}}$ and mutually decrypt the ciphertext via $\mathcal{F}_{\mathsf{DecZero}}$. If the result is different from 0, they abort the protocol.

3: All parties mutually solve the two linear systems above using $\mathcal{F}_{\mathsf{OLS}}$ such that each party gets $\mathsf{Enc}\left(\mathsf{pk}, \left(\mathbf{c}_v^{(1)}||\mathbf{c}_v^{(2)}\right)\right)$ and $\mathsf{Enc}\left(\mathsf{pk}, \left(\mathbf{c}_w^{(1)}||\mathbf{c}_w^{(2)}\right)\right)$, where $\mathbf{M}_r \begin{bmatrix} \mathbf{c}_r^{(1)} \\ \mathbf{c}_r^{(2)} \end{bmatrix} = \mathbf{y}_r$, for $r \in \{v, w\}$.

4: All parties compute the polynomials $C_r^{(b)}(x) = x^t + \sum_{j=0}^{t} \mathbf{c}_{r,j+1}^{(b)} x^j$, for $r \in \{v, w\}$ and $b \in \{1, 2\}$, and compute

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

by invoking $\mathcal{F}_{\mathsf{OMM}}$.[10]

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

Comments. Suppose that, for an interpolation point α_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 [11] , the parties evaluate $\tilde{P}(x) = P(x)/(x-\alpha_i)$ and $\tilde{Q}(x) = Q(x)/(x-\alpha_i)$ on α_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 [MTZ03]. In more details, instead of using $\mathsf{Enc}(\mathsf{pk}, f_i)$ for α_i, we will use a tagged pair $\left(\mathsf{Enc}\left(\mathsf{pk}, s_i^{(1)}\right), \mathsf{Enc}\left(\mathsf{pk}, s_i^{(2)}\right)\right)$ 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 $\mathsf{Enc}(\mathsf{pk}, \mathbf{M}_r)$ and $\mathsf{Enc}(\mathsf{pk}, \mathbf{y}_r)$ with

$$\mathsf{Enc}\left(\mathsf{pk}, \begin{bmatrix} s_i^{(2)} r_i^t & \dots & s_i^{(2)} & -s_i^{(1)} r_i^{t-1} & \dots & -s_i^{(1)} \end{bmatrix}\right)$$

and $\mathsf{Enc}\left(\mathsf{pk}, \begin{bmatrix} s_i^{(1)} r_i^t \end{bmatrix}\right)$, 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 them with $2t$ 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 $t, t' \in \mathsf{poly}(\lambda)$. 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

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

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. 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 $f_V = f_W$, 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,

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

[11] In the case that only $Q(\alpha_i) = 0$, use a different tagged pair $(\mathsf{Enc}(\mathsf{pk}, s_i^{(1)}), \mathsf{Enc}(\mathsf{pk}, 0))$.

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* secDT *EUC-securely realizes* $\mathcal{F}_{\mathsf{SDT}}$ *with shared ideal functionality* $\mathcal{F}_{\mathsf{Gen}}$ *in the* $(\mathcal{F}_{\mathsf{ORank}}, \mathcal{F}_{\mathsf{OMM}}, \mathcal{F}_{\mathsf{OLS}}, \mathcal{F}_{\mathsf{DecZero}})$*-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 $(\mathcal{F}_{\mathsf{ORank}}, \mathcal{F}_{\mathsf{OMM}}, \mathcal{F}_{\mathsf{OLS}}, \mathcal{F}_{\mathsf{DecZero}})$ 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 $\mathcal{F}_{\mathsf{ORank}}, \mathcal{F}_{\mathsf{OMM}}, \mathcal{F}_{\mathsf{OLS}}$ and, thus, simulate $\mathcal{F}_{\mathsf{DecZero}}$ in the right way.

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

Communication complexity. When we instantiate $\mathcal{F}_{\mathsf{OLS}}$ with the protocol from the previous section, the communication complexity of 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 [GS19a] 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 output 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.

$\mathcal{F}_{\mathsf{MPCT}}$ functionality

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

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

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

Protocol 2 Private Cardinality Test for Multi-party MPCT

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

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

1: Each party P_i sends request $(\mathsf{sid}, \mathsf{request}_i)$ to $\mathcal{F}_{\mathsf{Gen}}$ and receives a secret key share sk_i and a public key 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 $P_i(\alpha_1), \ldots, P_i(\alpha_{4t+2})$. It encrypts the points, that is, $c_i^{(j)} \leftarrow \mathsf{Enc}(\mathsf{pk}, r_i \cdot P_i(\alpha_j))$ 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 sk_1 to the ideal functionality $\mathcal{F}_{\mathsf{SDT}}$.[12] Each party P_i, for $i = 2, \ldots, N$, send sk_i to $\mathcal{F}_{\mathsf{SDT}}$ 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 $\mathcal{F}_{\mathsf{SDT}}$, every party outputs b.

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 $P_1(x), \ldots, P_n(x)$, 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 $1 - \mathsf{negl}(\log |\mathbb{F}|)$ if $P_1(x), \ldots, P_n(x)$ 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 $P_1(x), \ldots, P_n(x)$, 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. □

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

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

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 $\mathcal{F}_{\mathsf{SDT}}$, the protocol outputs 1 if $S_\cap \geq n - t$, and 0 otherwise. □

Theorem 4. *The protocol* MPCT *securely realizes functionality* $\mathcal{F}_{\mathsf{MPCT}}$ *in the* $(\mathcal{F}_{\mathsf{Gen}}, \mathcal{F}_{\mathsf{SDT}})$*-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 $k = 1, \ldots, N-1$. The simulator creates the secret keys and the public key of a threshold PKE in the setup phase while simulating $\mathcal{F}_{\mathsf{Gen}}$ and distributes the secret keys between every party. The simulator Sim takes the inputs (which are sets of size n, say $S_{i_1}, \ldots, S_{i_{N-k}}$) of the corrupted parties and send them to the ideal functionality $\mathcal{F}_{\mathsf{MPCT}}$. 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 $\mathcal{F}_{\mathsf{SDT}}$ 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. □

Communication Complexity. When we instantiate the $\mathcal{F}_{\mathsf{SDT}}$ 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 [GS19a] to the multi-party setting using TPKE. We make use of the cardinality testing designed above to get the Protocol 3.

Protocol 3 Multi-Party Threshold PSI MTPSI

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

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

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

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.*

Proof. Assume that $|S_1 \setminus \left(\cap_{i=2}^N S_i\right)| \leq t$ (note that this condition is guaranteed after resorting to the functionality $\mathcal{F}_{\mathsf{MPCT}}$ 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

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

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 $S_1 \setminus (\cap_{k \neq 1} S_k)$. ☐

Theorem 6. *The protocol* MTPSI *securely realizes functionality* $\mathcal{F}_{\mathsf{MTPSI}}$ *in the* $(\mathcal{F}_{\mathsf{Gen}}, \mathcal{F}_{\mathsf{MPCT}})$*-hybrid model against any semi-honest adversarie 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 $\mathsf{P}_{i_1}, \ldots, \mathsf{P}_{i_k}$ be the corrupted parties.

The simulator Sim works as follows:

1. It sends the inputs of the corrupted parties, $S_{i_1}, \ldots, S_{i_k}$, to the ideal functionality $\mathcal{F}_{\mathsf{MTPSI}}$. Sim either receives $\perp$ or $\cap_i S_i$ from the ideal functionality $\mathcal{F}_{\mathsf{MTPSI}}$.

2. Sim waits for $\mathcal{A}$ to send the corrupted parties' inputs to the ideal functionality $\mathcal{F}_{\mathsf{MPCT}}$. If Sim has received $\perp$ from $\mathcal{F}_{\mathsf{MPCT}}$, then Sim leaks 0 to $\mathcal{A}$ (and $\mathcal{Z}$) and terminates the protocol. Else, Sim leaks 1 and continues.

3. Sim waits for $\mathcal{A}$ to send a request $(\mathsf{sid}, \mathsf{request}_{i_j})$ for each of the corrupted parties (that is, for $j \in [k]$) to $\mathcal{F}_{\mathsf{Gen}}$. Upon receiving such requests, Sim generates $(\mathsf{pk}, \mathsf{sk}_1, \ldots, \mathsf{sk}_N) \leftarrow \mathsf{Gen}(1^\lambda, N)$ and returns $(\mathsf{pk}, \mathsf{sk}_{i_j})$ for each of the requests.

4. For each party P_ℓ such that $\ell \neq i_j$ (where $j \in [k]$), Sim picks a random polynomial $U_\ell(x)$ of degree $n - |\cap_i S_i| + t$ and sends $\mathsf{Enc}(\mathsf{pk}, R_\ell(\alpha_j) \cdot P_{\cap_i S_i}(\alpha_j) \cdot U_\ell(\alpha_j))$, where $R_\ell(x)$ is chosen uniformly at random such that $\deg R_\ell(x) = t$. From now on, 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 ℓ 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

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

Moreover, we have that, for any $i \in [N]$

$$\deg P_{S_i \setminus (\cap_{k \neq i} S_k)} \leq t,$$

$\deg R_i(x) = t$ and

$$\gcd \left(P_{S_i \setminus (\cap_{k \neq i} S_k)}, P_{S_j \setminus (\cap_{k \neq j} S_k)} \right) = 1$$

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. □

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

Acknowledgment

Pedro Branco: Part of this work was done while the author was at CISPA. 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.

References

[BBS04] Dan Boneh, Xavier Boyen, and Hovav Shacham. Short group signatures. In Matthew Franklin, editor, *Advances in Cryptology – CRYPTO 2004*, volume 3152 of *Lecture Notes in Computer Science*, pages 41–55, Santa Barbara, CA, USA, August 15–19, 2004. Springer, Heidelberg, Germany.

[BBV+20] Alex Berke, Michiel Bakker, Praneeth Vepakomma, Kent Larson, and Alex 'Sandy' Pentland. Assessing disease exposure risk with location data: A proposal for cryptographic preservation of privacy, 2020.

[BdV18] Niek J. Bouman and Niels de Vreede. New protocols for secure linear algebra: Pivoting-free elimination and fast block-recursive matrix decomposition. *IACR Cryptology ePrint Archive*, 2018:703, 2018.

[BGN05] Dan Boneh, Eu-Jin Goh, and Kobbi Nissim. Evaluating 2-DNF formulas on ciphertexts. In Joe Kilian, editor, *TCC 2005: 2nd Theory of Cryptography Conference*, volume 3378 of *Lecture Notes in Computer Science*, pages 325–341, Cambridge, MA, USA, February 10–12, 2005. Springer, Heidelberg, Germany.

[BMR20] Saikrishna Badrinarayanan, Peihan Miao, and Peter Rindal. Multi-party threshold private set intersection with sublinear communication. Cryptology ePrint Archive, Report 2020/600, 2020. `https://eprint.iacr.org/2020/600`.

[Can01] Ran Canetti. Universally composable security: A new paradigm for cryptographic protocols. In *42nd Annual Symposium on Foundations of Computer Science*, pages 136–145, Las Vegas, NV, USA, October 14–17, 2001. IEEE Computer Society Press.

[CD01] Ronald Cramer and Ivan Damgård. Secure distributed linear algebra in a constant number of rounds. In Joe Kilian, editor, *Advances in Cryptology – CRYPTO 2001*, volume 2139 of *Lecture Notes in Computer Science*, pages 119–136, Santa Barbara, CA, USA, August 19–23, 2001. Springer, Heidelberg, Germany.

[CDN01] Ronald Cramer, Ivan Damgård, and Jesper Buus Nielsen. Multiparty computation from threshold homomorphic encryption. In Birgit Pfitzmann, editor, *Advances in Cryptology – EUROCRYPT 2001*, volume 2045 of *Lecture Notes in Computer Science*, pages 280–299, Innsbruck, Austria, May 6–10, 2001. Springer, Heidelberg, Germany.

[CDPW07] Ran Canetti, Yevgeniy Dodis, Rafael Pass, and Shabsi Walfish. Universally composable security with global setup. In Salil P. Vadhan, editor, *TCC 2007: 4th Theory of Cryptography Conference*, volume 4392 of *Lecture Notes in Computer Science*, pages 61–85, Amsterdam, The Netherlands, February 21–24, 2007. Springer, Heidelberg, Germany.

[DCW13] Changyu Dong, Liqun Chen, and Zikai Wen. When private set intersection meets big data: an efficient and scalable protocol. In Ahmad-Reza Sadeghi, Virgil D. Gligor, and Moti Yung, editors, *ACM CCS 2013: 20th Conference on Computer and Communications Security*, pages 789–800, Berlin, Germany, November 4–8, 2013. ACM Press.

[DKT10] Emiliano De Cristofaro, Jihye Kim, and Gene Tsudik. Linear-complexity private set intersection protocols secure in malicious model. In Masayuki Abe, editor, *Advances in Cryptology – ASIACRYPT 2010*, volume 6477 of *Lecture Notes in Computer Science*, pages 213–231, Singapore, December 5–9, 2010. Springer, Heidelberg, Germany.

[DMRY09] Dana Dachman-Soled, Tal Malkin, Mariana Raykova, and Moti Yung. Efficient robust private set intersection. In Michel Abdalla, David Pointcheval, Pierre-Alain Fouque, and Damien Vergnaud, editors, *ACNS 09: 7th International Conference on Applied Cryptography and Network Security*, volume 5536 of *Lecture Notes in Computer Science*, pages 125–142, Paris-Rocquencourt, France, June 2–5, 2009. Springer, Heidelberg, Germany.

[Elg85] T. Elgamal. A public key cryptosystem and a signature scheme based on discrete logarithms. *IEEE Transactions on Information Theory*, 31(4):469–472, 1985.

[FNP04] Michael J. Freedman, Kobbi Nissim, and Benny Pinkas. Efficient private matching and set intersection. In Christian Cachin and Jan Camenisch, editors, *Advances in Cryptology – EUROCRYPT 2004*, volume 3027 of *Lecture Notes in Computer Science*, pages 1–19, Interlaken, Switzerland, May 2–6, 2004. Springer, Heidelberg, Germany.

[GJR10] Elena Grigorescu, Kyomin Jung, and Ronitt Rubinfeld. A local decision test for sparse polynomials. *Inf. Process. Lett.*, 110(20):898–901, September 2010.

[GN19] Satrajit Ghosh and Tobias Nilges. An algebraic approach to maliciously secure private set intersection. In Yuval Ishai and Vincent Rijmen, editors, *Advances in Cryptology – EUROCRYPT 2019, Part III*, volume 11478 of *Lecture Notes in Computer Science*, pages 154–185, Darmstadt, Germany, May 19–23, 2019. Springer, Heidelberg, Germany.

[GS19a] Satrajit Ghosh and Mark Simkin. The communication complexity of threshold private set intersection. In Alexandra Boldyreva and Daniele Micciancio, editors, *Advances in Cryptology – CRYPTO 2019, Part II*, volume 11693 of *Lecture Notes in Computer Science*, pages 3–29, Santa Barbara, CA, USA, August 18–22, 2019. Springer, Heidelberg, Germany.

[GS19b] Satrajit Ghosh and Mark Simkin. The communication complexity of threshold private set intersection. Cryptology ePrint Archive, Report 2019/175, 2019. `https://eprint.iacr.org/2019/175`.

[HOS17] P. Hallgren, C. Orlandi, and A. Sabelfeld. Privatepool: Privacy-preserving ridesharing. In *2017 IEEE 30th Computer Security Foundations Symposium (CSF)*, pages 276–291, 2017.

[HV17] Carmit Hazay and Muthuramakrishnan Venkitasubramaniam. Scalable multi-party private set-intersection. In Serge Fehr, editor, *PKC 2017: 20th International Conference on Theory and Practice of Public Key Cryptography, Part I*, volume 10174 of *Lecture Notes in Computer Science*, pages 175–203, Amsterdam, The Netherlands, March 28–31, 2017. Springer, Heidelberg, Germany.

[IKN$^+$17] Mihaela Ion, Ben Kreuter, Erhan Nergiz, Sarvar Patel, Shobhit Saxena, Karn Seth, David Shanahan, and Moti Yung. Private intersection-sum protocol with applications to attributing aggregate ad conversions. Cryptology ePrint Archive, Report 2017/738, 2017. `http://eprint.iacr.org/2017/738`.

[KDS91] Erich Kaltofen and B. David Saunders. On wiedemann's method of solving sparse linear systems. In Harold F. Mattson, Teo Mora, and T. R. N. Rao, editors, *Applied Algebra, Algebraic Algorithms and Error-Correcting Codes*, pages 29–38, Berlin, Heidelberg, 1991. Springer Berlin Heidelberg.

[KKRT16] Vladimir Kolesnikov, Ranjit Kumaresan, Mike Rosulek, and Ni Trieu. Efficient batched oblivious PRF with applications to private set intersection. In Edgar R. Weippl, Stefan Katzenbeisser, Christopher Kruegel, Andrew C. Myers, and Shai Halevi, editors, *ACM CCS 2016: 23rd Conference on Computer and Communications Security*, pages 818–829, Vienna, Austria, October 24–28, 2016. ACM Press.

[KMWF07] Eike Kiltz, Payman Mohassel, Enav Weinreb, and Matthew K. Franklin. Secure linear algebra using linearly recurrent sequences. In Salil P. Vadhan, editor, *TCC 2007: 4th Theory of Cryptography Conference*, volume 4392 of *Lecture Notes in Computer Science*, pages 291–310, Amsterdam, The Netherlands, February 21–24, 2007. Springer, Heidelberg, Germany.

[KS05] Lea Kissner and Dawn Xiaodong Song. Privacy-preserving set operations. In Victor Shoup, editor, *Advances in Cryptology – CRYPTO 2005*, volume 3621 of *Lecture Notes in Computer Science*, pages 241–257, Santa Barbara, CA, USA, August 14–18, 2005. Springer, Heidelberg, Germany.

[Mea86] C. Meadows. 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*, pages 134–134, 1986.

[MTZ03] Yaron Minsky, Ari Trachtenberg, and Richard Zippel. Set reconciliation with nearly optimal communication complexity. *IEEE Trans. Information Theory*, 49(9):2213–2218, 2003.

[NW06] Kobbi Nissim and Enav Weinreb. Communication efficient secure linear algebra. In Shai Halevi and Tal Rabin, editors, *TCC 2006: 3rd Theory of Cryptography Conference*, volume 3876 of *Lecture Notes in Computer Science*, pages 522–541, New York, NY, USA, March 4–7, 2006. Springer, Heidelberg, Germany.

[Pai99] Pascal Paillier. Public-key cryptosystems based on composite degree residuosity classes. In Jacques Stern, editor, *Advances in Cryptology – EUROCRYPT'99*, volume 1592 of *Lecture Notes in Computer Science*, pages 223–238, Prague, Czech Republic, May 2–6, 1999. Springer, Heidelberg, Germany.

[PRTY19] Benny Pinkas, Mike Rosulek, Ni Trieu, and Avishay Yanai. SpOT-light: Lightweight private set intersection from sparse OT extension. In Alexandra Boldyreva and Daniele Micciancio, editors, *Advances in Cryptology – CRYPTO 2019, Part III*, volume 11694 of *Lecture Notes in Computer Science*, pages 401–431, Santa Barbara, CA, USA, August 18–22, 2019. Springer, Heidelberg, Germany.

[PSSZ15] Benny Pinkas, Thomas Schneider, Gil Segev, and Michael Zohner. Phasing: Private set intersection using permutation-based hashing. In Jaeyeon Jung and Thorsten Holz, editors, *USENIX Security 2015: 24th USENIX Security Symposium*, pages 515–530, Washington, DC, USA, August 12–14, 2015. USENIX Association.

[PSWW18] Benny Pinkas, Thomas Schneider, Christian Weinert, and Udi Wieder. Efficient circuit-based PSI via cuckoo hashing. In Jesper Buus Nielsen and Vincent Rijmen, editors, *Advances in Cryptology – EUROCRYPT 2018, Part III*, volume 10822 of *Lecture Notes in Computer Science*, pages 125–157, Tel Aviv, Israel, April 29 – May 3, 2018. Springer, Heidelberg, Germany.

[PSZ14] Benny Pinkas, Thomas Schneider, and Michael Zohner. Faster private set intersection based on OT extension. In Kevin Fu and Jaeyeon Jung, editors, *USENIX Security 2014: 23rd USENIX Security Symposium*, pages 797–812, San Diego, CA, USA, August 20–22, 2014. USENIX Association.

[RR17a] Peter Rindal and Mike Rosulek. Improved private set intersection against malicious adversaries. In Jean-Sébastien Coron and Jesper Buus Nielsen, editors, *Advances in Cryptology – EUROCRYPT 2017, Part I*, volume 10210 of *Lecture Notes in Computer Science*, pages 235–259, Paris, France, April 30 – May 4, 2017. Springer, Heidelberg, Germany.

[RR17b] Peter Rindal and Mike Rosulek. Malicious-secure private set intersection via dual execution. In Bhavani M. Thuraisingham, David Evans, Tal Malkin, and Dongyan Xu, editors, *ACM CCS 2017: 24th Conference on Computer and Communications Security*, pages 1229–1242, Dallas, TX, USA, October 31 – November 2, 2017. ACM Press.

[ST06] Berry Schoenmakers and Pim Tuyls. Efficient binary conversion for Paillier encrypted values. In Serge Vaudenay, editor, *Advances in Cryptology – EUROCRYPT 2006*, volume 4004 of *Lecture Notes in Computer Science*, pages 522–537, St. Petersburg, Russia, May 28 – June 1, 2006. Springer, Heidelberg, Germany.

Appendix 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 [HV17].

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

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

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

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

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

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

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

for any $k < N$.

Additive Homomorphism. We also assume that the TPKE (or plain PKE) is homomorphic for additive operation.[13] That is, for all $\mathsf{pk}, \mathsf{sk}_1, \ldots, \mathsf{sk}_N) \leftarrow \mathsf{Gen}(1^\lambda, N)$, we can define two groups $(\mathcal{M}, \oplus), (\mathcal{C}, \otimes)$ such that, given two ciphertexts $c_1 \leftarrow \mathsf{Enc}(\mathsf{pk}, m_1)$ and $c_2 \leftarrow \mathsf{Enc}(\mathsf{pk}, m_2)$, we require that

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

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

Ideal Functionalities. We present some ideal functionalities regarding TPKE schemes. In the following, N is the number of parties.

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

Moreover, we define the functionality $\mathcal{F}_{\mathsf{DecZero}}$, which allows N parties, each of them holding a secret share sk_i, to learn if a ciphertext is an encryption of 0 and nothing else. That is, $\mathcal{F}_{\mathsf{DecZero}}$ receives as input a ciphertext c and the secret shares of each of the parties. It outputs 0, if $0 \leftarrow \mathsf{Dec}(\mathsf{sk}_, \ldots \mathsf{Dec}(\mathsf{sk}_N, c) \ldots)$, and 1 otherwise.

Note that these functionalities can be securely realized using on varies PKE schemes such as El Gamal PKE or Pailler PKE [HV17].

[13]From now on, we always assume that PKE and TPKE used in this work fulfill this property, unless stated otherwise.

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.

A.2.1 Minimal Polynomial of a Matrix

Let $\mathbb{F}$ be field and V be a vector space over $\mathbb{F}$. An infinite sequence $\mathfrak{a} = (a_i)_{i\in\mathbb{N}} \in V^{\mathbb{N}}$ is linearly recurrent (over $\mathbb{F}$) if there exists $n \in \mathbb{N}$ and $f_0, \dots, f_n \in \mathbb{F}$ with $f_n \neq 0$ such that $\sum_{j=0}^{n} f_j a_{i+j} = 0$, for all $i \in \mathbb{N}$. We can define the multiplication of a sequence by a polynomial $f \in \mathbb{F}[x]$ of degree n by $f \cdot \mathfrak{a} = \sum_{j=0}^{n} f_j a_{i+j}$.

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

The minimal polynomial of a matrix $\mathbf{A} \in \mathbb{F}^{n\times n}$ is the least degree polynomial $m_{\mathbf{A}}$ over $\mathbb{F}$ such that $m_{\mathbf{A}}(\mathbf{A}) = 0$.

We denote the minimal polynomial for the sequence $\mathfrak{a}' = (\mathbf{u}^T \mathbf{A}^i \mathbf{v})_{i\in\mathbb{N}}$ by $m_{\mathfrak{a}'}$, where $\mathbf{u}, \mathbf{v} \leftarrow_{\$} \mathbb{F}^n$ are uniformly chosen vectors.

The following lemma is rephrased from [KMWF07] and shows how we can compute the minimal polynomial of a matrix A.

Lemma 6 ([KMWF07])**.** *Let $\mathbf{A} \in \mathbb{F}^{n\times n}$ and let $m_{\mathbf{A}}$ be the minimal polynomial of matrix $\mathbf{A}$. For $\mathbf{u}, \mathbf{v} \leftarrow_{\$} \mathbb{F}^n$, we have $m_{\mathbf{A}} = m_{\mathfrak{a}'}$ with probability at least $1 - 2\deg(m_{\mathbf{A}})/|\mathbb{F}|$. Moreover, $m_{\mathfrak{a}'}$ can be calculated using a Boolean circuit of size $\mathcal{O}(nk \log n \log k \log\log k)$ where $k = \log|\mathbb{F}|$*

A.2.2 Compute the Rank of a Matrix and Solve a Linear System

We will use the following results from [KDS91]. Recall that a unit upper (resp., lower) triangular Toeplitz matrix is an upper (resp., lower) triangular Toeplitz matrix 1's in the diagonal.

Lemma 7 ([KDS91])**.** *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 ([KDS91])**.** *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 $1 - n^2/|\mathbb{F}|$.*

To solve a linear system $\mathbf{Mx} = \mathbf{y}$, we follow the method of Kiltz et al. [KMWF07] which is based on Kaltofen and Saunders's algorithm [KDS91]. We briefly describe the algorithm here: (i) Perturb the linear system $\mathbf{Mx} = \mathbf{y}$ to obtain a new system $\mathbf{M}'\mathbf{x} = \mathbf{y}'$ with the same solution space. The perturbation has the property that, if $\mathbf{M}$ is of rank r, then $\mathbf{M}'_r$, the top-left $r \times r$ sub-matrix of $\mathbf{M}'$, is non-singular, except with negligible probability. (ii) Pick a random vector $\mathbf{u} \in \mathbb{F}^n$ and set $\mathbf{y}'_r$ to be the first r coordinates of the vector $\mathbf{y}' + \mathbf{M}'\mathbf{u}$. (iii) Solve the linear system $\mathbf{M}'_r \mathbf{x}_r = \mathbf{y}'_r$, and denote the solution by $\mathbf{u}_r$. (iv) Let $\mathbf{u}^* \in \mathbb{F}^n$ be a vector with the first r coordinates $\mathbf{u}_r$ and the remaining coordinates 0^{n-r}. It can be shown that $\mathbf{x} = \mathbf{u}^* - \mathbf{u}$ is a uniformly random solution of the system $\mathbf{M}'\mathbf{x} = \mathbf{y}'$ and thus is a uniformly random solution of the original system.

Appendix 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 $\mathsf{secMult}$

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

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

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

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

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

Lemma 9 (Correctness)**.** *The protocol* $\mathsf{secMult}$ *is correct.*

Proof. The value outputted by every party is

$$\begin{aligned}
e &= \tilde{d} - \sum_j d_l^{(j)} - \sum_j d_r^{(j)} - \sum_j \tilde{c}^{(j)} \\
&= \mathsf{Enc}\left(\mathsf{pk}, \mathbf{M}_l' \cdot \mathbf{M}_r' - \sum_i \mathbf{R}_l^{(i)} \cdot \mathbf{M}_r - \sum_i \mathbf{M}_l \cdot \mathbf{R}_r^{(i)} - \sum_j \sum_{k \neq j} \mathbf{R}_l^{(i)} \cdot \mathbf{R}_r^{(j)}\right) \\
&= \mathsf{Enc}\left(\mathsf{pk}, \left(\mathbf{M}_l + \sum_j \mathbf{R}_l^{(j)}\right)\left(\mathbf{M}_r + \sum_j \mathbf{R}_r^{(j)}\right) - \sum_i \mathbf{R}_l^{(i)} \cdot \mathbf{M}_r - \right. \\
&\qquad \left. \sum_i \mathbf{M}_l \cdot \mathbf{R}_r^{(i)} - \sum_j \sum_{k \neq j} \mathbf{R}_l^{(i)} \cdot \mathbf{R}_r^{(j)}\right) \\
&= \mathsf{Enc}(\mathsf{pk}, \mathbf{M}_l \cdot \mathbf{M}_r).
\end{aligned}$$

So, every party outputs exactly an encryption of the matrix product $\mathbf{M}_l \cdot \mathbf{M}_r$. □

Lemma 10 (Security)**.** *The protocol* $\mathsf{secMult}$ *securely EUC-realizes* $\mathcal{F}_{\mathsf{OMM}}$ *with shared ideal functionality* $\mathcal{F}_{\mathsf{Gen}}$ *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 $\mathsf{Enc}(\mathsf{pk}, \mathbf{M}_l \cdot \mathbf{M}_r)$, 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 $\mathbf{M}'_r$. □

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 now present the Protocol 5 to compute the rank of an encrypted matrix.

Protocol 5 Secure Rank secRank

Setup: Each party has a secret key share sk_i for a public key pk of a TPKE $\mathsf{TPKE} = (\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$. The parties have access to the oblivious matrix multiplication ideal functionality $\mathcal{F}_{\mathsf{OMM}}$.

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

1: Each party P_i broadcasts an encrypted uniformly chosen at random unit upper and lower triangular Toeplitz matrices $\mathsf{Enc}(\mathsf{pk}, \mathbf{U}_i)$ and $\mathsf{Enc}(\mathsf{pk}, \mathbf{L}_i)$ and a uniformly chosen at random diagonal matrix $\mathsf{Enc}(\mathsf{pk}, \mathbf{X}_i)$, where $\mathbf{U}_i, \mathbf{L}_i \in \mathbb{F}^{t \times t}$ and $\mathbf{X}_i \in \mathbb{F}^{t \times t}$.

2: Each party P_i computes: i) $\mathsf{Enc}(\mathsf{pk}, \mathbf{X}) = \sum_i \mathsf{Enc}(\mathsf{pk}, \mathbf{X}_i)$, ii) $\mathsf{Enc}(\mathsf{pk}, \mathbf{U}) = \sum_i \mathsf{Enc}(\mathsf{pk}, (\sum_i \mathbf{U}_i) - (N - 1)\mathbf{I})$, and iii) $\mathsf{Enc}(\mathsf{pk}, \mathbf{L}) = \mathsf{Enc}(\mathsf{pk}, (\sum_i \mathbf{L}_i) - (N-1)\mathbf{I})$, where $\mathbf{I}$ is the identity matrix.

3: All parties mutually compute $\mathsf{Enc}(\mathsf{pk}, \mathbf{N}) = \mathsf{Enc}(\mathsf{pk}, \mathbf{XUML})$ via three invocations of $\mathcal{F}_{\mathsf{OMM}}$.

4: Each party P_i samples $\mathbf{u}_i, \mathbf{v}_i \leftarrow_\$ \mathbb{F}^t$ and broadcasts $\mathsf{Enc}(\mathsf{pk}, \mathbf{u}_i), \mathsf{Enc}(\mathsf{pk}, \mathbf{v}_i)$.

5: Each party P_i computes $\mathsf{Enc}(\mathsf{pk}, \mathbf{u}) = \sum_j \mathsf{Enc}(\mathsf{pk}, \mathbf{u}_j)$ and $\mathsf{Enc}(\mathsf{pk}, \mathbf{v}) = \sum_j \mathsf{Enc}(\mathsf{pk}, \mathbf{v}_j)$. Then, it computes the sequence $\mathsf{Enc}(\mathfrak{a})$ with $2\log t$ invocations of $\mathcal{F}_{\mathsf{OMM}}$,[14]where $\mathfrak{a} = \{\mathbf{a}_0, \ldots, \mathbf{a}_{2t-1}\}$ and $\mathsf{Enc}(\mathsf{pk}, \mathbf{a}_j) = \mathsf{Enc}(\mathsf{pk}, \mathbf{u}\mathbf{N}^j\mathbf{v})$ for $0 \leq j \leq 2t-1$.

6: All parties mutually compute $\mathsf{Enc}(\mathsf{pk}, r-1)$ where r is the degree of $m_{\mathfrak{a}}$, the minimal polynomial of the (encrypted) sequence $\mathsf{Enc}(\mathfrak{a})$. This can be calculated using a Boolean circuit with size $O(t^2 k \log t)$ (which can be securely constructed from TPKE [ST06]).

Analysis. We analyze the correctness and security of the protocol.

Lemma 11 (Correctness)**.** *The protocol* secRank *is correct.*

Proof. The correctness of the protocol is guaranteed by Lemma 7 and Lemma 8. □

Lemma 12 (Security)**.** *The protocol* secRank *securely EUC-realizes* $\mathcal{F}_{\mathsf{ORank}}$ *with shared ideal functionality* $\mathcal{F}_{\mathsf{Gen}}$ *in the* $\mathcal{F}_{\mathsf{OMM}}$*-hybrid model against semi-honest adversaries corrupting up to* $N-1$ *parties, given that* TPKE *is IND-CPA.*

Proof (Sketch). The simulator takes the corrupted parties input, sends them to the ideal functionality and simulates the protocol as the honest parties would do. It is easy to see that, even when the adversary corrupts $N-1$ parties, the information is hidden by the TPKE and thus no information on $\mathbf{M}$ is leaked to the adversary by the IND-CPA of the underlying TPKE. □

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 $\mathfrak{a}$ and this can be implemented with communication cost of $\mathcal{O}(t^2 k \log t)$ [KMWF07]. Assuming a broadcast channel, the communication complexity is $\tilde{\mathcal{O}}(Nt^2)$

[14]We can perform t multiplications in $\mathcal{O}(\log t)$ calls to $\mathcal{F}_{\mathsf{OMM}}$ by performing multiplications in a batched fashion [KMWF07].

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 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.

> **$\mathcal{F}_{\mathsf{OInv}}$ functionality**
>
> **Parameters:** $\mathsf{sid}, N, q, t \in \mathbb{N}$ and $\mathbb{F}$, where $\mathbb{F}$ is a field of order q, known to the N parties involved in the protocol. pk public-key of a threshold PKE scheme.
>
> - Upon receiving $(\mathsf{sid}, \mathsf{P}_1, \mathsf{Enc}(\mathsf{pk}, \mathbf{M}))$ from party P_1 (where $\mathbf{M} \in \mathbb{F}^{t \times t}$ is a non-singular matrix), $\mathcal{F}_{\mathsf{ORank}}$ outputs $\mathsf{Enc}(\mathsf{pk}, \mathbf{M}^{-1})$ to P_1 and $(\mathsf{Enc}(\mathsf{pk}, \mathbf{M}), \mathsf{Enc}(\mathsf{pk}, \mathbf{M}^{-1}))$ to all other parties P_i, for $i = 2, \ldots, N$.

Protocol. We now describe the Protocol 6 that allows N parties to jointly compute the encryption of the inverse of a matrix, given that this matrix is non-singular.

Protocol 6 Secure Matrix Invert secInv

Setup: Each party has a secret key share sk_i for a public key pk of a TPKE $\mathsf{TPKE} = (\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$.
Input: Party P_1 inputs $\mathsf{Enc}(\mathsf{pk}, \mathbf{M})$ where $\mathbf{M} \in \mathbb{F}^{t \times t}$ is a non-singular matrix.

Each party P_i samples a non-singular matrix $\mathbf{R}_i \leftarrow_{\$} \mathbb{F}^{t \times t}$.
Set $\mathsf{Enc}(\mathsf{pk}, \mathbf{M}') := \mathsf{Enc}(\mathsf{pk}, \mathbf{M})$.
for i from 1 to N **do**
  P_i calculates $\mathsf{Enc}(\mathsf{pk}, \mathbf{M}') = \mathsf{Enc}(\mathsf{pk}, \mathbf{R}_i \mathbf{M}')$
  P_i broadcasts $\mathsf{Enc}(\mathsf{pk}, \mathbf{M}')$.
end for
All parties mutually decrypt the final $\mathsf{Enc}(\mathsf{pk}, \mathbf{M}')$. Then they compute its inverse to obtain $\mathsf{Enc}(\mathsf{pk}, \mathbf{N}') = \mathsf{Enc}(\mathsf{pk}, \mathbf{M}'^{-1} \prod_i \mathbf{R}_i^{-1})$.
for i from N to 1 **do**
  P_i computes $\mathsf{Enc}(\mathsf{pk}, \mathbf{N}') = \mathsf{Enc}(\mathsf{pk}, \mathbf{N}' \mathbf{R}_i^{-1})$.
  P_i broadcasts $\mathsf{Enc}(\mathsf{pk}, \mathbf{N}')$
end for
Finally, P_1 outputs $\mathsf{Enc}(\mathsf{pk}, \mathbf{M}^{-1}) = \mathsf{Enc}(\mathsf{pk}, \mathbf{N}')$.

Analysis. The proofs of the following lemmas follow the same lines as the proofs in the analysis of $\mathsf{secMult}$ protocol. We state the lemmas but omit the proofs for briefness.

Lemma 13. *The protocol* secInv *is correct.*

Lemma 14. *The protocol* secInv *securely EUC-realizes* $\mathcal{F}_{\mathsf{OInv}}$ *with shared ideal functionality* $\mathcal{F}_{\mathsf{Gen}}$ *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 [KMWF07], 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.

> $\mathcal{F}_{\mathsf{SUR}}$ **functionality**
>
> **Parameters:** $\mathsf{sid}, N, q, t \in \mathbb{N}$ and $\mathbb{F}$, where $\mathbb{F}$ is a field of order q, known to the N parties involved in the protocol. pk public-key of a threshold PKE scheme.
>
> - Upon receiving $(\mathsf{sid}, \mathsf{P}_1, \mathsf{Enc}(\mathsf{pk}, r))$ from party P_1 (where $r \in \mathbb{F}$ and $r \leq t$), $\mathcal{F}_{\mathsf{SUR}}$ computes $(\mathsf{Enc}(\mathsf{pk}, \delta_1), \ldots, \mathsf{Enc}(\mathsf{pk}, \delta_t))$ such that $\delta_i = 1$ if $i \leq r$, and $\delta_i = 0$ otherwise. The functionality outputs $(\mathsf{Enc}(\mathsf{pk}, \delta_1), \ldots, \mathsf{Enc}(\mathsf{pk}, \delta_t))$ to P_1 and $(\mathsf{Enc}(\mathsf{pk}, r), (\mathsf{Enc}(\mathsf{pk}, \delta_1), \ldots, \mathsf{Enc}(\mathsf{pk}, \delta_t)))$ to all other parties P_i, for $i = 2, \ldots, N$.

Protocol. A protocol for secure unary representation can be implemented with the help of a binary-conversion protocol [ST06]. That is, given $\mathsf{Enc}(\mathsf{pk}, r)$, all parties jointly compute $\mathsf{Enc}(\mathsf{pk}, \delta_i)$, 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 $O(Nr \log t)$.

B.5 Solve a Linear System

Protocol. We now present the Protocol 7 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 $\mathsf{secRank}$ protocol).

Analysis.

Lemma 15 (Correctness)**.** *The protocol* secLS *is correct.*

Proof. The proof follows directly from [KDS91, KMWF07]. □

Lemma 16. *The protocol* secLS *securely EUC-realizes* $\mathcal{F}_{\mathsf{OLS}}$ *with shared ideal functionality* $\mathcal{F}_{\mathsf{Gen}}$ *in the* $(\mathcal{F}_{\mathsf{ORank}}, \mathcal{F}_{\mathsf{OInv}}, \mathcal{F}_{\mathsf{SUR}})$*-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 $\tilde{\mathcal{O}}(t^2)$.

Protocol 7 Secure Linear Solve secLS

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

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

All parties jointly compute an encryption of the rank $\mathsf{Enc}(\mathsf{pk}, r)$ of $\mathbf{M}$ via the ideal functionality $\mathcal{F}_{\mathsf{ORank}}$.

Set $\mathsf{Enc}(\mathsf{pk}, \mathbf{M}') := \mathsf{Enc}(\mathsf{pk}, \mathbf{M})$ and $\mathsf{Enc}(\mathsf{pk}, \mathbf{y}') := \mathsf{Enc}(\mathsf{pk}, \mathbf{y})$.

for i from 1 to N **do**

    P_i samples two non-singular matrices $\mathbf{R}_i, \mathbf{Q}_i$ from $\mathbb{F}^{t \times t}$. It calculates $\mathsf{Enc}(\mathsf{pk}, \mathbf{M}') = \mathsf{Enc}(\mathsf{pk}, \mathbf{R}_i \mathbf{M}' \mathbf{Q}_i)$ and $\mathsf{Enc}(\mathsf{pk}, \mathbf{y}') = \mathsf{Enc}(\mathsf{pk}, \mathbf{R}_i \mathbf{y}')$. P_i broadcasts $\mathsf{Enc}(\mathsf{pk}, \mathbf{M}'), \mathsf{Enc}(\mathsf{pk}, \mathbf{y}')$.

end for

All the parties jointly compute $\mathsf{Enc}(\delta_1), \ldots, \mathsf{Enc}(\delta_t)$ by invoking $\mathcal{F}_{\mathsf{SUR}}$ on input $\mathsf{Enc}(\mathsf{pk}, r)$. They set $\mathsf{Enc}(\mathsf{pk}, \Delta) := \mathsf{Enc}\left(\mathsf{pk}, \begin{bmatrix} \delta_1 & \ldots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \ldots & \delta_t \end{bmatrix}\right)$. Finally, they compute $\mathsf{Enc}(\mathsf{pk}, \mathbf{N}) := \mathsf{Enc}(\mathsf{pk}, \mathbf{M}' \cdot \Delta + \mathbf{I}_t - \Delta)$, where $\mathbf{I}_t \in \mathbb{F}^{t \times t}$ is the identity matrix.

All the parties jointly compute $\mathsf{Enc}(\mathbf{N}^{-1})$ by invoking $\mathcal{F}_{\mathsf{OInv}}$ on input $\mathsf{Enc}(\mathsf{pk}, \mathbf{N})$.

Each party P^i samples $\mathbf{u}_i \leftarrow_{\$} \mathbb{F}^t$ and broadcasts $(\mathsf{Enc}(\mathsf{pk}, \mathbf{M}'\mathbf{u}_i), \mathsf{Enc}(\mathsf{pk}, \mathbf{u}_i))$.

All parties jointly compute $\mathsf{Enc}(\mathsf{pk}, \mathbf{u}') = \mathsf{Enc}(\mathsf{pk}, \mathbf{N}^{-1}\mathbf{y}'_r)$ by invoking $\mathcal{F}_{\mathsf{OMM}}$, where $\mathsf{Enc}(\mathsf{pk}, \mathbf{y}'_r) = \mathsf{Enc}(\mathsf{pk}, (\mathbf{y}' + \sum_j \mathbf{M}'\mathbf{u}_j)\Delta)$. Then they set $\mathsf{Enc}(\mathsf{pk}, \mathbf{x}) = \mathsf{Enc}(\mathsf{pk}, (\sum_j \mathbf{u}_j) - \mathbf{u}')$.

for i from N to 1 **do**

    P_i calculates $\mathsf{Enc}(\mathsf{pk}, \mathbf{x}) = \mathsf{Enc}(\mathsf{pk}, \mathbf{Q}_i^{-1}\mathbf{x})$. P_i broadcasts $\mathsf{Enc}(\mathsf{pk}, \mathbf{x})$.

end for

P_1 outputs $\mathsf{Enc}(\mathsf{pk}, \mathbf{x})$.